

Sterling Standards Library



# Services and Adapters

*Version 7.0*



Sterling Standards Library



# Services and Adapters

*Version 7.0*

**Note**

Before using this information and the product it supports, read the information in "Notices" on page 301.

**Copyright**

This edition applies to Version 7 Release 0 of Sterling Standards Library and to all subsequent releases and modifications until otherwise indicated in new editions.

© **Copyright IBM Corporation 2000, 2012.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

---

## Contents

Chapter 1. ACH Deenvelope Service . . . 1	Chapter 22. EDIINT MDN Building Service . . . . . 107
Chapter 2. ACH Envelope Service. . . . 5	Chapter 23. EDIINT Message Service 115
Chapter 3. ACH Return Generation Service . . . . . 9	Chapter 24. EDIINT Pipeline Service 123
Chapter 4. AS3 Build Service . . . . . 11	Chapter 25. Generic Deenvelope Service . . . . . 133
Chapter 5. AS3 Parse Service . . . . . 15	Chapter 26. Generic Envelope Service 137
Chapter 6. CHIPS Adapter . . . . . 21	Chapter 27. Image Cash Letter Join Service (Build 7002 or higher) . . . . 141
Chapter 7. CHIPS Utility Service . . . . 35	Chapter 28. Image Cash Letter Split Service (Build 7002 or higher) . . . . 145
Chapter 8. CII Deenvelope Service . . . 39	Chapter 29. Map Test Service . . . . 147
Chapter 9. CII Envelope Service . . . . 41	Chapter 30. RosettaNet Message Builder Service . . . . . 151
Chapter 10. Correlation Service . . . . 43	Chapter 31. RosettaNet Message Parser Service . . . . . 153
Chapter 11. Document Extraction Service . . . . . 47	Chapter 32. RosettaNet Message Sending Service . . . . . 155
Chapter 12. EDI Deenveloping Service 57	Chapter 33. RosettaNet PIP Tracking Service . . . . . 157
Chapter 13. EDI Encoder Service (Build 7000-7001) . . . . . 61	Chapter 34. RosettaNet Profile Service 159
Chapter 14. EDI Encoder Service (Build 7002 or higher) . . . . . 67	Chapter 35. Standards Translation Service . . . . . 161
Chapter 15. EDI Enveloping Service . . 73	Chapter 36. SWIFTNet Client Service 169
Chapter 16. EDI Overdue Acknowledgment Check Service. . . . 81	Chapter 37. SWIFTNet Server Adapter (Build 5100 - 5103) . . . . . 199
Chapter 17. EDI Post Processor Service 85	Chapter 38. SWIFTNet Server Adapter (Build 5104 or higher). . . . . 237
Chapter 18. EDIFACT Deenvelope Service . . . . . 89	Chapter 39. SWIFT Reconciliation Service . . . . . 277
Chapter 19. EDIFACT Envelope Service 93	Chapter 40. Translation Service . . . 283
Chapter 20. EDIINT Acknowledge Check Service . . . . . 99	
Chapter 21. EDIINT Header Scanning Service . . . . . 103	

<b>Chapter 41. X12 Develope Service</b>	<b>291</b>	<b>Notices . . . . .</b>	<b>301</b>
<b>Chapter 42. X12 Envelope Service . .</b>	<b>295</b>		

---

## Chapter 1. ACH Deenvelope Service

The following table provides an overview of the ACH Deenvelope service:

**Note:** ACH Deenvelope service will detect if incoming ACH data contains non-numeric data in cumulative fields and end processing with an appropriate error message.

Service name	DeenvelopeACH
Graphical Process Modeler (GPM) categories	All Services, EDI > ACH
Description	Deenvelopes entry detail records for any ACH SEC code while extracting the associated addenda records
Business usage	Support for the ACH translation standard.
Usage example	This service may convert ACH data into application format and invoke a BP on it.
Preconfigured?	Yes.
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	For SEC codes that contain compressed X12 or EDIFACT messages, the appropriate deenveloping service will typically be invoked. Related BPs: ACH Deenvelop BP; EDI Deenvelope
Application requirements	None
Initiates business processes	This service can invoke a business process if the ACH batch envelope is configured to do so. There are no special business process requirements for this service; however, you cannot use the service outside a business process.
Invocation	You must configure a business process or envelope to call the predefined deenveloping business process. The event calling this service must provide the ACH file as the primary document.
Business process context considerations	Deenveloping details are written to process data.

Returned status values	<p>The Advanced Status message will indicate the de-enveloping error and the status report will give additional information about the error. If an error occurs, a parameter called "ERROR_CODE" is written to the process data. This parameter can be used by the appropriate business process to determine the type of error that occurred and take the appropriate action. If an interchange level error occurs, this error code is written to the process data of the parent business process. If a batch level error occurs, the error code is written to the process data of the business process invoked when the 820 document was extracted, or to the error business process. The ERROR_CODE parameter can have one of the following values:</p> <p>Interchange level error codes (and descriptions):</p> <ul style="list-style-type: none"> <li>• NO_PRIMARY_DOCUMENT (Primary document was not supplied to the service)</li> <li>• INVALID_ACH_HEADER (ACH Header is invalid)</li> <li>• INVALID_ACH_TRAILER (ACH Trailer is invalid)</li> </ul> <p>Batch level error codes (and descriptions):</p> <ul style="list-style-type: none"> <li>• 820_TRANSLATION_ERROR 820 (Error while translating the extracted 820)</li> <li>• ACH_820_DISCREPANCY (Discrepancy between data in ACH and extracted 820.)</li> <li>• ACH_VALIDATION_ERROR (ACH document does not conform to specification)</li> <li>• ENVELOPE_NOT_FOUND (ACH Envelope was not found)</li> <li>• AMBIGUOUS_ENVELOPES (More than one matching ACH envelopes were found)</li> <li>• SUCCESS_BPNAME_ERROR (Error determining the BP name to be invoked on extracted 820)</li> <li>• DUPLICATE_FILEID_MODIFIER (File ID Modifier of the ACH document was previously recorded.)</li> </ul>
Restrictions	<p>The ACH Deenvelope supports ACH batch level rejection as follows: if an incoming message contains multiple batches, but some batches are non-compliant, only those non-compliant batches are rejected. The remaining (compliant) batches are processed.</p>
Persistence level	<p>Full persistence (default)</p>
Testing considerations	<p>Need a valid version ACH message. Need ACH file level and batch level inbound envelopes created. For debug messages to be logged, the system log needs to be turned on.</p>



## Parameters Passed from Service to Business Process

Parameter Name	Description
ExtractedFields	Batch header record fields are put into "Extracted Fields/ACHBatchHeaderRecord". File header records are put into "ExtractedFields/ACHFileHeaderRecord"

## Business Process Example

This process de-envelopes the ACH interchange contained in the primary document. It then writes the resultant X12 820 message to the file system.

```
<process
name="ACHDeEnvelopeTest">
  <sequence>
    <operation
name="ACHDeenvelope" >
      <participant
name="InvokeSubProcessService" />
      <output
message="Xout" >
        <assign to="INVOKE_MODE">INLINE</assign>
        <assign
to="WFD_NAME" >ACHDeenvelope</assign>
      </output>
      <input
message="Xin" >
        <assign to="." from="*"></assign>
      </input>
    </operation>
    <operation>
      <participant
name="EDITEST"/>
      <output message="FileSystemInputMessage">
        <assign
to="." from="*"></assign>
        <assign
to="Action">FS_EXTRACT</assign>
        <assign to="extractionFolder">
/ais_local/share/kwedinger/sandbox/woodstock2/tests/
scripts/edi/ach/resultdata/</assign>
        <assign
to="assignFilename">>true</assign>
        <assign
to="assignedFilename">ACHDeenvelopeTest.out</assign>
      </output>
      <input
message="inmsg">
        <assign to="." from="*"></assign>
      </input>
    </operation>
  </sequence>
</process>
```



## Chapter 2. ACH Envelope Service

The following table provides an overview of the ACH Envelope service:

Service name	EnvelopeACH
Graphical Process Modeler (GPM) categories	All Services, EDI > ACH
Description	Envelope ACH messages including any associated addenda records.
Business usage	Support for the ACH translation standard in the Sterling Platform.
Usage example	Allows the system to envelope an ACH message, including any addenda records which may be in the ACH or other ACH approved standards syntax, using specified envelopes.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	Available for all platforms.
Related services	EDI Envelope ServiceBPs: EDI Envelope BPEDIEncoder service should be called first to get the enveloping parameters.
Application requirements	None
Initiates business processes?	This service can invoke a business process if the selected ACH outbound envelope is configured to do so. There are no special business process requirements for this service. This service cannot be used outside a business process.
Invocation	If there is an associated addenda doc, then the BP calling this service must provide the data necessary to generate the addenda doc message inside the primary document.
Business process context considerations	None
Returned status values	Enter the possible status values that can be returned from this service. <ul style="list-style-type: none"> <li>• Status: Description</li> <li>• Error: The Advanced Status message will indicate the error and the status report will give additional information.</li> </ul>
Restrictions	None
Persistence level	Default

Testing considerations	<ul style="list-style-type: none"> <li>• Need a valid version 003020 X12 820 message to work with CCD and CTX message types. For other message types, need valid maps that will convert data from application format to ACH format. Need ACH Batch and File level outbound envelopes created. For debug messages to be logged, the system log needs to be turned on.</li> </ul>
------------------------	---

## Business Process Example

This process uses EDIEncoder to get the envelope settings. It then envelopes the data in the primary document into an ACH message. It then writes the resultant ACH to the file system.

```

<process
name="ACHEnvelopeTest1">
  <sequence>

    <operation>
      <participant
name="EDIEncoder"/>
      <output message="EDIEncoderTypeInputMessage1">

        <assign to="." from="*"></assign>

        <assign to="AcceptorLookupAlias">CIE</assign>

        <assign to="ReceiverID">11111111</assign>

        <assign to="SenderID">22222222</assign>

        <assign to="EDIStandard">ACH</assign>

      </output>
      <input message="inmsg">

        <assign to="." from="*"></assign>

      </input>
    </operation>

    <operation>
      <participant
name="EnvelopeACH"/>
      <output message="EDIEnvelopeTypeInputMessage">

        <assign to="." from="*"></assign>

      </output>
      <input message="inmsg">

        <assign to="." from="*"></assign>

      </input>
    </operation>

    <operation>
      <participant
name="EDITEST"/>

```

```

        <output message="FileSystemInputMessage">
            <assign to="." from="*"></assign>
            <assign to="Action">FS_EXTRACT</assign>
            <assign to="extractionFolder">
/ais_local/share/kwedinger/sandbox/woodstock2/tests/
scripts/edi/ach/resultdata/</assign>
            <assign to="assignFilename">true</assign>
            <assign to="assignedFilename">ACHEnvelopeTest1.out</assign>
        </output>
        <input message="inmsg">
            <assign to="." from="*"></assign>
        </input>
    </operation>

</sequence>
</process>

```

## Using Wildcards in Enveloping

As a way to help reduce the number of envelopes you need to create and use, the ACH Envelope service supports use of an asterisk (\*) as a wildcard character in mandatory envelope fields. By using wildcards, you can create one set of envelopes that can be used for multiple trading partners. Then, when the ACH envelope service runs, it will replace the wildcards with correlation values. If certain trading partners have specific requirements, you can still have envelopes that pertain just to them, and the ACH Envelope service chooses the envelope that is the best match. In other words, the envelope that has the most matches to specific fields in the data (for example Receiver ID, Receiver ID Qualifier), is the one selected.

The following list contains the correlation values that need to be set inside of process data in order to support wildcards:

- ACHEnvelopeParms/Out\_DestinationIdentification
- ACHEnvelopeParms/Out\_OriginIdentification
- ACHEnvelopeParms/Out\_DestinationName
- ACHEnvelopeParms/Out\_OriginName
- ACHEnvelopeParms/Out\_CompanyDiscretionaryData
- ACHEnvelopeParms/Out\_DiscretionaryData
- ACHEnvelopeParms/Out\_ReferenceCode

The following example shows how you might set correlation values in a business process:

```

<!-- Set up generic envelope correlation
data -->

<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_DestinationIdentification">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_DestinationIdentification">
    11111111</assign>

```

```

<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_OriginIdentification">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_OriginIdentification">
    222222222</assign>

<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_DestinationName">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_DestinationName">
    WildcardDestName</assign>
    <assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_OriginName">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_OriginName">
    WildcardOriginName</assign>

<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_ServiceClassCode">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_ServiceClassCode">
    999</assign>
    <assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_CompanyDiscretionaryData">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_CompanyDiscretionaryData">
    WildcardCDD</assign>

<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_DiscretionaryData">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_DiscretionaryData">
    WC</assign>

<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_ReferenceCode">
<assign name="Assign" to="/ProcessData/ACHEnvelopeParms/
Out_ReferenceCode">
    RefCode</assign>

```

**Note:** All EDI services assign a Unique ID to each log message.

## Adding Translation Map Name to Process Data

The ACH Envelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The ACH Envelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.

## Chapter 3. ACH Return Generation Service

The following table provides an overview of the ACH Return Generation service:

Service name	ReturnGenerationACH
Graphical Process Modeler (GPM) categories	All Services, EDI > ACH
Description	Generates ACH Return entry detail records and forwards them to ACH Enveloping service.
Business usage	Support for the ACH translation standard.
Usage example	Allows the back end system to return those entry detail records which could not be posted successfully.
Preconfigured?	Yes.
Requires third party files?	No
Platform availability	All supported platforms.
Related services	EDI Envelope Service Related Business Processes: EDI Envelope BP
Application requirements	None
Initiates business processes	There are no special business process requirements for this service. This service cannot be used outside a BP.
Invocation	The service should be called with an XML document as the primary document. This XML document should conform to a schema published by Sterling Commerce along with other ACH components. This message should contain information necessary to identify the entry detail records being returned and should also contain ACH specific reasons for rejection of those entry detail records.
Business process context considerations	None.
Returned status values	Enter the possible status values that can be returned from this service. Error: The Advanced Status message will indicate the error and the status report will give additional information.
Restrictions	None.
Persistence Level	Default.
Testing Considerations	The system needs to have processed some ACH Entries. Need a valid XML file referencing these entries as the input. Also need to have outbound envelopes configured so that the generated return entries can be packaged into an ACH interchange. For debug messages to be logged, the system log needs to be turned on.

## Business Process Example

This sample business process calls the ACH Return Generation Service, using the supplied document as the primary document.

```
<process name="ACHReturnGeneration">
  <sequence>
    <operation
      name="ReturnGenerationACH">
      <participant
        name="ReturnGenerationACH"/>
      <output
        message="Xout" >
          <assign to="." from="*"></assign>
        </output>
      <input
        message="Xin" >
          <assign to="." from="*"></assign>
        </input>
      </operation>
    </sequence>
  </process>
```



---

## Chapter 4. AS3 Build Service

The following table provides an overview of the AS3 Build service:

System name	AS3 Build Service.
Graphical Process Modeler (GPM) categories	All Services, INTERNETB2B-EDIINT.
Description	This service is used to build an AS3 message, including constructing the message header.
Business usage	A user needs to build an AS3 message and initiate an AS3 message exchange with a trading partner.
Usage example	An example of the usage of this service is as follows: <ol style="list-style-type: none"><li>1. The user wants to initiate an AS3 message exchange with a trading partner.</li><li>2. The user writes a business process and includes the AS3 Build service in the process definition.</li><li>3. The AS3 Build service starts the AS3 message exchange as configured in the trading partner contract.</li></ol>
Preconfigured?	Yes, a preconfigured instance, AS3BuildService, is created during installation process.
Requires third party files?	No.
Platform availability	All supported application platforms.
Related services	This service is used in conjunction with the AS3 Parse service, File System adapter, and FTP adapter.
Application requirements	An AS3 message exchange agreement is configured with an external trading partner. A trading partner contract must be configured to initiate the AS3 message exchange.
Initiates business processes?	This service initiates the AS3MessageInitiation internal business process.
Business process context considerations	None.
Invocation	This service is invoked from a business process. It can be used in any business process for which this functionality is desired.
Restrictions	No.
Returned status values	Success, Failure
Testing considerations	Debug information for this service can be found in the AS3 log files.

## How the AS3 Build Service Works

The following steps summarize how the AS3 Build service works within a business process:

1. The AS3 Build service starts the AS3 message exchange as configured in the trading partner contract.
2. The AS3 Build service initiates a message exchange with the trading partner as specified in the AS3 contract.
3. The AS3 message is exchanged based on the contract configuration.
4. For Bulk Message Generation, a scheduled business process is created. This business process has an instance of a File System adapter that collects the documents from the specified folder and invokes a business process containing the AS3 Build service for each document collected.

## Implementing the AS3 Build Service

To implement the AS3 Build service, complete the following tasks:

1. Install the AS3 Build service. See *Managing Services and Adapters*.
2. Create an AS3 Build service configuration.
3. Configure the AS3 Build service only once in the user interface.
4. Configure the AS3 Build service parameters only once in the GPM.
5. Use the AS3 Build service in a business process or create the one necessary AS3 trading partner contract with the Bulk Message Generation parameter enabled so a scheduled business process that includes the AS3 Build service is created.

**Note:** Two AS3 contracts are automatically created when you use the AS3 Partner Wizard to create a trading partner contract. The second contract is used by the AS3 Parse Service when parsing incoming AS3 messages.

## Configuring the AS3 Build Service

To configure the AS3 Build service, you must complete the following steps:

1. Select **Deployment > Services > Configuration**.
2. Search for AS3 Build service or select it from the list and click **Go!**.
3. Click **Edit**.
4. Specify field settings in the Admin Console (*Creating or Setting Up a Service Configuration in the Admin Console*) and the GPM (*Setting Up the Service in the GPM*).
5. On the Confirm page, verify that the **Enable Service for Business Processes** check box is selected and click **Finish**.

## Creating or Setting Up a Service Configuration in the Admin Console

To configure the AS3 Build service, you must specify settings for the following fields in the application user interface one time only. Additionally, you will need to specify settings in the GPM:

Field	Description
Name	Unique and meaningful name for the service configuration. Required.

Field	Description
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	<p>Select one of the options:</p> <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul> <p>Optional.  <b>Note:</b> See <i>Managing Services and Adapters</i> for more information.</p>

## Setting Up the Service in the GPM

Use the field definitions in the following table to set up the service configuration in the GPM:

Parameter	Description
ContractName	The contract to use when initiating an AS3 message exchange. Value is any valid contract name. Required. BPML element value is <b>ContractName</b> .
CustomizedBPML	The customized business process to invoke. Value is any valid application business process name. By default, the predefined (internal) AS3MessageInitiation business process is invoked. Optional.
BPInvokeMode	The business process invocation mode. Valid values are ASYNC (asynchronous) or SYNC (synchronous). Default is ASYNC. Optional.
MessageStoreDirectory	The directory in which to store the AS3 message that is created or picked up from the trading partner. Value is any valid file directory. Optional.
FileSystemInstance	The File System adapter instance that will write the AS3 message to the file system. Value is any File System adapter instance. Optional.
FTPAdapterName	The FTP adapter instance to use to transport the AS3 message to the trading partner. Value is any valid FTP adapter instance. Default is FTPClientAdapter. Optional.
PipelineTimeoutInSecs	The timeout value for building an AS3 message. Valid value is any numeric value. Default is 1800 seconds. Optional.

## Process Data Example

This example shows an example business process using the AS3 Build service to exchange AS3 messages:

```
<process name="test_AS3Build">

  <sequence name="seq1">
    <operation
      name="AS3 BUILD SERVICE">
      <participant
        name="AS3Build"/>
      <output message="outmsg">

        <assign to="ContractName">SendToCleo</assign>

      </output>
      <input message="inmsg">

        <assign to="." from="*"></assign>

      </input>
    </operation>

  </sequence>
</process>
```

## Chapter 5. AS3 Parse Service

The following table provides an overview of the AS3 Parse service:

System name	AS3 Parse Service.
Graphical Process Modeler (GPM) categories	All Services, INTERNETB2B-EDIINT.
Description	This service is used to parse an AS3 message.
Business usage	A user needs to parse an AS3 message during AS3 message exchange.
Usage example	<p>An example of the usage of this service is as follows:</p> <ol style="list-style-type: none"> <li>1. The user wants to send a Message Disposition Notification (MDN) to a trading partner.</li> <li>2. The AS3 Parse service extracts the payload from the AS3 message.</li> <li>3. The AS3 Parse service generates an MDN for the received AS3 message.</li> <li>4. The AS3 Parse service invokes a system business process to send the MDN to the trading partner.</li> <li>5. If the MDN is sent successfully, the AS3 Parse service places the payload into a PAYLOAD mailbox.</li> </ol> <p>If the MDN is not sent successfully, the AS3 Parse service places the payload into an ERROR mailbox.</p>
Preconfigured?	Yes, a default instance, AS3ParseService, is created during the installation process.
Requires third party files?	No.
Platform availability	All supported application platforms.
Related services	This service is used in conjunction with the AS3 Build service and the FTP adapter.
Application requirements	An AS3 message exchange agreement is specified with an external trading partner. The trading partner contract must be configured (in the application) to respond to the AS3 message exchange.
Initiates business processes?	This service initiates the AS3MDNInitiation internal business process.
Business process context considerations	None.
Invocation	This service is invoked by the AS3MbxProcessing internal business process.
Restrictions	No.
Returned status values	Success, Failure.
Testing considerations	Debug information for this service is located in the AS3 log files.

## How the AS3 Parse Service Works

The AS3 Parse service parses AS3 messages during an AS3 message exchange. The application gives you two options in delivering MDNs to your trading partners:

1. Send the MDN directly to the trading partner.
2. Place the MDN in an OUTGOING mailbox for the trading partner to pick it up within a specific time period. Default 60 minutes.

The following steps summarize how the AS3 Parse service works to send an MDN directly to a trading partner:

1. The AS3 Parse service extracts the payload from the AS3 message.
2. The AS3 Parse service generates an MDN for the received AS3 message.
3. The AS3 Parse service invokes a system business process to send the MDN to the trading partner.
4. If the MDN is sent successfully, the AS3 Parse service places the payload into a PAYLOAD mailbox.

If the MDN is not sent successfully, the AS3 Parse service places the payload into an ERROR mailbox.

The following steps summarize how the AS3 Parse service works to place an MDN in an OUTGOING mailbox for the trading partner to pick it up within a specific time period:

1. The AS3 Parse service extracts the payload from the AS3 message.
2. The AS3 Parse service generates an MDN for the received AS3 message.
3. The AS3 Parse service places the payload into a HOLDING mailbox and the MDN into the MDN\_OUTGOING mailbox.
4. The AS3 Parse service invokes a system business process to monitor if the MDN has been picked up by the trading partner within a specified time period (the maximum time permitted is 60 minutes).
5. If the MDN is picked up within the timeout period, the AS3 Parse service moves the payload into the PAYLOAD mailbox.

If the MDN is not picked up within the timeout period, the AS3 Parse service removes the MDN from the MDN\_OUTGOING mailbox and moves the payload into the ERROR mailbox.

**Note:** When you use the Mailbox Auto Creation option when creating AS3 partner profiles, this feature automatically creates the necessary mailboxes and routing rule to process messages and MDNs. You need a routing rule for both the Incoming AS3 Message (InboundAS3) and the Incoming AS3 MDN (Inbound MDN) mailboxes. This routing rule is used to invoke predefined business processes to process the AS3 messages and MDNs. If you do not use the automatic mailbox creation option, you must create the necessary mailboxes, routing rule, and schedule, and assign the appropriate user permissions.

If you chose to have the necessary mailboxes and routing rule automatically created during the partner profile creation process, the routing rule created is named **Routing Rule created by the AS3 auto create option**. The Incoming AS3 Message (InboundAS3) mailbox and the Incoming AS3 MDN (Inbound MDN) mailbox are attached to the rule. If this routing rule already exists, the AS3 system just adds the two mailboxes to the rule. The routing rule is set up to invoke the predefined AS3MbxProcessing business process, which contains the AS3 Parse service.

Whether you create the mailboxes and routing rule automatically or manually, you need to schedule the routing rule to be run. See *Using AS3* for more information.

## Implementing the AS3 Parse Service

To implement the AS3 Parse service, complete the following tasks:

1. Install the AS3 Parse service. See *Managing Services and Adapters*.
2. Create an AS3 Parse service configuration.
3. Configure the AS3 Parse service only once in the user interface
4. Configure the AS3 Parse service parameters only once in the GPM.
5. Use the AS3 Parse service in a business process or, if you are using the predefined internal AS3MbxProcessing business process, you do not have to create another business process.

## Configuring the AS3 Parse Service

To configure the AS3 Parse service, you must complete the following steps:

1. Select **Deployment > Services > Configuration**.
2. Search for AS3 Parse service or select it from the list and click **Go!**
3. Click **Edit**.
4. Specify field settings in the Admin Console (*Creating or Setting Up a Service Configuration in the Admin Console*) and the GPM (*Setting Up the Service in the GPM*).
5. On the Confirm page, verify that the **Enable Service for Business Processes** check box is selected and click **Finish**.

## Creating or Setting Up a Service Configuration in the Admin Console

To configure the AS3 Parse service, you must specify settings for the following fields in the application user interface one time only. Additionally, you will need to specify settings in the GPM:

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.

Field	Description
Select a Group	<p>Select one of the options:</p> <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul> <p>Optional.  <b>Note:</b> See <i>Managing Services and Adapters</i> for more information.</p>

## Setting Up the Service in the GPM

Use the field definitions in the following table to set up the service configuration in the GPM:

Parameter	Description
CustomizedBPML	The customized business process to invoke. Value is any valid business process name in the system. Default is the AS3MessageInitiation business process. Optional.
MDNStoreDirectory	The directory in which to store the MDN that is created or received from a trading partner. Value is any valid directory in the system. Optional.
MessageStoreDirectory	The directory in which to store the AS3 message created or picked up by the trading partner. Value is any valid directory in the system. Optional.
FileSystemInstance	The file system adapter instance to use to write the AS3 message to the file system. Value is any valid file system adapter instance in the system. Optional.
FTPAdapterName	The FTP adapter instance to use. Value is any FTP adapter configured in the system. Default is FTPClientAdapter. Optional.
PipelineTimeoutInSecs	The time out value for building an AS3 message. Valid value is any numeric value. Default is 1800 seconds. Optional.
mdnDocument	The MDN document generated. Optional. <b>Note:</b> The MDN is only generated when it is requested by the incoming AS3 message. This Value is returned by the service if an MDN exists.



Parameter	Description
payloadDocument	The document ID of the payload document extracted from the incoming AS3 message. Optional. <b>Note:</b> This parameter is only available when an AS3 message is received. A payload is not extracted when an MDN is received.

## Process Data Example

By default, the AS3 Parse service is invoked from the AS3MbxProcessing internal business process. The AS3MbxProcessing business process is triggered by a mail routing rule when an AS3 message is received from a trading partner.

This example shows an example business process for invoking the AS3 Parse service:

```
<process name="test_AS3Parse">
  <sequence name="seq1">
    <operation
      name="AS3_PARSE_SERVICE">
      <participant
        name="AS3Parse"/>
      <output message="outmsg">
        </output>
      <input message="inmsg">
        <assign to="." from="*"></assign>
      </input>
    </operation>
  </sequence>
</process>
```



## Chapter 6. CHIPS Adapter

The CHIPS adapter sends CHIPS messages using MQ or SWIFTNet.

The following table provides an overview of the CHIPS adapter:

System Name	CHIPSAdapter
Graphical Process Modeler (GPM) categories	All Services.
Description	This adapter sends CHIPS messages using MQ or SWIFTNet to The Clearing House.
Business usage	Use this adapter to send CHIPS financial information using MQ or SWIFTNet to the CHIPS network. The business value of this adapter is inherent in using the benefits of the CHIPS network to exchange financial messages.
Usage example	A user needs to send CHIPS payment messages to The Clearing House and does so using the CHIPS adapter.
Preconfigured?	A default version of the CHIPS adapter is preconfigured.
Requires third party files?	MQSeries, com.ibm.mq.jar and mqji_en_US.properties must be installed to use MQ delivery.
Platform availability	All supported application platforms.
Related services	This adapter works with the CHIPS Utility service, the Websphere MQ Suite Async Receive adapter, the SWIFTNet HTTP Server adapter, and the SWIFTNet Server adapter.
Application requirements	None.
Initiates business processes?	Yes, either a user-created business process or the sample business process provided with the application (CHIPS business process).
Invocation	A user who has permission to perform this activity must execute the business process that invokes this adapter.
Business process context considerations	The configuration parameters are picked up by the adapter in the workflow. In receiving mode, the adapter puts the incoming documents into the workflow.
Returned status values	Success, Failure.
Restrictions	Either MQ or SWIFTNet transport can be used with the CHIPS adapter.
Persistence level	N/A
Testing considerations	To test this adapter, run the CHIPS adapter business process and verify that it completes successfully. Debug information for this service is located at: Operations > System > Logs > CHIPS

## How the CHIPS Adapter Works

The CHIPS adapter sends CHIPS messages to the CHIPS Central Computer, using either the SWIFTNet network (optionally using IBM Websphere MQ) or The Clearing House Frame Relay Network (a proprietary network that uses IBM Websphere MQ). The Clearing House provides a TCP/IP interface for communicating with CHIPS. All CHIPS messages include a message header and all message requests require a message acknowledgement. When the participant sends a message request, the participant expects a CHIPS acknowledgement or CHIPS Invalid Message Acknowledgement message. When CHIPS sends a message request, the participant returns a participant acknowledgement message. Acknowledgements are sent to CHIPS based on the transport mode for which the CHIPS Adapter is configured. When the CHIPS line is inactive, the CHIPS Adapter enables you to send supervisory STATUS messages to CHIPS to test the connection.

You must configure the CHIPS adapter prior to sending any messages to CHIPS. The message payload is passed through a business process that you must also create (or configure the predefined business process).

The CHIPS adapter uses five handlers to send and receive messages:

- The **Send Handler** sends outbound messages (if the communications method selected is functioning) and stores outbound messages in a local database. The Send Handler creates two mailboxes, based on the participant number, to handle the sending of outbound messages. It stores all the messages into the mailbox for auditing purposes, regardless of the status of CHIPS line. The Send Handler checks the line status in the application database and, if the line is functioning, it sends the message based on the transport mode (either Websphere MQ or SWIFTNet). If the line is inactive, all messages are stored in a local database table until they can be processed (when the line is functioning again, the Resend Handler batches and sends the locally stored messages in the database).
- The **Resend Handler** checks for messages that have exceeded the specified resend count. If there are no messages that have exceeded the resend count, it will collect all outbound messages that were stored locally (that is, when the messages were first sent, the selected communication method was unavailable and the messages were stored in the local database). Additionally, the Resend Handler resends messages that have not been acknowledged for more than sixty seconds. If the number of messages collected is greater than the batch number, the messages are sent. If the number of messages collected is less than the batch number, the Resend Handler continues to collect messages for the CHIPS adapter instance (for which the line status is down). In case of a payment message, this handler also includes a possible duplicate tag (PSN [271]) to indicate that the message could be a duplicate. If there are messages that have exceeded the resend count, the Resend Handler will send a STATUS supervisory message to CHIPS. All the other resend messages or new incoming messages are not sent. When the acknowledgement is received, the locally stored messages will be sent in the next time interval. If it does not receive an acknowledgement, it sets the line status to inactive.

Each time a message is resent, the Resend Handler generates a new message number for the message from the database. If the resend count of a message is greater than three (that is, the Resend Handler has attempted to send a particular message three times with no success), the Resend Handler sends an event to the Heartbeat Handler to send a supervisory message to CHIPS. If this

occurs, other resend messages or new incoming messages are not sent. You can configure the Resend Handler to start at specific intervals (for example, every ten minutes).

- The **Heartbeat Handler** is a “listener event”; it is initiated when an event is called by the Resend Handler or the business process that checks the status of CHIPS line. The Heartbeat Handler sends a STATUS supervisory message. It starts the sixty second timer and waits for the response. If there is no response, it updates the database with the status that the line is down. Otherwise, it updates the database with the status “Line up.”
- The **Receive Handler** receives all messages from CHIPS (through the communication method you specify, either MQ or SWIFTNet), sets the timestamp of each received messages in the appropriate database table, and returns an acknowledgement message if the incoming message is not the CHIPS Acknowledgement message. The Receive Handler stores all incoming messages except heartbeat messages in the appropriate mailbox and the application database table, and parses and validates the header of all incoming CHIPS messages. Heartbeat messages are available in FS\_INBOUND database table. If a message received is a Response Acknowledgement from CHIPS, the Receive Handler sets the Acknowledged flag in the appropriate database table, notes the timestamp of the acknowledgement, and decrements the MQ counter (if you are using MQ as your transport mode). If a message received is not a Response Acknowledgement from CHIPS, the Receive Handler notifies the Acknowledgement Handler to send a a participant acknowledgement response to CHIPS.
- The **Acknowledgement Handler** sends the appropriate participant acknowledgement response (transaction code 05) to CHIPS (the response is based on the incoming message). If you are using MQ transport mode, the Acknowledgement Handler uses the MQ parameters you specified for the CHIPS adapter to send the acknowledgement response. If you are using SWIFTNet transport mode, the Acknowledgement Handler encodes the acknowledgement response and sets it in the Primary Document, and this is used to return the SWIFTNet Server response.

Please note that the CHIPS adapter automatically performs the following:

- If the transport mode used is SWIFTNet, the payload *must* be base64 encoded, and the response that is received must be decoded.
- If the transport mode used is SWIFTNet, the Request Type is set to **chips.payment** if the transaction code is 10; for all other transaction codes, the Request Type is set to **chips.message**.

## How the CHIPS Adapter Communicates with SWIFTNet

When the CHIPS adapter is used with the SWIFTNet network, it receives acknowledgement messages from CHIPS in the SWIFTNet Response within sixty seconds, and any incoming messages (for example, heartbeat message, resolver notification) are received by SWIFTNet Server adapter. The return acknowledgement of the incoming messages is performed by the Receive Handler and Acknowledgement Handler within the CHIPS adapter (the business process is bootstrapped using the SWIFTNet Routing Rule).

**Note:** The SWIFTNet adapter must be preconfigured to start up the SWIFTNet MEFG Server to listen for incoming messages.

The SWIFTNet transport process handles batches of messages as a sequential request and response process.

The SWIFTNet Client service is executed to create the SWIFTNet message header based on the configuration set in the CHIPS adapter. The request type is either chips.payment (if the transaction code is 10) or chips.message (for all transaction codes except 10).

## How the CHIPS Adapter Communicates with MQ

When the CHIPS adapter is used with MQ, any acknowledgement from CHIPS and any incoming messages (for example, heartbeat message, resolver notification, and so forth) are received by the Websphere MQ Suite Async Receive adapter.

The return acknowledgement of the incoming messages is performed by the Receive Handler and Acknowledgement Handler within the CHIPS adapter (the business process is bootstrapped from the Websphere MQ Suite Async Receive adapter).

**Note:** A unique set of the MEFG Server IP, MEFG Server Port, Queue Manager, Channel Name, and Send Queue parameters can only be used in *one* CHIPS adapter configuration. Therefore, if a unique set of these parameters is used, it must not be used in any other CHIPS adapter configuration. Additionally, each line connection can only have one queue active.

The MQ transport process handles batches of messages as follows: open session, open queue, send multiple messages, close queue, and close session.

With MQ, a participant can have more than one connection through clustering, and each connection can be one set of a line configuration. Each line configuration represents a CHIPS adapter configuration, so each participant in this scenario can have up to two CHIPS adapter configurations when using MQ.

## Implementing the CHIPS Adapter

To implement the CHIPS adapter, complete the following tasks:

1. Create a configuration of the CHIPS adapter to enable you to send a CHIPS message. For information about the fields specific to this adapter, see *Configuring the CHIPS Adapter*.

**Note:** If you create a new configuration, you must also create a new business process or edit a copy of the appropriate predefined business process, to update it to use your adapter configuration. You do not need to create an instance of the CHIPS adapter for every message; you can reuse the CHIPS adapter instance and pass the parameters that differ from the sample adapter through the business process.

2. Specify field settings for the adapter configuration in the application admin console and in the GPM as necessary. See *Configuring the CHIPS Adapter*.
3. Perform the additional tasks necessary to use the CHIPS adapter. See *Additional Tasks Necessary to Use the CHIPS Adapter* for more information.
4. If you are communicating through Websphere MQ Server, configure the Websphere MQ Suite Async Receive adapter. See the *Websphere MQ Suite Async Receive Adapter* documentation for more information.
5. If you are communicating through SWIFTNet, configure the following:
  - SWIFTNet Server Adapter (see the *SWIFTNet Server Adapter* documentation for more information)

- SWIFTNet HTTP Server Adapter (see the *SWIFTNet HTTP Server Adapter* documentation for more information)
  - SWIFTNet Routing Rule S (see the *Using SWIFTNet* documentation for more information)
6. Create the necessary business process to pass the payload.
- For a single document, you must pass the payload as an input document or indicate the `document_id` as a BPML parameter.
  - For multiple documents (`document_list`), you must indicate the `document_list` and `document_id` as BPML parameters, or, if you do not enter the `document_list` in the BPML, the adapter reads (from process data) the following format for processing multiple documents, which is taken from Process Data:

```
<document_list>
<document_id>abc1</document_id>
<document_id>abc2</document_id>
<document_id>abc3</document_id>
</document_list>
```

**Note:** See *Parameters Passed From Business Process to Adapter* for more information on creating the appropriate BPML.

- Per CHIPS, the CHIPS message input should not exceed 12,200 characters.

## Configuring the CHIPS Adapter

1. Select **Deployment > Adapters > Configuration**.
2. Search for CHIPS adapter or select it from the list and click **Go!**
3. Click **Edit**.
4. Specify field settings in the Admin Console or Business Process (*Creating or Setting Up a Adapter Configuration in the Admin Console or Business Process*), or the GPM ().
5. On the Confirm page, verify that the **Enable Adapter for Business Processes** check box is selected.

## Creating or Setting Up a Adapter Configuration in the Admin Console or Business Process

Use the field definitions in the following table to create a new configuration of the CHIPS adapter, or to set up the configuration provided with the application. Some fields are available in both the Admin Console and in the GPM.

Field	Description
Name	Unique and meaningful name for the adapter configuration. Required.
Description	Meaningful description for the adapter configuration, for reference purposes. Required.

Field	Description
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – Do not include the configuration in a group at this time.</li> <li>• Create New Group – Enter a unique name for a new group, which will be created with this configuration. (You can then add other adapters to the group as well.)</li> <li>• Select Group – If adapter groups already exist for this adapter type, they are displayed in the list. Select a group from the list.</li> </ul> <b>Note:</b> See <i>Managing Adapters and Adapters</i> .
Participant Number	Participant number assigned by CHIPS (either the Sender ID or ABA number). Required. The BPML element is <b>participant_num</b> .
Number of msgs in a batch	The number of messages allowed in a batch during the resend process. The default is 40 and the number should be 40 or less to allow new incoming messages to be processed. Required. The BPML element is <b>msg_num_in_batch</b> .
Resend Count	Maximum number of times that a message may be resent. If the maximum number specified is exceeded, a supervisory message is sent to CHIPS. Default is 3. Required. The BPML element is <b>resend_count</b> .
Workflow to bootstrap during CHIPS/WebSphere MQ failure	Select the workflow to invoke if the CHIPS or MQ Server fails, to inform the administrator of the failure. Optional. The BPML element is <b>workflow_failure</b> . <b>Note:</b> This associates a customized business process that enables you to perform administrative work such as notifying an administrator through e-mail when CHIPS or MQ is down.
Environment	Whether the environment is test or production (default). Required. The BPML element is <b>environment</b> .
Transport Mode	The mode of transport to the CHIPS Network. Valid selections are WebSphere MQ Server (default) or SWIFTNet. Required. The BPML element is <b>transport_mode</b> .
WebSphere MQ Server Name	The host name of the MQ Server. Required. The BPML element is <b>mq_hostname</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>WebSphere MQ Server</b> .
WebSphere MQ Server Port No.	The port number of the MQ Server. Required. The BPML element is <b>mq_port</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>WebSphere MQ Server</b> .



Field	Description
Websphere MQ Server User ID	The MQ user identifier used to log in to the MQ server. Optional. The BPML element is <b>mq_userId</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b> .
Websphere MQ Server Password	The MQ password used to log in to the MQ server. Optional. The BPML element is <b>mq_password</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b> .
Channel Name	The channel for the MQ server (server-connection type). Required. The BPML element is <b>mq_channelName</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b> . The Channel Name, Reply-To Queue Manager, and Reply-To Queue parameters are mandatory and must be configured the same as you configure for the WebsphereMQ Async Receiver adapter. When the incoming message is received, it invokes the Receive Handler based on the three values to look up for the matched CHIPS adapter. Based on the CHIPS adapter configuration, it will then be able to send back the acknowledgement message using the same configured MQ information.
Queue Manager	The name of the queue manager. Required. The BPML element is <b>mq_qManager</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b> .
Send Queue	The name of the send queue (Remote Definition queue). Required. The BPML element is <b>mq_sendQueue</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b> .
Reply-To Queue Manager	The name of the reply-to queue manager. Required. The BPML element is <b>mq_replyToqManager</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b> . The Channel Name, Reply-To Queue Manager, and Reply-To Queue parameters are mandatory and must be configured the same as you configure for the WebsphereMQ Async Receiver adapter. When the incoming message is received, it invokes the Receive Handler based on the three values to look up for the matched CHIPS adapter. Based on the CHIPS adapter configuration, it will then be able to send back the acknowledgement message using the same configured MQ information.

Field	Description
Reply-To Queue	<p>The name of the reply-to queue. Required. The BPML element is <b>mq_qManager</b>.</p> <p><b>Note:</b> Only displayed when Transport Mode is set to <b>Websphere MQ Server</b>. The Channel Name, Reply-To Queue Manager, and Reply-To Queue parameters are mandatory and must be configured the same as you configure for the WebsphereMQ Async Receiver adapter. When the incoming message is received, it invokes the Receive Handler based on the three values to look up for the matched CHIPS adapter. Based on the CHIPS adapter configuration, it will then be able to send back the acknowledgement message using the same configured MQ information.</p>
Requestor DN	<p>Distinguished name of the requestor. Required. The BPML element is <b>swift_requestorDN</b>.</p> <p><b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation. Only displayed when Transport Mode is set to <b>SWIFTNet</b>. The Requestor DN, Responder DN, and Service Name are automatically saved in the SWIFTNet Routing Rule to allow the incoming CHIPS messages to bootstrap the Receive Handler based on the three values.</p>
Responder DN	<p>Distinguished name of the responder. Required. The BPML element is <b>swift_responderDN</b>.</p> <p><b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation. Only displayed when Transport Mode is set to <b>SWIFTNet</b>. The Requestor DN, Responder DN, and Service Name are automatically saved in the SWIFTNet Routing Rule to allow the incoming CHIPS messages to bootstrap the Receive Handler based on the three values.</p>
Service Name	<p>Name of the service to which both SWIFT correspondents have subscribed. Required. This must be a SWIFTNet service to which you are subscribed. The BPML element is <b>swift_serviceName</b>.</p> <p><b>Note:</b> Only displayed when Transport Mode is set to <b>SWIFTNet</b>. The Requestor DN, Responder DN, and Service Name are automatically saved in the SWIFTNet Routing Rule to allow the incoming CHIPS messages to bootstrap the Receive Handler based on the three values.</p>

Field	Description
SWIFTNet Operation	Whether the communication is handled in Synchronous (default) or Asynchronous mode. Required. The BPML element is <b>swift_op</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>SWIFTNet</b> .
Request Reference	User reference of the request. Optional. The BPML element is <b>swift_requestReference</b> . Only displayed when Transport Mode is set to <b>SWIFTNet</b> .
MEFG Server IP	The IP address of the SWIFTNet MEFG Server. Required. The BPML element is <b>mefg_server_ip</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>SWIFTNet</b> .
MEFG Server Port	The port of the SWIFTNet MEFG Server. Required. The BPML element is <b>mefg_server_port</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>SWIFTNet</b> .
MEFG HTTP Response Timeout	The timeout value for SWIFT to return a response. Optional. Default is 60. The BPML element is <b>mefg_response_timeout</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>SWIFTNet</b> .
Use SSL	Specify whether to use SSL between the application and MEFG Server. Valid selections are True and False (default). Required. The BPML element is <b>useSSL</b> . <b>Note:</b> Only displayed when Transport Mode is set to <b>SWIFTNet</b> .
Run As User	Identify the user who has permission to run the scheduled activity. Optional. You can type the user ID. Or you can click the button, select the user ID from the list, and click <b>Save</b> . Required. <b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.
Use 24 Hour Clock Display	By default, the scheduling wizard displays times using a 12-hour clock (which designates hours as a.m. or p.m.). Use this option to display times using a 24-hour clock. Optional. <b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.

Field	Description
Do not use schedule	<p>If you select this option, you cannot enable the schedule in the future. You must recreate the schedule instead. Use this option only when you do not need a schedule for a service or report.</p> <p><b>Note:</b> You must select one of the <b>Schedule</b> options. For this scheduling wizard, the type of schedule you select determines what options are displayed on subsequent pages.</p> <p><b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.</p>
Run based on timer	<p>Run the scheduled activity at a certain time or time interval, such as every 2 hours.</p> <p><b>Note:</b> You must select one of the <b>Schedule</b> options. For this scheduling wizard, the type of schedule you select determines what options are displayed on subsequent pages.</p> <p><b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.</p>
Run daily	<p>Run the scheduled activity one or more times every day.</p> <p><b>Note:</b> You must select one of the <b>Schedule</b> options. For this scheduling wizard, the type of schedule you select determines what options are displayed on subsequent pages.</p> <p><b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.</p>
Run based on day(s) of the week	<p>Run the scheduled activity on certain days of the week, such as every Monday.</p> <p><b>Note:</b> You must select one of the <b>Schedule</b> options. For this scheduling wizard, the type of schedule you select determines what options are displayed on subsequent pages.</p> <p><b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.</p>

Field	Description
Run based on day(s) of the month	Run the scheduled activity on certain days of the month, such as the 1st or 15th of every month. <b>Note:</b> You must select one of the <b>Schedule</b> options. For this scheduling wizard, the type of schedule you select determines what options are displayed on subsequent pages. <b>Note:</b> The scheduler is used to start the resend business process at a specified time interval (preferably every ten minutes), so that any locally stored messages or messages that have not yet been acknowledged can be sent.

## Business Process Examples

The examples in this section involve the CHIPS adapter sending these four message types:

- sendCHIPSRequest
- resendCHIPSRequest
- runSupervisoryCheck

This example sends a document using the CHIPS adapter:

```
<process
name="CHIPSadapter">
  <sequence name="CHIPSadapter">
    <operation
name="set user token">
      <participant
name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign
to="USER_TOKEN">admin</assign>
        <assign
to="." from="*"/>
      </output>
      <input
message="inmsg">
        <assign to="." from="*"/>
      </input>
    </operation>
    <operation>
      <participant
name="CHIPSAdapter"/>
      <output message="sendCHIPSRequest">
        <assign
to="." from="*"/>
        <assign to="document_id">test-doc-id</assign>
      </output>
      <input
message="testing">
        <assign to="." from="*"/>
      </input>
    </operation>
  </sequence>
</process>
```

This example does not need to be created. It is automatically created when the schedule is configured. When the CHIPS adapter is scheduled, the Schedule\_CHIPSAdapter business process is

automatically created.

```
<process name="Schedule_CHIPS_ADAPTER_MQ">
  <sequence>
    <operation
      name="Service">
      <participant name="CHIPSAdapter"/>
      <output
        message="resendCHIPSRequest">
        <assign
          to="." from="*"/>
        </output>
        <input
          message="Xin">
          <assign to="." from="*"/>
          </input>
        </operation>
      </sequence>
    </process>
```

This example sends a Supervisory STATUS message:

**Note:** In the application, the business process used to send the supervisory STATUS is CHIPSAdapter\_SupervisoryCheck.

```
<process
  name="CHIPSAdapterSupervisoryCheck">
  <sequence
    name="CHIPSAdapter">
    <operation name="set
      user token">
      <participant name="SetUserToken"/>
      <output
        message="SetUserTokenMessage">
        <assign
          to="USER_TOKEN">admin</assign>
        <assign
          to="." from="*"/>
        </output>
        <input
          message="inmsg">
          <assign to="." from="*"/>
          </input>
        </operation>
      <operation>
      <participant
        name="CHIPSAdapter"/>
      <output message="runSupervisoryCheck">
      <assign
        to="." from="*"/>
      </output>
      <input
        message="testing">
        <assign to="." from="*"/>
        </input>
      </operation>
    </sequence>
  </process>
```

## Parameters Passed From Business Process to Adapter

The following table contains the parameters passed from the business process to the CHIPS adapter:

Parameter	Description
document_list	<p>The list of documents. If you do not enter the document_list in the BPML, the adapter reads (from process data) the following format for processing multiple documents:</p> <pre>&lt;document_list&gt; &lt;document_id&gt;abc1&lt;/document_id&gt; &lt;document_id&gt;abc2&lt;/document_id&gt; &lt;document_id&gt;abc3&lt;/document_id&gt; &lt;document_list&gt;</pre> <p>Optional.</p> <p><b>Note:</b> The payload is passed using the BPML, and you can override the CHIPS adapter configuration using these BPML parameters. For example, the payload can be sent in batches. If you want to send a single document, use the document_id parameter. You can also predefine a Process Data tag name (such as DocList) that enables you to put multiple document_id parameters into the tag so the CHIPS adapter picks up all the document_ids at once.</p>
document_id	<p>This is the document identifier if the document is already present in the Document area for single document processing. If document_list or document_id is not specified in the BPML, the document is passed in as the Primary Document. If no document is found, the BPML fails.</p> <p>Optional.</p>

## Additional Tasks Necessary to Use the CHIPS Adapter

In addition to configuring the CHIPS adapter, you must also perform the following tasks:

- Create a mailbox routing rule to invoke the CHIPSExtractMailboxMessage business process.
- Enable the predefined MailboxEvaluateAllAutomaticRulesSubMin schedule (installed with the application).

## Creating the Mailbox Routing Rule

You must create a mailbox routing rule to invoke the CHIPSExtractMailboxMessage business process, which extracts each mailbox message received by the CHIPS adapter and bootstraps the EDIDevelope business process for each extracted mailbox message.

To create the necessary mailbox routing rule:

1. From the **Deployment** menu, select **Mailboxes > Routing Rules**.
2. Next to **Create a new Routing Rule** click **Go!**

3. Specify a Name for the routing rule. This name must be unique for each routing rule. It is used to identify the routing rule in other parts of the application.
4. In the Rule Application page, select **Evaluate Manually** as the **Evaluation Mode**. This specifies that the rule must be evaluated manually or evaluated using a scheduled business process.
5. For **Action Type**, accept the default **Business Process** selection. This specifies that the rule will notify a business process when a match is found.
6. Click **Next**.
7. In the Rule Pattern page, select **Filter by Name**.
8. From the **Available Mailboxes** list, select the **mailbox that contains your sender ID**, and click the single down arrow to add the mailbox to the Selected Mailboxes list.

**Note:** All groups in the Selected Mailboxes list are searched by the routing rule.

9. For **Message Name Pattern**, type **CHIPSIN\_\*** and click **Next**.

**Note:** This is the message name or pattern that the routing rule searches for in the mailboxes specified.

10. In the Rule Action page, select the **CHIPSExtractMailboxMessage** business process and click **Next**.
11. In the Run Rule as User page, select the **admin** user ID and click **Next**.
12. In the Confirm page, verify the parameters and click **Finish**.
13. When the system update is complete, click **Return**.

## Enable the Predefined Schedule

The mailbox routing rule you created above is executed automatically when the predefined MailboxEvaluateAllAutomaticRulesSubMin schedule is enabled. This means that the application will evaluate all mailbox routing rules on an automatic basis.

To enable the MailboxEvaluateAllAutomaticRulesSubMin schedule:

1. From the **Deployment** menu, select **Schedules**.
2. In the Search section, type **Mailbox** and click **Go!**.
3. Locate the **MailboxEvaluateAllAutomaticRulesSubMin** schedule in this list and select the check box in the Enable column.
4. Click **Return**.

## Enabling CHIPS Document Tracking

When you are creating or editing your CHIPS business process in the business process text editor, you can enable CHIPS document tracking in the application by selecting the **Document Tracking** check box on the Process Levels page. Set the following options as needed and leave the rest of the business process parameters as the defaults:

- On the **Deadline Settings** page, set the deadline and notification options, if necessary.
- On the **Life Span** page, set the life span, if necessary.



## Chapter 7. CHIPS Utility Service

The CHIPS Utility service is responsible for the start of day, end of day, and CHIPS adapter lookup functions.

The following table provides an overview of the CHIPS Utility service:

System Name	CHIPSUtilityservice
Graphical Process Modeler (GPM) categories	All Services.
Description	This service is responsible for the start of day, end of day, and CHIPS adapter lookup functions.
Business usage	<p>This service is used by the CHIPS adapter to execute:</p> <ul style="list-style-type: none"> <li>• Start of Day functions (reset the CHIPS status to 1, CHIPS message number to 1, and the MQ_counter to 0)</li> <li>• End of Day functions (move the outbound messages into the history table, move the inbound messages into the history table, update Mailbox_id in the MBX table with the Mailbox_id of the &lt;participant_num&gt;_history mailbox)</li> <li>• CHIPS adapter lookups (based on the channel information, get the CHIPS adapter name related to the appropriate set)</li> </ul>
Usage example	Used to prepare for daily CHIPS usage, perform end of day housekeeping functions to properly store the data in the database, and look up the CHIPS adapter name when a message is received from the WebsphereMQ Async Receiver adapter (to allow the application to send back an acknowledgement message using the correct CHIPS adapter and transport mode). The predefined CHIPSUtility_ReceiveHandler business process is provided to handle incoming messages from CHIPS.
Preconfigured?	Yes.
Requires third party files?	No.
Platform availability	All supported application platforms.
Related services	This service works with the CHIPS adapter.
Application requirements	None.

Initiates business processes?	No. However, a predefined business process (CHIPSUtilitySOD) is provided to execute the CHIPS Utility service start-of-day actions. You can schedule this predefined business process to be started every morning. Also, a predefined business process (CHIPSUtilityEOD) is provided to execute the CHIPS Utility service end-of-day actions. You can schedule this predefined business process to be started every evening.
Invocation	A user who has permission to perform this activity must execute the business process that invokes this service.
Business process context considerations	N/A
Returned status values	Success, Failure.
Restrictions	None.
Persistence level	N/A
Testing considerations	To test this service, run the CHIPS Utility service business process and verify that it completes successfully. Debug information for this service is located at: Operations > System > Logs > CHIPS

## How the CHIPS Utility Service Works

The CHIPS Utility service prepares for daily CHIPS usage, performs end of day housekeeping functions to properly store the data in the database, and perform lookups to the CHIPS adapter name when a message is received from the WebsphereMQ Async Receiver adapter (to allow the application to send back an acknowledgement message using the correct CHIPS adapter and transport mode).

## Implementing the CHIPS Utility Service

You do not need to do anything to implement the CHIPS Utility service.

## Configuring the CHIPS Utility Service

You do not need to do anything to configure the CHIPS Utility service.

## Parameters Passed From Business Process to Service

The following table contains the parameter passed from the business process to the CHIPS Utility service:

Parameter	Description
serviceName	This is the CHIPS Adapter Instance Name.

## Business Process Example

This example uses the following message types for CHIPS Utility service:

- startOfDay
- endOfDay

- handleCHIPSReceiveRequest

This example sets the start of day parameters:

```
<process name="CHIPSUtilitySOD">
  <sequence
name="CHIPSUtility_StartOfDay">
  <operation
name="set user token">
  <participant
name="SetUserToken"/>
  <output message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*"/>
  </output>
  <input
message="inmsg">
  <assign to="." from="*"/>
  </input>
  </operation>
  <!--
Start of Day Process-->
  <operation>
  <participant
name="CHIPSUtilityService"/>
  <output
message="startOfDay">
  <assign to="."
from="*"/>
  <assign to="serviceName">CHIPSAdapter</assign>
  </output>
  <input
message="testing">
  <assign to="." from="*"/>
  </input>
  </operation>
  </sequence>
</process>
```

This example sets the end of day parameters:

```
<process
name="CHIPSUtilityEOD">
  <sequence name="CHIPSUtility_EndOfDay">
  <operation
name="set user token">
  <participant
name="SetUserToken"/>
  <output message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*"/>
  </output>
  <input
message="inmsg">
  <assign to="." from="*"/>
  </input>
  </operation>
  <!--
End of Day Process-->
  <operation>
  <participant
name="CHIPSUtilityService"/>
  <output
```

```

message="endOfDay">
  <assign to="."
from="*" />
  <assign to="serviceName">CHIPSAdapter</assign>
  </output>
  <input
message="testing">
  <assign to="." from="*" />
  </input>
</operation>
</sequence>
</process>

```

This example handles incoming CHIPS requests. The CHIPSUtility\_ReceiveHandler is preloaded into the application:

```

<process name="CHIPSUtility_ReceiveHandler">
  <sequence
name="CHIPSAdapter">
  <operation name="set
user token">
  <participant name="SetUserToken" />
  <output
message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*" />
  </output>
  <input
message="inmsg">
  <assign to="." from="*" />
  </input>
  </operation>
  <!--
handle incoming CHIPS request -->
  <operation>
  <participant
name="CHIPSUtilityService" />
  <output
message="handleCHIPSReceiveRequest">
  <assign
to="." from="*" />
  </output>
  <input
message="testing">
  <assign to="." from="*" />
  </input>
  </operation>
  </sequence>
</process>

```

---

## Chapter 8. CII Deenvelope Service

### CAUTION:

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the CII Deenvelope service:

System name	DeenvelopeCIIType
Graphical Process Modeler (GPM) categories	All Services, EDI > CII
Description	Parses the message group. Using values found in the message group header, the service attempts to find a matching envelope. If a matching envelope is found, the envelope defines how the message group should be further processed. If a matching envelope is not found, the service fails and the compliance report indicates that the envelope could not be found.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	No
Application requirements	No

### Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

#### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

#### Outbound

- EDI Encoder service
- CII Envelope service

- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM: it must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

**Note:** All EDI services assign a Unique ID to each log message.

### Adding Translation Map Name to Process Data

The CII Deenvelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The CII Deenvelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.

---

## Chapter 9. CII Envelope Service

### CAUTION:

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the CII Envelope service:

System name	EnvelopeCIIType
Graphical Process Modeler (GPM) categories	All Services, EDI > CII
Description	Relies on the EDI Encoder service to locate the document envelope to be used to envelope the messages. Takes as input one or more messages. Envelopes the messages with a message group header and trailer. Values defined in the CII document envelope associated with the messages are used to build the message group header. Similar to the X12 and EDIFACT envelope services.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	EDI Encoder service
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a predefined EDI business process.

### Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

#### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

## Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM: it must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

**Note:** All EDI services assign a Unique ID to each log message.

## Adding Translation Map Name to Process Data

The CII Envelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The CII Envelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.



---

## Chapter 10. Correlation Service

The following table provides an overview of the Correlation service:

System name	CorrelationServiceType
Graphical Process Modeler (GPM) categories	All Services, System
Description	Adds a record to the correlation table to enable you to track a document or business process. Collects the information for a specific name and value pair from either documents that pass through a business process, or from the business process itself. The correlation name and value pairs are saved in the correlation table.
Preconfigured?	No
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a business process.
Business process context considerations	No
Returned status values	None
Restrictions	No
Persistence level	None

### Using Correlations in the Application

The application uses correlations to define specific data items as tracking points for business processes and documents. Rather than having to search all of the application or a particular invoice, for example, you can define a correlation that enables you to search for that invoice number, which saves you time.

The data for the correlations is stored as name/value pair records in the Correlation table in the application. You can then search the data using the Correlation Search option.

Data for correlations can be collected by:

- Using the Correlation service in a business process.
- Using an Update standard rule in a map.
- Using one or more of the EDI services in a business process.

For information about these services, see the sections on the EDI services in this guide.

## How the Correlation Service Works

Each configuration of the Correlation service is set up to collect the information for a specific name and value pair from either documents that pass through a business process, or from the business process itself. If you choose to collect information from a document, the document's correlation value is written associated to the primary/non-primary document identifier. If the Correlation service is used to collect information about the business process, the correlation value for that process is written associated with the process ID. The correlation name and value pair are saved in the correlation table. You can then locate the document or business process using its associated name/value pair.

### Example

You want to be able to view the contents of incoming purchase orders. In this case, the inbound purchase orders go through a business process that includes the Translation service. The Correlation service configuration is added to the business process after the Translation service, so it starts after translation.

The Correlation service has the following parameters set: the Correlation name is PONumber, and the Type is Document. When purchase orders pass through the Correlation service during the business process, they pass the information for the PONumber to the service.

The name and value pair you specify can then be used in Correlation Search to locate the correlation object, which in this case is a document. For this example, the Correlation Search will display a link to the primary document created from the Translation service, since the Correlation service followed the start of the Translation service.

For example, purchase order number 12345 passes through the Correlation service, and passes this information to the service:

- Correlation Name: PONumber
- Value: 12345
- Type: Document

The service adds a correlation called PONumber with a value of 12345 for the primary document. Using the Correlation Search option, you can enter the name/value pair PONumber/12345, and view the contents of that purchase order.

You can dynamically assign NAME and VALUE from the process data in your business process to associate with a document or business process. The following figure shows an example of how the Correlation service could be used in a business process:

```
<process name="test">
  <sequence> <sequence>
    <operation
name="SaveId"><operation
name="SaveId">
      <participant
name="CorrelationService"/><participant
name="CorrelationService"/>
        <output
message="Xout">
          <assign
to="NAME">PO_Number</assign>
          <assign
```

```

to="VALUE">P012345</assign>
  <assign
to="TYPE">DOCUMENT</assign>T<assign
to="TYPE">DOCUMENT</assign>
  <assign
to="OBJECT_ID" from="/ProcessData/PrimaryDocument/@SCIOBJECTID/text()"/><assign
to="OBJECT_ID" from="/ProcessData/PrimaryDocument/@SCIOBJECTID/text()"/>
  <assign
to="." from="*"</assign><assign
to="." from="*"</assign>
  </output>
  <input
message="xin">
  <assign
to="." from="*"</assign><assign
to="." from="*"</assign>
  </input>

</operation>
</sequence>
</process>

```

## Implementing the Correlation Service

To implement the Correlation service, complete the following tasks:

1. Create a Correlation service configuration. For information, see *Creating a Service Configuration* on page 15.
2. Configure the Correlation service.
3. Use the Correlation service in a business process.

## Configuring the Correlation Service

To configure the Correlation service, you must specify settings for the following fields in the GPM:

Field	Description
Config	Name of the service configuration. Required.
NAME	Name associated with this correlation. For example, PONUMBER.
Object_ID	ID of the document or business process that correlates with a specific name/value pair. Generally, this field is left blank.
Type	The information this correlation will track. Valid values are Document and Business Process.
Value	Value of the correlation. For example, a purchase order number such as 12345.



---

## Chapter 11. Document Extraction Service

The Document Extraction service can be used to split individual documents out of a batch file to make each one a separate document. It can also be used to initiate EDI enveloping and outbound document processing. This service also enables you to batch multiple XML documents for processing.

The following table provides an overview of the Document Extraction service:

System name	DocumentExtractionService
Graphical Process Modeler (GPM) categories	All Services, Translation
Description	The Document Extraction service takes a file that contains one or more individual document, and attempts to find the map (from a specified list of maps) that produces output. The service extracts each individual document and puts it into the business process context with the name DOC-SPLIT- <i>N</i> where <i>N</i> is a 1-based number. Additionally, you have the option of having the extracted documents batched, EDI encoded, and EDI enveloped. The Document Extraction service can also extract XML sub-documents out of an XML compound document. Additionally, for the CHIPS and Fedwire standards, the Document Extraction service can be used to obtain values for the Application Sender ID and Application Receiver ID. These values can be passed to the EDI Encoder service to find an envelope for a document that needs to be extracted, encoded, and enveloped. Also, the Document Extraction service contains XML document extraction parameters that are supported when extracting XML documents. The values for these service parameters are XPath statements used by this service to locate the Application Sender ID and Application Receiver ID values within the XML document.
Business usage	The Document Extraction service allows you to concatenate many files into one, making tracking of the data more manageable.
Usage example	Your application generates purchase orders in a batch file, once a day. The application is scheduled to retrieve this file each day. When it is received, the first step in the business process has the Document Extraction service parse the file and produce an order document for each trading partner. Each document is passed on to the translator for conversion, and sent to the appropriate trading partner.
Preconfigured?	An instance of this service is created upon installation but is not configured, nor is any configuration required other than specifying values for the parameters when used within a business process.
Requires third party files?	No
Platform availability	All supported application platforms
Related services	EDI Encoder, EDI Envelope, For Each
Application requirements	None
Initiates business processes?	No
Invocation	By a business process

Business process context considerations	No
Returned status values	<p>Values:</p> <ul style="list-style-type: none"> <li>• Success – Extraction (and optional encoding and enveloping) were successful.</li> <li>• Error – Errors were encountered during extraction, encoding, enveloping. Consult the status report in the business process context.</li> </ul>
Restrictions	No

## Requirements

To use the Document Extraction service, you should have advanced knowledge about translation maps and extended rules.

The Document Extraction service can only be used to extract documents from a batch file if the documents in that file are all in the same format. For example, they must all be in application (positional) format, or they must all be in EDI format.

### CAUTION:

**An attempt to use this service to handle multiple data formats will produce unpredictable results and a potential loss of data.**

## How the Document Extraction Service Works

The Document Extraction service uses one or more translation maps to perform extraction using:

- Extended rules to find the start and end of a single document
- The Update standard rule to set sender ID, application sender ID, receiver ID, application receiver ID, and AcceptorLookupAlias values from the document

The Document Extraction service provides the option to batch together similar documents during this extraction. If this option is specified, all documents extracted that have the same sender ID, receiver ID, and AcceptorLookupAlias will be batched into a single document.

Additionally, the Document Extraction service contains XML document extraction parameters that are supported when extracting XML documents. The values for these service parameters (XMLAppSenderIDPath and XMLAppReceiverIDPath) are XPath statements used by the Document Extraction service to locate the Application Sender ID and Application Receiver ID values within the XML document.

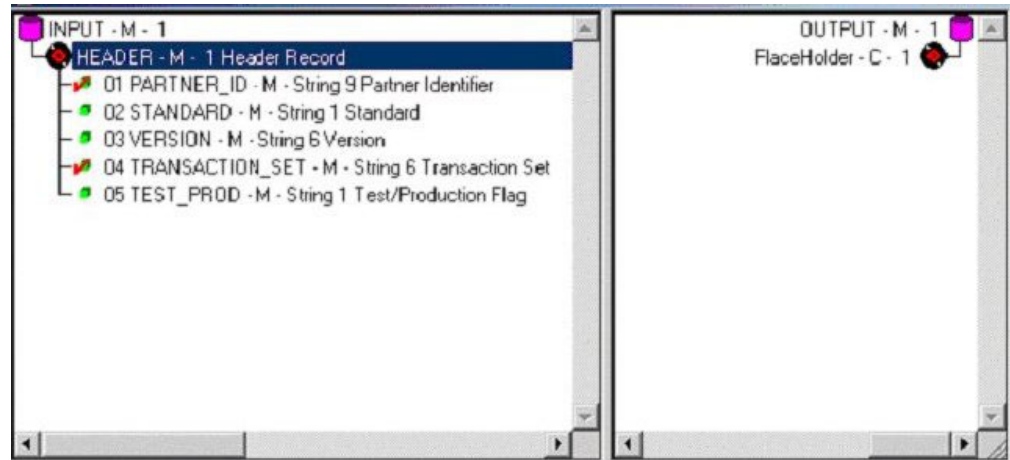
### CAUTION:

**You must write all XPath expressions (for example, Sender ID, Receiver ID, ALA, and so forth) in accordance with the extracted subdocument. See *Example of XML Code* for an example of a correct XPath expression.**

## How the Document Extraction Service Uses Translation Maps

Translation maps define how a single document looks and where to find the sender ID, receiver ID, and acceptor lookup alias values. Defining how a document looks is really the same as defining where a document starts and ends.

The following sample map is used to find a document that starts with a HDR record and ends with a SUM record:



This particular map defines only the first record of the file it is attempting to match, and it does not link any field from the input side of the map to the output side. The data is extracted using an extended rule on the first field of this header record. The function of the extended rule is to read and write records until it finds the end of the document. For this example, all records up to and including the SUM record are read and written.

The following example shows the extended rule that is defined on the PARTNER\_ID field:

```
//***** HEADER -> PARTNER_ID *****//

string[250] buffer;
string[3] match;
integer match_len;

// set these next two variables as desired
match = "SUM"; // the tag of the last record in the document
match_len = 3; // the length of the tag

// read the block we're on and write it
readblock(buffer);
writeblock(buffer);

// keep reading and writing records until the end of the document
while readblock(buffer) do
begin
```

```

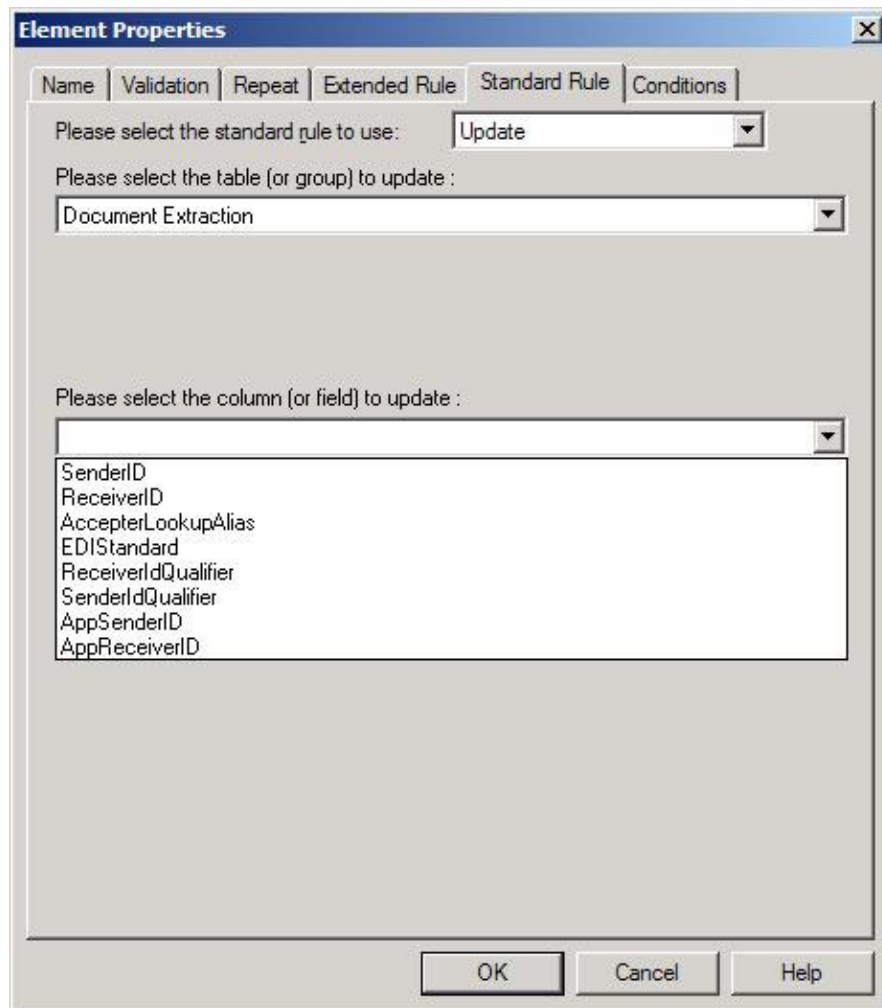
writeblock(buffer);

if left(buffer, match_len) = match then
    begin
        break;
    end
end

```

After specifying the start and end of the document, you can specify the sender ID, application sender ID, receiver ID, application receiver ID, and acceptor lookup alias. The Document Extraction service relies on the translator to set these values using the Update standard rule. In the previous example, the receiver ID and acceptor lookup alias are defined in the HDR record, and the sender ID is not used.

The following figure shows the standard rule that should be set up on the PARTNER\_ID field:



It sets the receiver ID value to the value in that field in the data. The Document Extraction service accesses this value and puts it into process data. This will be explained in further detail later.



Similarly, the field that is to be used as the accepter lookup alias should have an Update standard rule defined with the Document Extraction table and AcceptorLookupAlias column. In this example, the AcceptorLookupAlias value is populated by the data in the TRANSACTION\_SET field. Similar logic also applies for the sender ID.

This example is fairly straightforward, where the metadata to be extracted (receiver ID, and accepter lookup alias) is in the first record. If this information is not in the first record of a document, you must include more than one record in the map. In this case, the extended rule on a field in the first record reads and writes records until it locates the record that contains the metadata. Then this record should contain a field with an extended rule that reads and writes blocks until the end of the document is reached.

## What Happens When the Service Runs

When a document is processed by the Document Extraction service, zero or more extracted documents are produced. The documents are named with the convention DOC-SPLIT-1, DOC-SPLIT-2, ... DOC-SPLIT-*n*. Values for sender ID, application sender ID, receiver ID, application receiver ID, and accepter lookup alias are placed into process data as child elements of the document.

In the following example of process data after the Document Extraction service runs, two documents are extracted from the primary document:

```
<?xml version="1.0" encoding="UTF-8"?>
<ProcessData>
  <PrimaryDocument SCIObjectID="server1:328145:f197c7bb55:-7250"/>
  <DOC-SPLIT-1 SCIObjectID="server1:328145:f197c7bb55:-7247">
    <ReceiverID>PETTEST1</ReceiverID>
    <AcceptorLookupAlias>810</AcceptorLookupAlias>
  </DOC-SPLIT-1>
  <DOC-SPLIT-2 SCIObjectID="server1:328145:f197c7bb55:-7246">
    <ReceiverID>PETTEST5</ReceiverID>
    <AcceptorLookupAlias>850</AcceptorLookupAlias>
  </DOC-SPLIT-2>
</ProcessData>
```

Any data found in the primary document that cannot be matched against any of the specified maps is placed in a document called *unrecognized*. The service status report describes what was processed by the Document Extraction service. The report includes the number of documents extracted and the number of each type if batch mode is set to Yes.

**Note:** When the Document Extraction parameter PDToProcessData is set to No, the DOC-SPLIT information will be placed in an array named SplitDocs. Use the For Each Document service to process the SplitDocs array. Each iteration through the For Each Document service will update process data with the current DOC\_SPLIT and remove the previous split.

## Implementing the Document Extraction Service

To implement the Document Extraction service, complete the following tasks:

1. Create the translation maps necessary to define how a single document looks and where to send the sender ID, application sender ID, receiver ID, application receiver ID, and acceptor lookup alias values.
2. Create a Document Extraction service configuration. See *Creating a Service Configuration*.
3. Configure the service. See *Configuring the Document Extraction Service*.
4. Create a business process that includes the Document Extraction service and enable it.
5. Test and run the business process and the adapter.
6. To batch XML documents, set the XMLRootTagForBatches property to a non-null value. See *Batching Multiple XML Documents* for more information.

**Note:** If the XMLRootTagForBatches property is null, the Document Extraction service generates malformed XML (a document without a root tag) if the batch contains more than one sub-document.

## Configuring the Document Extraction Service

To configure the Document Extraction service, you must specify field settings in the Graphical Process Modeler (GPM). For general information about service configurations, see *Creating a Service Configuration*.

The following table describes the fields used to configure the Document Extraction service in the GPM:

Field	Description
Config	Name of the service configuration.
BatchLikeDocuments	Whether to combine documents extracted by the service into a single document, or split each document out individually. Valid values are Yes and No. <ul style="list-style-type: none"><li>• If Yes is specified, documents that are extracted by the service are combined into a single document if they have the same sender ID, receiver ID, and acceptor lookup alias. This parameter also allows you to batch XML documents.</li><li>• If No is specified, each document is split out individually, regardless of sender ID, receiver ID, and acceptor lookup alias values.</li></ul> <b>Note:</b> While batch processing is faster and more efficient, the output contains only one batch document with correlations that are not associated to individual transactions. If you need to view individual transactions and their specific correlations, do not select batch processing.
DocExtractMapList	List of map names defined in the Map Editor that should be used to extract documents from a single batch file. Valid values are the names of the maps created. Make sure that all maps in this list are active in application. Map names in the list are separated by a space. <b>Note:</b> This parameter is mutually exclusive with the XMLInput parameter. If XMLInput is set to Yes, this parameter will be ignored.
DOCUMENT_NAME_PREFIX	Allows the user to specify a name prefix other than DOC-SPLIT-. When specified, the document extraction service will label the document with the DOCUMENT_NAME_PREFIX specified by the user.

Field	Description
EDIEncodeDocument	Whether each document extracted should be immediately EDI-encoded. Valid values are Yes and No. Must be set to Yes if EDIEnvelopeDocument is set to Yes. Otherwise, the service will generate an error and halt the business process. <b>Note:</b> If this parameter and the XMLInput parameter are set to Yes, the XMLSenderIDPath, XMLReceiverIDPath, and XMLAcceptorLookupAliasPath parameters must be set.
EDIEnvelopeDocument	Whether each document extracted and encoded should be immediately EDI-enveloped. Valid values are Yes and No. If this parameter is set to Yes, the EDIEncodeDocument parameter must also be set to Yes. Otherwise, the service will generate an error and halt the business process.
ErrorBP	Business process to execute when unrecognized data is received.
ErrorOnUnrecognizedData	Whether to generate an error when unrecognized data is received. Valid values are Yes and No.
HALT_ON_TRANS_ERROR	Whether to stop processing transactions when a translation error occurs. Valid values are: <ul style="list-style-type: none"> <li>• Yes – Stop processing transactions when a translation error occurs. This is the default in Immediate Enveloping mode.</li> <li>• No – Continue processing transactions if a translation error occurs. This is the default in Deferred Enveloping mode.</li> </ul>
PDToProcessData	Specifies whether split documents will be stored in process data or in the business process context. Valid values are: <ul style="list-style-type: none"> <li>• Yes – Include each split document in process data (default).</li> <li>• No – Do not include each split document on process data; instead, store the information for each document in an array named SplitDocs. Retrieve documents from SplitDocs one at a time using the For Each Document service.</li> </ul> <p>The PDToProcessData parameter can be used to improve performance by reducing the overhead associated with persisting large process data information. The major performance improvement is realized at the current step (Document Extraction) and each subsequent step by persisting a much smaller process data which does not contain multiple split documents. By using the For Each Document service in conjunction with PDToProcessData = No, only the current document is on process data. this avoids repetitive writing of non-current documents. Once documents are in the SplitDocs array, they can be retrieved only by using the For Each Document service. Setting this parameter to No, combined with setting EDIEncodeDocument and EdiEnvelopeDocument to Yes, may yield an improvement by reducing the amount of persisted process data information. The benefit will depend on the complexity of the business process. In this situation, you would not use the For Each Document service, because the documents are passed to invoked business processes.</p>

Field	Description
UseInputEdiDelimiters	<p>Whether the delimiters read from the syntax record for an EDI map are used when generating the extracted documents. Valid values:</p> <ul style="list-style-type: none"> <li>• Yes – Extract the document using the same delimiters that were used to read the input.</li> <li>• No – Do not use the delimiters.</li> </ul> <p><b>Note:</b> When using the Document Extraction service to split documents in a delimited EDI file, only one set of delimiters is supported. The input file cannot contain multiple documents that each use a different set of delimiters. Therefore, if you have multiple documents using different delimiters, you must first send the data through the EDI Deenvelope service (in Document mode) to split out the individual documents that use differing delimiters. When you invoke the EDI Deenvelope service, the Mode service parameter must be set to Document. This instructs the EDI Deenvelope service not to bootstrap a business process after it finishes splitting the input file. You must also update the customer_overrides.properties file to include the START and END tag of the documents to be extracted, as well as the delimiters (or the location of delimiters) to use.</p>
XMLAcceptorLookupAliasPath	<p>XPath-like string indicating the location of the XMLAcceptorLookupAliasPath element tag. Required if XMLInput and EDIEncoding are set to Yes. Valid value: SubDocument/AcceptorLookupAlias</p>
XMLEDIEnvelopeStandard	<p>Standard to be used by this instance of the Document Extraction service. For example: X12, EDIFACT, TRADACOMS, or CII. Optional.</p>
XMLInput	<p>Specifies whether XML input will be received. Required if extracting XML documents. Valid values:</p> <ul style="list-style-type: none"> <li>• Yes – XML documents can be extracted.</li> <li>• No – XML documents will not be used as input.</li> </ul> <p><b>Note:</b> If this parameter is set to Yes, at least the XMLRootTag parameter must be set. This parameter is mutually exclusive with the DocExtractMapList parameter. If this parameter is set to Yes, the DocExtractMapList parameter will be ignored.</p>
XMLReceiverIDPath	<p>XPath-like string indicating the location of the XMLReceiverIDPath element tag. Required if XMLInput and EDIEncoding are set to Yes. Valid value: SubDocument/Receiver</p>
XMLReceiverIDQualifierPath	<p>XPath-like string indicating the location of the XMLReceiverIDQualifierPath element tag. Optional. Valid value: SubDocument/ReceiverQual</p>
XMLRootTagForBatches	<p>Root tag for the XML document. Required if the BatchLikeDocuments and XMLInput parameters are set to Yes. Valid value is the DocumentRootTag. If this parameter is null, the batched XML documents are generated. However, the end document will be malformed if it contains more than one sub-document. If you define the XMLRootTagForBatches parameter, the Document Extraction service generates a valid XML document with this defined value as the root tag of the document.</p>
XMLRootTag	<p>Root tag for the subdocument. Required if XMLInput is set to Yes. Valid value: SubDocument</p>
XMLSenderIDPath	<p>XPath-like string indicating the location of the XMLSenderIDPath element tag. Required if XMLInput and EDIEncoding are set to Yes. Valid value: SubDocument/Sender</p>
XMLAppSenderIDPath	<p>XPath-like string indicating the location of the XMLAppSenderIDPath element tag. Optional. Valid value: SubDocument/AppSenderID</p>
XMLSenderIDQualifierPath	<p>XPath-like string indicating the location of the XMLSenderIDQualifierPath element tag. Optional. Valid value: SubDocument/SenderQual</p>

## Example of XML Code

An example of XML code that could be used with the Document Extraction service when extracting XML sub-documents from an XML compound document is shown below:

**Note:** You must write all XPath expressions (for example, Sender ID, Receiver ID, ALA, and so forth) in accordance with the extracted subdocument. For the code example provided below, the following XPath for Sender ID is:  
/SubDocument/Sender.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
```

```
△<CompoundDocument>
```

```
  △△<SubDocument>
```

```
    △△△<!-- subdocument # 1 -->
```

```
      △△△<Sender>DOCEXTRACTTEST</Sender>
```

```
      △△△<SenderQual>ZZ</SenderQual>
```

```
      △△△<Receiver>PETTEST1</Receiver>
```

```
      △△△<ReceiverQual>AA</ReceiverQual>
```

```
      △△△<AcceptorLookupAlias>810</AcceptorLookupAlias>
```

```
      △△△<Manifest>
```

```
      △△△...
```

```
      △△△</Manifest>
```

```
    △△</SubDocument>
```

```
  △△<SubDocument>
```

```
    △△△<!-- subdocument # 2 -->
```

```
    △△△...
```

```
  △△</SubDocument>
```

```
△...
```

```
</CompoundDocument>
```

## Batching Multiple XML Documents

The application enables you to specify a root tag if you are batching multiple XML documents, which allows you to generate valid XML output. There are two ways to perform XML batching, depending on whether or not the Document Extraction service instance is configured to do EDI Enveloping as part of the extraction. If the service instance is configured to do EDI enveloping, you do not have to use a root tag even if you are batching XML documents, because the service puts all of the documents that would go into a batch in the same call to the EDI Envelope service (which produces the same effect as batching without putting all of the documents in the same file with a common root tag). If the service instance is not configured to do EDI enveloping, the only way to batch and create valid XML output is to provide a root tag to use. Therefore, if you are doing EDI enveloping, the root tag

can be NULL. If you are not doing EDI enveloping, the root tag can also be NULL, but will not generate a valid XML output document.

Please see the following table for a further explanation of when you should specify a root tag:

<b>Value Specified for the BatchLikeDocuments Parameter (flag)</b>	<b>Value Specified for the EDIEnvelopeDocument Parameter (flag)</b>	<b>Value Specified for the XMLRootTagForBatches Parameter (string)</b>	<b>Document Extract Service Result</b>
Yes	No	Present (non-null)	Valid XML output
Yes	No	NULL	Invalid XML output
Yes	Yes	Present (non-null)	Documents are EDI processed as usual
Yes	Yes	NULL	Documents are EDI processed as usual

---

## Chapter 12. EDI Deenveloping Service

The following table provides an overview of the EDI Deenveloping service:

System name	EDIDevelopeType
Graphical Process Modeler (GPM) categories	All Services, EDI
Description	Identifies the EDI interchanges (including VDA, SWIFTNet, and RND) contained within a message, extracts them to separate messages, and starts the appropriate business process to handle each one.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms

### Implementing the EDI Deenveloping Service

To implement the EDI Deenveloping service, complete the following tasks:

1. Create an EDI Deenveloping service configuration. See *Creating a Service Configuration*.
2. Configure the EDI Deenveloping service. For information, see *Configuring the EDI Deenveloping Service*.
3. Use the EDI Deenveloping service in a business process.

### Configuring the EDI Deenveloping Service

To configure the EDI Deenveloping service, you must define following fields in the GPM:

Field	Description
Config	Name of the service configuration.
InterchangeTypes	Enables you to specify what interchange type is used when the service runs. Used only for ACH, CII, VDA, SWIFT, and RND. Optional. Valid values are ACH, CII, VDA, SWIFT, and RND.

### Correlation Data

The EDI Enveloping service and EDI Deenveloping service automatically collect the following correlation information from EDI documents:

Category	Data Collected
Sender ID	<ul style="list-style-type: none"><li>• InterchangeSenderID</li><li>• GroupSenderID</li><li>• TransactionSenderID</li></ul>

Category	Data Collected
Receiver ID	<ul style="list-style-type: none"> <li>InterchangeReceiverID</li> <li>GroupReceiverID</li> <li>TransactionReceiverID</li> </ul>
Control Numbers	<ul style="list-style-type: none"> <li>InterchangeControlNumber</li> <li>GroupControlNumber</li> <li>TransactionControlNumber</li> </ul>
Acknowledgement Requested	<ul style="list-style-type: none"> <li>InterchangeAckRequested</li> <li>GroupAckRequested</li> </ul>
Acknowledgement Status	<ul style="list-style-type: none"> <li>InterchangeAckStatus</li> <li>GroupAckStatus</li> </ul>
Standard	Standard – values are CII, EDIFACT, VDA, RND, SWIFT, and X12
ID	<ul style="list-style-type: none"> <li>FunctionalID</li> <li>TransactionSetID</li> </ul>
Versions	<ul style="list-style-type: none"> <li>InterchangeVersion</li> <li>GroupVersion</li> <li>TransactionVersion</li> </ul>
Date and Time	<ul style="list-style-type: none"> <li>InterchangeDateTime</li> <li>GroupDateTime</li> </ul>
Overdue Time	<ul style="list-style-type: none"> <li>InterchangeOverdueTime</li> <li>GroupOverdueTime</li> </ul>
Level	Level – values are Interchange, Group, and Transaction
Direction	Direction – values are Inbound and Outbound
Envelope Type	<ul style="list-style-type: none"> <li>InterchangeEnvelopeType</li> <li>GroupEnvelopeType</li> <li>TransactionEnvelopeType</li> </ul>
Envelope Name	<ul style="list-style-type: none"> <li>InterchangeEnvelopeName</li> <li>GroupEnvelopeName</li> <li>TransactionEnvelopeName</li> </ul>
Envelope Version	<ul style="list-style-type: none"> <li>InterchangeEnvelopeVersion</li> <li>GroupEnvelopeVersion</li> <li>TransactionEnvelopeVersion</li> </ul>
Compliance Status	<ul style="list-style-type: none"> <li>InterchangeComplianceStatus</li> <li>GroupComplianceStatus</li> <li>TransactionComplianceStatus</li> </ul> Values are OK and NOT OK
Test Mode	TestMode – values are Production, Test Data and Information
Counts	<ul style="list-style-type: none"> <li>TransactionCount</li> <li>GroupCount</li> </ul>



Category	Data Collected
Container Doc ID	ContainerDocID

The information for these correlations is automatically collected at each envelope and develope stage for a document, which facilitates the tracking of individual documents as they move through the application.

There is no setup required for using the correlation information collected by the EDI Enveloping and EDI Deenveloping services. After you use one of these services in a business process, the information is available through the Correlation Search option.

## Using Wildcards in Enveloping

As a way to help reduce the number of envelopes you need to create and use, the EDI Envelope and EDI Deenveloping services support use of an asterisk (\*) as a wildcard character in mandatory envelope fields for X12 and EDIFACT only. By using wildcards, you can set up one set of envelopes that can be used for multiple trading partners. If certain trading partners have specific requirements, you can still have envelopes that pertain just to them, and the EDI Envelope service chooses the envelope that is the best match. In other words, the envelope that has the most matches to specific fields in the data (for example Receiver ID, Receiver ID Qualifier), is the one selected.

## Configuring EDI Deenveloping to Indicate an Error When Processing Data With No Valid Interchanges

To configure EDI Develope to indicate an error when you process data that does not contain valid interchanges, add the following lines to the EDI Develope business process:

```
<assign to="ErrorOnUnrecognizedData">yes</assign>
<assign
to="ProcessInterchangesDespiteError">yes</assign>
```

These parameters function as follows:

Parameter	Function
ErrorOnUnrecognizedData	If this option is set to <b>Yes</b> , the business processes fails if it encounters an interchange that is not a valid interchange (based on the interchange types in the enveloping.properties file).
ProcessInterchangesDespite Error	If this option is set to <b>Yes</b> , any valid interchanges continue to bootstrap the develope process. If this option is set to <b>No</b> and one invalid interchange is found, none of the interchanges will bootstrap the develope process.

## Configuring EDI Deenveloping to Check for Missing End Tags

To configure EDI Develope to indicate an error when end tags are missing, add the following lines to the EDI Develope business process:

```
<assign to="BASIC_CHECK_FOR_MISSING_END_TAG">yes</assign>
<assign
to="COMPREHENSIVE_CHECK_FOR_MISSING_END_TAG">yes</assign>
```

These parameters function as follows:

Parameter	Function
BASIC_CHECK_FOR_MISSING_END_TAG	If this option is set to <b>Yes</b> , the business process checks to see if any end tags are missing.
COMPREHENSIVE_CHECK_FOR_MISSING_END_TAG	If this option is set to <b>Yes</b> , the business process kicks off a comprehensive check to see if any end tags are missing.

## Configuring the Number of Interchange Types Allowable with EDI Deenveloping

To configure EDI Deenvelope to override the default number of interchange types set in the enveloping.properties files, add the following line to the EDI Deenvelope business process:

```
<assign to="interchangetypes">15</assign>
```

This parameter functions as follows:

Parameter	Function
InterchangeTypes	Overrides the default set in the enveloping.properties file to specify the number of interchange types for deenveloping.

## Configuring SWIFT FIN Extraction for ACK with Original Message

Depending on the configuration of SWIFT Alliance Access, FIN acknowledgments that are received by the application may be accompanied by a full or partial copy of the corresponding original message, similar to the following:

```
{1:F21PTSCFRN0AXX0208003695}{4:{177:1001131958}{451:0}
{108:24}}{1:F01PTSCFRN0AXX0208003695}{2:I999PTSCFRN0XXXXN}
{3:{108:24}}{5:{CHK:6F227EC3C468}{TNG:}{PDE:}}{S:{CON:}
{UNT:None}{USR:all_adm}}
```

By default, the EDI Deenveloping service breaks up this type of message into two separate documents for processing. To direct the service to keep the acknowledgment and the corresponding original message together in a single document for processing, you must set the workflow parameter **SWIFT\_FIN\_EXTRACT\_MESSAGE\_WITH\_ACK** to **Yes** or **True** when you invoke the service. For example:

```
<assign to="SWIFT_FIN_EXTRACT_MESSAGE_WITH_ACK">true</assign>
```

This parameter functions as follows:

Parameter	Function
SWIFT_FIN_EXTRACT_MESSAGE_WITH_ACK	Set to True or Yes to process a SWIFT FIN acknowledgment and the corresponding original message as one document. The default for this parameter is False.

## Chapter 13. EDI Encoder Service (Build 7000-7001)

**Note:** If the input document character encoding is specified, it overrides the encoding specified in the map. The output document content type and character encoding are set based on the information contained in the map.

The following table provides an overview of the EDI Encoder service:

System name	EDIEncoderType
Graphical Process Modeler (GPM) categories	<ul style="list-style-type: none"> <li>• All Services</li> <li>• EDI &gt; X12</li> <li>• EDI &gt; EDIFACT</li> <li>• EDI &gt; CII</li> <li>• EDI &gt; SWIFT</li> </ul>
Description	<p>Determines which transaction-level envelope will be used on the document. If translations are specified in an envelope, the service determines which map to use. Additionally, for the CHIPS standard, allows you to specify the Sender ID and/or Application Sender ID along with the Acceptor Lookup Alias, to allow for an envelope lookup for a document that is to be enveloped. For the Fedwire standard, allows you to specify the Sender ID and/or Application Sender ID, Receiver ID and/or Application Receiver ID, along with the Acceptor Lookup Alias, which allows for an envelope lookup for a document that is to be enveloped (outbound).</p> <p><b>Note:</b> In previous releases, the document lifespan default was zero so that when the workflow expired, all associated documents were purged/archived with the workflow. Now the lifespan is configurable for documents (the default is 30 days) and standards that use the EDI Encoder service.</p>
Preconfigured?	A configuration of this service is installed with the product, but is not configured. The only configuration required for the service is to specify parameter values to be used within a business process, but you must also define the envelopes within the application.
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	EDI Envelope Service
Application requirements	No
Initiates business processes?	None
Invocation	Runs by a predefined business process.

Returned status values	<ul style="list-style-type: none"> <li>• Success – The envelope for the document was found and if the envelope specified a map, this information is passed to the EDI Envelope service.</li> <li>• Error – The envelope for the document could not be located.</li> </ul>
Restrictions	This service is used in outbound business processes only.

## Implementing the EDI Encoder Service

To implement the EDI Encoder service, complete the following tasks:

1. Create an EDI Encoder service configuration. For information, see *Managing Services and Adapters*.
2. Configure the EDI Encoder service. For information, see *Configuring the EDI Encoder*.
3. Use the EDI Encoder service in a business process.

## Configuring the EDI Encoder

To configure the EDI Encoder service, you must specify settings for the following fields in the GPM:

Field	Description
Config	Name of the service configuration.
AcceptorLookupAlias	Identifying string used with the sender ID (and/or Application Sender ID for CHIPS) and receiver ID (and/or Application Receiver ID for CHIPS) to look up this envelope. This alias associates a document with the service it requires. This same field is specified in the transaction-level outbound envelope. Also, this identifying string is used with the Sender ID and Receiver ID to look up this envelope for the Fedwire Outbound Envelope. Required. <b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Acceptor Lookup Alias</b> parameter.
EDI Standard	Enter the EDI standard to be used (including CHIPS or Fedwire).

Field	Description
Mode	<p>Determines whether documents are enveloped immediately or deferred to be enveloped at a later time. Optional. The mode used here must correspond to the mode in which the EDI Envelope service will be called. Valid values are:</p> <ul style="list-style-type: none"> <li>• Immediate – The envelopes are determined and associated with the document to be used during enveloping. Assumes the EDI Envelope service will be used to envelope the document later in the process.</li> <li>• Deferred – The service marks the document to be enveloped at a later time, usually according to a schedule. The document is stored in the database until enveloping takes place. (Default)</li> </ul> <p><b>Note:</b> If the mode is not specified, the default will write an entry to the CORRELATION_SET table that the document needs to be enveloped. This makes it eligible for deferred enveloping.</p>
ReceiverID	<p>Receiver identification, 2 characters minimum, 15 maximum. Required. Must match the receiver ID in the transaction-level document envelope.</p> <p><b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Receive Participant Number</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Receiver ID</b> parameter.</p>
SenderID	<p>Sender identification, 2 characters minimum, 15 maximum. Required. Must match the sender ID in the transaction-level document envelope.</p> <p><b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Send Participant Number</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Sender ID</b> parameter.</p>
ReceiverIDQual	<p>Receiver ID qualifier. Optional. Must match the receiver ID qualifier in the transaction-level document envelope.</p>

Field	Description
SenderIDQual	Sender ID qualifier. Optional. Must match the sender ID qualifier in the transaction-level document envelope.
AppSenderID	Coded identifier of the application data sender. Optional. <b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Application Sender ID</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Application Sender ID</b> parameter.
AppReceiverID	Coded identifier of the customer number or data source number. Optional. <b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Application Receiver ID</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Application Receiver ID</b> parameter.

## Using Wildcards in Enveloping

As a way to help reduce the number of envelopes you need to create and use, the EDI Envelope and EDI Develope services support use of an asterisk (\*) as a wildcard character in mandatory envelope fields for X12, EDIFACT, CHIPS and Fedwire only. The exception to this rule is when the field is Sender ID, Receiver ID, or a qualifier for one of those fields. For example, in EDIFACT the following fields are conditional, but are considered to be part of the Sender / Receiver ID and therefore must have a "\*" placed in the field if you want to override those values:

- (0008) Interchange Sender Internal Identification:
- (0042) Interchange Sender Internal Sub-identification:
- (0014) Interchange Recipient Internal Identification
- (0046) Interchange Recipient Internal Sub-identification

By using wildcards, you can set up one set of envelopes that can be used for multiple trading partners. If certain trading partners have specific requirements, you can still have envelopes that pertain just to them, and the EDI Envelope service chooses the envelope that is the best match. In other words, the envelope that has the most matches to specific fields in the data (for example, Receiver ID, Receiver ID Qualifier), is the one selected.

## Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM: it must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the best EDI performance with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides a good EDI performance while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.



## Chapter 14. EDI Encoder Service (Build 7002 or higher)

**Note:** If the input document character encoding is specified, it overrides the encoding specified in the map. The output document content type and character encoding are set based on the information contained in the map.

The following table provides an overview of the EDI Encoder service:

System name	EDIEncoderType
Graphical Process Modeler (GPM) categories	<ul style="list-style-type: none"> <li>• All Services</li> <li>• EDI &gt; X12</li> <li>• EDI &gt; EDIFACT</li> <li>• EDI &gt; CII</li> <li>• EDI &gt; SWIFT</li> </ul>
Description	<p>Determines which transaction-level envelope will be used on the document. If translations are specified in an envelope, the service determines which map to use. Additionally, for the CHIPS standard, allows you to specify the Sender ID and/or Application Sender ID along with the Acceptor Lookup Alias, to allow for an envelope lookup for a document that is to be enveloped. For the Fedwire standard, allows you to specify the Sender ID and/or Application Sender ID, Receiver ID and/or Application Receiver ID, along with the Acceptor Lookup Alias, which allows for an envelope lookup for a document that is to be enveloped (outbound).</p> <p><b>Note:</b> In previous releases, the document lifespan default was zero so that when the workflow expired, all associated documents were purged/archived with the workflow. Now the lifespan is configurable for documents (the default is 30 days) and standards that use the EDI Encoder service.</p>
Preconfigured?	A configuration of this service is installed with the product, but is not configured. The only configuration required for the service is to specify parameter values to be used within a business process, but you must also define the envelopes within the application.
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	EDI Envelope Service
Application requirements	No
Initiates business processes?	None
Invocation	Runs by a predefined business process.

Returned status values	<ul style="list-style-type: none"> <li>• Success – The envelope for the document was found and if the envelope specified a map, this information is passed to the EDI Envelope service.</li> <li>• Error – The envelope for the document could not be located.</li> </ul>
Restrictions	This service is used in outbound business processes only.

## Implementing the EDI Encoder Service

To implement the EDI Encoder service, complete the following tasks:

1. Create an EDI Encoder service configuration. For information, see *Managing Services and Adapters*.
2. Configure the EDI Encoder service. For information, see *EDI Encoder Service*.
3. Use the EDI Encoder service in a business process.

## Configuring the EDI Encoder

To configure the EDI Encoder service, you must specify settings for the following fields in the GPM:

Field	Description
Config	Name of the service configuration.
AccepterLookupAlias	Identifying string used with the sender ID (and/or Application Sender ID for CHIPS) and receiver ID (and/or Application Receiver ID for CHIPS) to look up this envelope. This alias associates a document with the service it requires. This same field is specified in the transaction-level outbound envelope. Also, this identifying string is used with the Sender ID and Receiver ID to look up this envelope for the Fedwire Outbound Envelope. Required. <b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Accepter Lookup Alias</b> parameter.
EDI Standard	Enter the EDI standard to be used (including CHIPS or Fedwire).

Field	Description
Mode	<p>Determines whether documents are enveloped immediately or deferred to be enveloped at a later time. Optional. The mode used here must correspond to the mode in which the EDI Envelope service will be called. Valid values are:</p> <ul style="list-style-type: none"> <li>• Immediate – The envelopes are determined and associated with the document to be used during enveloping. Assumes the EDI Envelope service will be used to envelope the document later in the process.</li> <li>• Deferred – The service marks the document to be enveloped at a later time, usually according to a schedule. The document is stored in the database until enveloping takes place. (Default)</li> </ul> <p><b>Note:</b> If the mode is not specified, the default will write an entry to the CORRELATION_SET table that the document needs to be enveloped. This makes it eligible for deferred enveloping.</p>
ReceiverID	<p>Receiver identification, 2 characters minimum, 15 maximum. Required. Must match the receiver ID in the transaction-level document envelope.</p> <p><b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Receive Participant Number</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Receiver ID</b> parameter.</p>
SenderID	<p>Sender identification, 2 characters minimum, 15 maximum. Required. Must match the sender ID in the transaction-level document envelope.</p> <p><b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Send Participant Number</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Sender ID</b> parameter.</p>
ReceiverIDQual	<p>Receiver ID qualifier. Optional. Must match the receiver ID qualifier in the transaction-level document envelope.</p>

Field	Description
SenderIDQual	Sender ID qualifier. Optional. Must match the sender ID qualifier in the transaction-level document envelope.
AppSenderID	Coded identifier of the application data sender. Optional. <b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Application Sender ID</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Application Sender ID</b> parameter.
AppReceiverID	Coded identifier of the customer number or data source number. Optional. <b>Note:</b> To specify this parameter in the CHIPS Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the CHIPS Outbound Envelope <b>Application Receiver ID</b> parameter. To specify this parameter in the Fedwire Outbound Envelope wizard to perform an envelope lookup for a document to be enveloped, type this value in the Fedwire Outbound Envelope <b>Application Receiver ID</b> parameter.

## Using Wildcards in Enveloping

As a way to help reduce the number of envelopes you need to create and use, the EDI Envelope and EDI Develope services support use of an asterisk (\*) as a wildcard character in mandatory envelope fields for X12, EDIFACT, CHIPS and Fedwire only. The exception to this rule is when the field is Sender ID, Receiver ID, or a qualifier for one of those fields. For example, in EDIFACT the following fields are conditional, but are considered to be part of the Sender / Receiver ID and therefore must have a "\*" placed in the field if you want to override those values:

- (0008) Interchange Sender Internal Identification:
- (0042) Interchange Sender Internal Sub-identification:
- (0014) Interchange Recipient Internal Identification
- (0046) Interchange Recipient Internal Sub-identification

By using wildcards, you can set up one set of envelopes that can be used for multiple trading partners. If certain trading partners have specific requirements, you can still have envelopes that pertain just to them, and the EDI Envelope service chooses the envelope that is the best match. In other words, the envelope that has the most matches to specific fields in the data (for example, Receiver ID, Receiver ID Qualifier), is the one selected.

## Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM: it must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the best EDI performance with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides a good EDI performance while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

---

## Chapter 15. EDI Enveloping Service

The EDI Enveloping service translates messages, determines which business processes need to run to apply EDI envelopes, and starts those business processes. The EDI Enveloping service has two methods of translating and enveloping messages:

- In Deferred mode, the EDI Enveloping service runs at a scheduled time and it translates and envelopes messages into interchanges in batches.
- In Immediate mode, the EDI Enveloping service translates and envelopes messages immediately after the EDI Encoder service processes the messages.

Before the application runs the EDI Enveloping service, you must have the EDI Encoder service process EDI messages that need to be translated and enveloped. The EDI Encoder service tags messages so that the EDI Enveloping service can properly translate and envelope them.

**Note:** If the input message character encoding is specified, it overrides the encoding specified in the map. The output message content type and character encoding are set based on the information contained in the map.

The following table provides an overview of the EDI Enveloping service:

System name	EDIEnvelopeType
Graphical Process Modeler (GPM) categories	<ul style="list-style-type: none"><li>• All Services</li><li>• EDI &gt; X12</li><li>• EDI &gt; EDIFACT</li><li>• EDI &gt; CII</li><li>• EDI &gt; VDA</li><li>• EDI &gt; SWIFT</li><li>• EDI &gt; RND</li></ul>

Description	<p>Responsible for translating and enveloping a message. It can be configured to run in either of two modes: Deferred or Immediate.</p> <ul style="list-style-type: none"> <li>In Deferred mode, the service queries the application to identify messages that require translation and enveloping. The query can incorporate an optional filter that allows the business process developer to restrict messages by any or all of Sender ID, Receiver ID, and Envelope ID. The service then divides the messages according to the envelopes they require and initiates the appropriate enveloping business processes. In deferred mode, the applications translates and envelops messages in batches at specified intervals.</li> </ul> <p><b>Note:</b> The EDI Enveloping service should be excluded from the business process when the EDI Encoding service is used; instead, the deferred EDI Enveloping service must be used in a different business process and should be triggered periodically.</p> <ul style="list-style-type: none"> <li>In Immediate mode, the service translates and envelops the current message as soon as the EDI Encoder service is finished with the message. It operates on the primary message it receives and initiates the appropriate enveloping business process.</li> </ul>
Preconfigured?	<p>A configuration of this service is created when the product is installed. However, before using a service configuration in a business process, you must specify values for the following parameters:</p> <ul style="list-style-type: none"> <li>MODE (required)</li> <li>SENDER_ID (optional)</li> <li>RECEIVER_ID (optional)</li> <li>ENVELOPE_ID (optional)</li> </ul>
Requires third party files?	No
Platform availability	All supported application platforms
Related services	EDI Encoder service
Application requirements	No
Initiates business processes?	None
Invocation	Runs as part of a predefined business process.
Returned status values	<ul style="list-style-type: none"> <li>Success – The messages were successfully submitted for enveloping.</li> <li>Error – Errors were encountered and the messages were not submitted for enveloping.</li> </ul>
Restrictions	No



## Implementing the EDI Enveloping Service

To implement the EDI Enveloping service, complete the following tasks:

1. Call the EDI Encoder service.
2. Create an EDI Enveloping service configuration. See *Creating a Service Configuration*.
3. Configure the EDI Enveloping service. For information, see *Configuring the EDI Enveloping Service*.
4. Use the EDI Enveloping service in a business process.

## Configuring the EDI Enveloping Service



To create a EDI Enveloping service configuration, you must specify field settings in the application and in the Graphical Process Modeler (GPM).

### Application Configuration<sup>△</sup>

The following table describes the fields used to configure the EDI Enveloping service in the application:

**Note:** The field names in parentheses represent the corresponding field names in the GPM. This information is provided for your reference.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	Select one of the options: <ul style="list-style-type: none"><li>• None – You do not want to include this configuration in a group at this time.</li><li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li><li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li></ul> <b>Note:</b> See <i>Using Service Groups</i> .

Field	Description
Enveloping Mode(MODE)	<p>Instructs the application when to translate and envelope messages:</p> <ul style="list-style-type: none"> <li>Deferred – Translate and envelope messages in batches at specified intervals.</li> </ul> <p><b>Note:</b> When using deferred mode, the EDI Enveloping service should be excluded from the business process when the EDI Encoding service is used; instead, the deferred EDI Enveloping service must be used in a different business process and should be triggered periodically.</p> <ul style="list-style-type: none"> <li>Immediate – Translate and envelope the current message as soon as the EDI Encoder service is finished with the message.</li> </ul>
Halt on Translation Error	<p>Whether to stop processing transactions when a translation error occurs. Valid values are:</p> <ul style="list-style-type: none"> <li>Yes – Stop processing transactions when a translation error occurs. This is the default in Immediate mode.</li> <li>No – Continue processing transactions if a translation error occurs. This is the default in Deferred mode.</li> </ul>
Output Report to Process Data	<p>Whether to output the Translation report to process data. Valid values are Yes (Send report to process data) and No.</p>
<b>Note:</b> The following fields are displayed only if Deferred mode was selected.	
Sender ID	<p>Enter a Sender ID to be used as a filter for selecting messages from the database for enveloping. Optional.</p>
Receiver ID	<p>Enter a Receiver ID to be used as a filter for selecting messages from the database for enveloping. Optional.</p>
Envelope	<p>Enter an envelope definition to be used as a filter for selecting messages from the database for enveloping. Optional.</p>
Run as User	<p>The Run As User field only displays as an option if Deferred mode was selected. Type the user ID to associate with the schedule, or</p> <p> click the  icon and select a user ID from the list. ΔValid value is any valid application user ID.</p> <p><b>Note:</b> This parameter allows someone who does not have rights to a specific business process to run it. If you select <i>Admin</i> as the user ID, you will inherit Administrative rights (for this run of the business process only), and enable the scheduled run.</p>
Use 24 Hour Clock Display	<p>If selected, the adapter will use the 24-hour clock instead of the default 12-hour clock.</p>

Field	Description
Schedule <sup>△</sup>	<p>Information about scheduling the service configuration to run in deferred mode. Valid values:<sup>△</sup></p> <ul style="list-style-type: none"> <li>• Do not use schedule <sup>△</sup> If this field is selected, this service does not start a business process and does not run on a schedule.<sup>△</sup></li> <li>• Run based on timer <sup>△</sup> Valid values are the hour and minutes at which to run the service. Indicate whether you want the service to run at startup.<sup>△</sup></li> <li>• Run daily Valid values are the hour and minutes at which to run the service, daily. You can also specify a time interval. Indicate whether you want the service to run at startup.<sup>△</sup></li> <li>• Run based on day(s) of the week <sup>△</sup> Valid values are the day of the week, the hour, and the minutes at which to run the service. You can also specify a time interval. Indicate whether you want the service to run at startup.<sup>△</sup></li> <li>• Run based on day(s) of the month<sup>△</sup> Valid values are the day of the month (including the last day of the month (LDM)), hour, and the minutes at which to run the service. You can also specify a time interval. Indicate whether you want the service to run at startup.<sup>△</sup></li> </ul>

## GPM Configuration<sup>△</sup>

The following table describes the fields used to configure the EDI Enveloping service in the GPM. You set the default global settings for these parameters in the enveloping.properties file. However, these global settings can be overridden in the EDI Enveloping service by using the following parameters in the GPM. Use these parameters when it is necessary to adjust the lock times and lock wait times for specific business processes.

Field	Description
Config	Name of the service configuration. Required.

Field	Description
EDIENVELOPE_MAX_LOCK_TIME	Maximum amount of time that the EDI Enveloping service will hold a lock on an enveloping process. This parameter enables you to fine tune your EDI processing. You can set this parameter in <code>envelope.properties.in</code> , and then use the same parameter here to override the properties file setting where necessary. The default value is 86400 seconds. This parameter is used in Deferred mode to avoid situations where enveloping service releases the lock before it is done processing a message—possibly due to slow response from the database. If there are concurrent EDI enveloping business processes running, an EDI Enveloping service in another business process could possibly retrieve the same message from the database and process it, which would result in duplicate outbound messages.
EDIENVELOPE_LOCK_WAIT_TIME	Maximum amount of time that the EDI Enveloping service will wait for a lock before returning an error and reporting failure to obtain the lock. The service would use this parameter value when another EDI Enveloping business process has the lock, which causes this EDI Enveloping service instance to have to wait for the lock. The default value is 86400 seconds. This parameter is used in Deferred mode only.

## Correlation Data

The EDI Enveloping service and EDI Deenveloping service automatically collect the following correlation information from EDI messages:

Category	Data Collected
Sender ID	<ul style="list-style-type: none"> <li>InterchangeSenderID</li> <li>GroupSenderID</li> <li>TransactionSenderID</li> </ul>
Receiver ID	<ul style="list-style-type: none"> <li>InterchangeReceiverID</li> <li>GroupReceiverID</li> <li>TransactionReceiverID</li> </ul>
Control Numbers	<ul style="list-style-type: none"> <li>InterchangeControlNumber</li> <li>GroupControlNumber</li> <li>TransactionControlNumber</li> </ul>
Acknowledgement Requested	<ul style="list-style-type: none"> <li>InterchangeAckRequested</li> <li>GroupAckRequested</li> </ul>
Acknowledgement Status	<ul style="list-style-type: none"> <li>InterchangeAckStatus</li> <li>GroupAckStatus</li> </ul>

Category	Data Collected
Standard	Standard – values are CII, EDIFACT, VDA, RND, SWIFT, and X12
ID	<ul style="list-style-type: none"> <li>• FunctionalID</li> <li>• TransactionSetID</li> </ul>
Versions	<ul style="list-style-type: none"> <li>• InterchangeVersion</li> <li>• GroupVersion</li> <li>• TransactionVersion</li> </ul>
Date and Time	<ul style="list-style-type: none"> <li>• InterchangeDateTime</li> <li>• GroupDateTime</li> </ul>
Overdue Time	<ul style="list-style-type: none"> <li>• InterchangeOverdueTime</li> <li>• GroupOverdueTime</li> </ul>
Level	Level – values are Interchange, Group, and Transaction
Direction	Direction – values are Inbound and Outbound
Envelope Type	<ul style="list-style-type: none"> <li>• InterchangeEnvelopeType</li> <li>• GroupEnvelopeType</li> <li>• TransactionEnvelopeType</li> </ul>
Envelope Name	<ul style="list-style-type: none"> <li>• InterchangeEnvelopeName</li> <li>• GroupEnvelopeName</li> <li>• TransactionEnvelopeName</li> </ul>
Envelope Version	<ul style="list-style-type: none"> <li>• InterchangeEnvelopeVersion</li> <li>• GroupEnvelopeVersion</li> <li>• TransactionEnvelopeVersion</li> </ul>
Compliance Status	<ul style="list-style-type: none"> <li>• InterchangeComplianceStatus</li> <li>• GroupComplianceStatus</li> <li>• TransactionComplianceStatus</li> </ul> Values are OK and NOT OK
Test Mode	TestMode – values are Production, Test Data and Information
Counts	<ul style="list-style-type: none"> <li>• TransactionCount</li> <li>• GroupCount</li> </ul>
Container Doc ID	ContainerDocID

The information for these correlations is automatically collected at each enveloping and deenveloping stage for a message, which facilitates tracking individual messages as they move through the application.

There is no setup required for using the correlation information collected by the EDI Enveloping and EDI Deenveloping services. After you use one of these services in a business process, the information is available through the Correlation Search option.

## Using Wildcards in Enveloping

As a way to help reduce the number of envelopes you need to create and use, the EDI Enveloping and EDI Develope services support use of an asterisk (\*) as a wildcard character in mandatory envelope fields for X12 and EDIFACT only. By using wildcards, you can set up one set of envelopes that can be used for multiple trading partners. If certain trading partners have specific requirements, you can still have envelopes that pertain just to them, and the EDI Enveloping service chooses the envelope that is the best match. In other words, the envelope that has the most matches to specific fields in the data (for example Receiver ID, Receiver ID Qualifier), is the one selected.

## Updated Envelope Translation Error Messages

When you are translating a batch of documents and need to differentiate between translation errors that occur because of empty sets and translation errors that occur when all sets are bad, you can add a property to the `customer_overrides.properties` file to enhance your error reporting by using the following procedure:

1. In the `install_dir/install/properties` directory, locate the `customer_overrides.properties` file.
2. Open the `customer_overrides.properties` file in a text editor.
3. Add this value:  
`enveloping.enveloping.X12.useUpdatedEnvelopeTranslationErrorMsgs=true`
4. Save the `customer_overrides.properties` file.

---

## Chapter 16. EDI Overdue Acknowledgment Check Service

The following table provides an overview of the EDI Overdue Acknowledgment Check service:

System name	OverdueAckType
Graphical Process Modeler (GPM) categories	All Services, EDI
Description	The EDI Overdue Acknowledgment Check service finds outbound EDI groups and interchanges whose inbound functional acknowledgments are overdue, and creates a status report listing them.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	No
Application requirements	No
Initiates business processes?	None
Invocation	Runs as part of the OverdueAckCheck business process.
Business process context considerations	No
Returned status values	<ul style="list-style-type: none"><li>• Success – The service has run successfully. Anything found to be overdue is listed in the status report.</li><li>• Error – The service encountered a database error.</li></ul>
Restrictions	No

### How the EDI Overdue Acknowledgment Check Service Works

The EDI Overdue Acknowledgment Check service works with the predefined Overdue Acknowledge Check business process.

**Note:** If the input document character encoding is specified, it overrides the encoding specified in the map. The output document content type and character encoding are set based on the information contained in the map.

### Implementing the EDI Overdue Acknowledgment Check Service

To implement the EDI Overdue Acknowledgment Check service, complete the following tasks:

1. Activate your license for the EDI Overdue Acknowledgment Check service. For information, see *Managing Services and Adapters*.
2. Create an EDI Overdue Acknowledgment Check service configuration. For information, see *Managing Services and Adapters*.
3. Configure the EDI Overdue Acknowledgment. For information, see *Configuring the EDI Overdue Acknowledgment Check Service*.

4. Use the EDI Overdue Acknowledgment Check service in a business process.

### Configuring the EDI Overdue Acknowledgment Check Service

To configure the EDI Overdue Acknowledgment Check service, you must specify settings for the following fields in the application:

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	Select one of the options: <ul style="list-style-type: none"><li>• None – You do not want to include this configuration in a group at this time.</li><li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li><li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li></ul>
Run as User	Enter (or select from the list) the user ID to be associated with business process instances of this service.
Use 24 Hour Clock Display	If selected, the adapter will use the 24-hour clock instead of the default 12-hour clock.



Field	Description
Schedule△	<p>Information about scheduling the service configuration to run. Valid values:△</p> <ul style="list-style-type: none"> <li>• Do not use schedule △ If this field is selected, this service does not start a business process and does not run on a schedule.△</li> <li>• Run based on timer △ Valid values are the hour and minutes at which to run the service. Indicate whether you want the service to run at startup.△</li> <li>• Run daily Valid values are the hour and minutes at which to run the service, daily. You can also specify a time interval. Indicate whether you want the service to run at startup.△</li> <li>• Run based on day(s) of the week △ Valid values are the day of the week, the hour, and the minutes at which to run the service. You can also specify a time interval. Indicate whether you want the service to run at startup.△</li> <li>• Run based on day(s) of the month△ Valid values are the day of the month (including the last day of the month (LDM)), hour, and the minutes at which to run the service. You can also specify a time interval. Indicate whether you want the service to run at startup.△</li> </ul>



## Chapter 17. EDI Post Processor Service

The following table provides an overview of the EDI Post Processor service:

System name	EDIPostProcessorType
Graphical Process Modeler (GPM) categories	All Services, EDI
Description	Handles deferred sequence checking and acknowledgement processing of X12 interchanges, as well as duplicate control number checking. It is called if the inbound envelopes require sequence checking. If sequence checking is not required, acknowledgement processing is done by the X12 Deenvelope service.
Business usage	Its processing is done in parallel with EDI translation, thereby improving system performance.
Usage example	The X12 Deenvelope service determines that an inbound purchase order matches envelope definitions which specify that sequence checking should be performed. The service continues to process the inbound purchase order, performing all required compliance checks except for the sequence check, and puts the inbound interchange in a sequence check queue. The EDI Post Processor service runs after the X12 Deenvelope service completes and picks up all interchanges from the sequence check queue. It performs the sequence check, generates any acknowledgements that are necessary, and invokes the specified business processes to handle the purchase order.
Preconfigured?	No
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	This service is intended to be used in conjunction with the X12 Deenvelope service.
Application requirements	No
Initiates business processes?	Yes
Invocation	Runs only by the X12DeenvelopeUnified business process.
Returned status values	<ul style="list-style-type: none"> <li>• Success – The service will return success if it processes all queued interchanges without finding compliance errors.</li> <li>• Error – The service will return error when it encounters non-compliant data and envelope definitions are set to not allow non-compliance.</li> </ul>

## Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM. It must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.



## Chapter 18. EDIFACT Deenvelope Service

The following table provides an overview of the EDIFACT Deenvelope service:

System Name	DeenvelopeEDIFACT
Graphical Process Modeler (GPM) categories	All Services, EDI > EDIFACT
Description	<p>Handles deenveloping of inbound EDIFACT interchanges. This service performs the following tasks:</p> <ul style="list-style-type: none"> <li>• Compliance checking, except for sequence checking.</li> <li>• Generates acknowledgements if no sequence checking is required.</li> </ul>
Business Usage	<p>Using this service can:</p> <ul style="list-style-type: none"> <li>• Improve performance by deferring sequence checking, as required.</li> <li>• Enable the EDI Processing service to update your database with applicable control numbers.</li> </ul>
Usage Example	<p>An inbound purchase order is received inside an EDIFACT interchange. The EDI envelopes are parsed and the document envelopes that match the envelope data are retrieved. With the document envelopes, this service perform the appropriate action for purchase order, such as starting a business process.</p>
Preconfigured?	No pre-configuration necessary
Third Party Files?	No
Platform Availability	All supported application platforms.
Related Services	If sequence checking required, this service works with the EDI Post Processor service.
Application Requirements	None
Bootstrapping	<p>This service may bootstrap business processes specified in envelope definitions for handling incoming documents, and may also invoke business processes to handle outbound acknowledgements associated with an incoming interchange.</p>
Invocation	<p>This service is invoked inside the EDIFACTDeenvelopeUnified business process. Normally, customers would use the business process, and should not need to directly work with this service.</p>
WFC Considerations	None

## Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM. It must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.



**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

**Note:** All EDI services assign a Unique ID to each log message.

### **Adding Translation Map Name to Process Data**

The EDIFACT Deenvelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The EDIFACT Deenvelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.



## Chapter 19. EDIFACT Envelope Service

The following table provides an overview of the EDIFACT Envelope service:

System name	EnvelopeEDIFACTType
Graphical Process Modeler (GPM) categories	All Services, EDI > EDIFACT
Description	Envelopes outbound EDIFACT interchanges.
Business usage	This service improves performance of EDI EDIFACT by consolidating EnvelopeUNH, EnvelopeUNG, and EnvelopeUNB into a single service.
Usage example	An outbound purchase order is to be sent inside an EDIFACT interchange. The document envelopes that match the SenderID, ReceiverID, and AcceptorLookupAlias specified in upstream EDIEncoder are retrieved. If required by the UNH envelope, translation is performed using the map specified by the envelope. The UNH, UNG, and UNB envelopes are applied to the output of this step. The service will then start a business process if specified in the UNB envelope definition.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	EDI Encoder, EDI Envelope
Application requirements	No
Initiates business processes?	This service may invoke a business process specified in the interchange envelope definition.
Invocation	Runs as part of the EDIFACTEnvelopeUnified business process.
Business process context considerations	No
Returned status values	<ul style="list-style-type: none"> <li>• Translation Error – Translation produced errors△</li> <li>• △No_Documents_To_Envelope – EDIEncoder has not run prior to EDIFACT Envelope service△</li> <li>• △No_Envelope_Defined – The UNH envelope defined has a SenderID, ReceiverID, or AcceptorLookupAlias different from that in the EDIEncoder step of the business process△</li> <li>• △Error – A database or other exception takes place△</li> <li>• △Success – Service returns success if none of the above takes place△</li> </ul>
Restrictions	No

## Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM. It must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

## Using Wildcards in Enveloping

To help reduce the number of envelopes that need to be created and maintained in the system, EDIFACT enveloping allows users to create wildcard envelope definitions. There are two aspects to this feature in outbound processing. The first is the use of an asterisk (\*) in any mandatory field in an outbound envelope. The exception to this rule is when the field is Sender ID, Receiver ID, or a qualifier for one of those fields. For example, in EDIFACT the following fields are conditional, but are considered to be part of the Sender / Receiver ID and therefore must have a “\*” placed in the field if you want to override those values:

- (0008) Interchange Sender Internal Identification
- (0042) Interchange Sender Internal Sub-identification
- (0014) Interchange Recipient Internal Identification
- (0046) Interchange Recipient Internal Sub-identification

The second use is the ability to override values set in an envelope definition through the use of correlations. By using an asterisk in the Sender ID, Receiver ID, and Acceptor Lookup Alias fields, it allows the EDI Encoder Service to match and use that envelope for every document it prepares for enveloping. You may use wildcards for one, two, or all three fields when you define an envelope, and the EDI Encoder will find and use the most specific match when it processes a document.

If an envelope field is set to an asterisk, the EDIFACT Envelope service must obtain the actual value to use from a different source—the correlations. You must provide a correlation for an envelope value that is set to asterisk, but you can also set others. Correlations set on the document for other fields in an envelope override what is in the envelope itself. This enables you to create an envelope with default values that you can override only when desired. The exception to this rule is when the field is Sender ID, Receiver ID, or a qualifier for one of these fields. In these fields, you must define the value as an asterisk in the envelope definition if you want to override it with a correlation, otherwise the value from the envelope is always used.

The following list contains the correlation values that can be set inside of process data prior to calling the Correlation service to override outbound envelope values:

- EDIFACTEnvelopeParms/Out\_InterchangeAcknowledgementOverdueTime
- EDIFACTEnvelopeParms/Out\_InterchangeAcknowledgementOverdueTimeMinutes
- EDIFACTEnvelopeParms/Out\_MessageDecimalMark
- EDIFACTEnvelopeParms/Out\_MessageType
- EDIFACTEnvelopeParms/Out\_MessageVersionNumber
- EDIFACTEnvelopeParms/Out\_MessageReleaseNumber
- EDIFACTEnvelopeParms/Out\_MessageControllingAgency
- EDIFACTEnvelopeParms/Out\_MessageAssociationAssignedCode
- EDIFACTEnvelopeParms/Out\_MessageCodeListDirectoryVersionNumber
- EDIFACTEnvelopeParms/Out\_MessageTypeSubFunctionID

- EDIFACTEnvelopeParms/Out\_MessageCommonAccessReference
- EDIFACTEnvelopeParms/Out\_MessageSequenceOfTransfers
- EDIFACTEnvelopeParms/Out\_MessageFirstAndLastTransfer
- EDIFACTEnvelopeParms/Out\_MessageSubsetID
- EDIFACTEnvelopeParms/Out\_MessageSubsetVersionNumber
- EDIFACTEnvelopeParms/Out\_MessageSubsetReleaseNumber
- EDIFACTEnvelopeParms/Out\_MessageSubsetControllingAgency
- EDIFACTEnvelopeParms/Out\_MessageImplementationGuidelineID
- EDIFACTEnvelopeParms/Out\_MessageImplementationGuidelineVersionNumber
- EDIFACTEnvelopeParms/Out\_MessageImplementationGuidelineReleaseNumber
- EDIFACTEnvelopeParms/Out\_MessageImplementationControllingAgency
- EDIFACTEnvelopeParms/Out\_MessageScenarioID
- EDIFACTEnvelopeParms/Out\_MessageScenarioVersionNumber
- EDIFACTEnvelopeParms/Out\_MessageScenarioReleaseNumber
- EDIFACTEnvelopeParms/Out\_MessageScenarioControllingAgency
- EDIFACTEnvelopeParms/Out\_UseGroups
- EDIFACTEnvelopeParms/Out\_GroupDecimalMark
- EDIFACTEnvelopeParms/Out\_GroupSyntaxVersionNumber
- EDIFACTEnvelopeParms/Out\_GroupApplicationSenderID
- EDIFACTEnvelopeParms/Out\_GroupSenderIDCodeQualifier
- EDIFACTEnvelopeParms/Out\_GroupApplicationRecipientID
- EDIFACTEnvelopeParms/Out\_GroupRecipientIDCodeQualifier
- EDIFACTEnvelopeParms/Out\_GroupControllingAgency
- EDIFACTEnvelopeParms/Out\_GroupApplicationPassword
- EDIFACTEnvelopeParms/Out\_InterchangeSyntaxIdentifier
- EDIFACTEnvelopeParms/Out\_InterchangeSyntaxVersionNumber
- EDIFACTEnvelopeParms/  
Out\_InterchangeServiceCodeListDirectoryVersionNumber
- EDIFACTEnvelopeParms/Out\_InterchangeCharacterEncoding
- EDIFACTEnvelopeParms/Out\_InterchangeSenderID
- EDIFACTEnvelopeParms/Out\_InterchangeSenderIDCodeQualifier
- EDIFACTEnvelopeParms/Out\_InterchangeSenderInternalID
- EDIFACTEnvelopeParms/Out\_InterchangeSenderInternalSubID
- EDIFACTEnvelopeParms/Out\_InterchangeRecipientID
- EDIFACTEnvelopeParms/Out\_InterchangeRecipientIDCodeQualifier
- EDIFACTEnvelopeParms/Out\_InterchangeRecipientInternalID
- EDIFACTEnvelopeParms/Out\_InterchangeRecipientInternalSubID
- EDIFACTEnvelopeParms/Out\_InterchangeControlReference
- EDIFACTEnvelopeParms/Out\_InterchangeRecipientReferencePassword
- EDIFACTEnvelopeParms/Out\_InterchangeRecipientReferencePasswordQualifier
- EDIFACTEnvelopeParms/Out\_InterchangeApplicationReference
- EDIFACTEnvelopeParms/Out\_InterchangeProcessingPriorityCode
- EDIFACTEnvelopeParms/Out\_InterchangeAcknowledgementRequest
- EDIFACTEnvelopeParms/Out\_InterchangeAgreementID
- EDIFACTEnvelopeParms/Out\_InterchangeTestIndicator

- EDIFACTEnvelopeParms/Out\_Una

The following example shows how you might set correlation values in a business process:

```

<operation
name="SetTheCorrelations">
  <participant
name="CorrelationService"/>
  <output
message="Xout">
    <assign to="TYPE">DOCUMENT</assign>

    <assign to="CORRELATION_PATH">/ProcessData/EDIFACTEnvelopeParms/*</assign>

    <assign to="SCOPE" from="'EDI'"/>

    <assign to="." from="*"></assign>

  </output>
  <input message="xin">

    <assign to="." from="*"></assign>

  </input>
</operation>
<operation
name="EDI Encoder">
  <participant
name="EDIEncoder"/>
  <output message="EDIEncoderTypeInputMessage">

    <assign to="AcceptorLookupAlias">837</assign>

    <assign to="EDIStandard">EDIFACT</assign>

    <assign to="ReceiverID">TestA-GS-R</assign>

    <assign to="SenderID">TestA-GS-S</assign>

    <assign to="." from="*"></assign>

  </output>
  <input message="inmsg">

    <assign to="." from="*"></assign>

  </input>
</operation>

<operation name="EDI Envelope">
<participant name="EDIEnvelope"/>
<output message="EDIEnvelopeTypeInputMessage">
  <assign to="MODE">DEFERRED</assign>
  <assign to="RECEIVER_ID">TestA-GS-R</assign>
  <assign to="SENDER_ID">TestA-GS-S</assign>
  <assign to="." from="*"></assign>
</output>
<input message="inmsg">

```

```
<assign  
to="." from="*"></assign>  
  </input>  
  
</operation>
```

After the steps shown in the previous example, you would include the Correlation service to set the values as correlations against your documents, then follow that with the EDI Encoder service.

**Note:** All EDI services assign a Unique ID to each log message.

## **Adding Translation Map Name to Process Data**

The EDIFACT Envelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The EDIFACT Envelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.



---

## Chapter 20. EDIINT Acknowledge Check Service

The following table provides an overview of the EDIINT Acknowledge Check service:

System name	None
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > EDIINT
Description	Determines whether an MDN acknowledgement has been received for an EDIINT message within a business process in the application. If the MDN acknowledgement is not received within a specific period of time or if a negative MDN is received, the service can cause the business process to fail (or it can continue successfully, depending on the service configuration). This service is designed to be used in a business process after a message has been sent. You must always include an EDIINT Message service or EDIINT Pipeline service configuration in a business process whenever you include an EDIINT Acknowledge Check service configuration.
Preconfigured?	No
Requires third party files?	No
Platform availability	All supported application platforms
Related services	<ul style="list-style-type: none"><li>• EDIINT Message service</li><li>• EDIINT Pipeline service</li></ul>
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a business process.
Restrictions	No

### How the EDIINT Acknowledge Check Service Works

The following steps summarize how the EDIINT Acknowledge Check service works within a business process:

1. The EDIINT Acknowledge Check service is used to verify that an acknowledgement was received for a message you sent.
2. The EDIINT Acknowledge Check service processes the information in the database about the EDIINT message sent and determines whether an acknowledgement was received for the EDIINT message.

You can configure the EDIINT Acknowledge Check service to check information in the database in either once, or on a recurring basis, at a specified time interval. The time interval can be configured in one of the following ways:

- Globally
- Using the MDN timeout interval configured in the trading profile (default)

3. The EDIINT Acknowledge Check service responds in one of the following ways:
  - The service fails the step in the business process where the service runs, if an MDN acknowledgement is not received during the time interval it checks (default).  
If you configured the service to function this way, it allows business processes that send EDIINT messages to complete successfully only when an MDN acknowledgement is received for the message sent in the business process.
  - The business process continues normally when the EDIINT Acknowledge Check service runs, whether or not an MDN acknowledgement was received during the specified time interval.  
If you configured the service to function this way, it allows you to determine whether an acknowledgement was received without interrupting the business process.

### Implementing the EDIINT Acknowledge Check Service

To implement the EDIINT Acknowledge Check service, complete the following tasks:

1. Activate your license for the EDIINT Acknowledge Check service.
2. Create an EDIINT Acknowledge Check service configuration.
3. Configure the EDIINT Acknowledge Check service.
4. Use the EDIINT Acknowledge Check service in a business process.

When creating business processes, determine whether you want to stop the business process or execute retry logic if an acknowledgement is not received within a defined time interval. If you want to stop the business process, insert the EDIINT Acknowledge Check service *last* in a business process for sending messages.

### Configuring the EDIINT Acknowledge Check Service

To configure the EDIINT Acknowledge Check service, you must specify settings for the following fields in the application and in the GPM.

**Note:** The field names in parentheses represent the corresponding field names in the GPM. This information is provided for your reference.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.

Field	Description
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul>
Poll(ACSPoll)	EDIINT Acknowledge Check service polls at specific intervals for an acknowledgement. Valid values are Yes (default) and No. Required.
Poll Interval (Seconds)(ACSPollInterval)	Interval at which to poll. Valid value is the number of seconds. Default is 180. Optional.
MDN Timeout (Seconds)(ACSPollTimeout)	Timeout interval during which to expect an acknowledgement. Valid value is the number of seconds. Optional. <b>Note:</b> If you specify the interval when you configure the service, the service configuration will always use this timeout value, regardless of any value specified in the trading profile. Valid value is the number of seconds.
Fail on expired message(ACSFailWFOOnExpiredMessage)	Whether to fail the EDIINT Acknowledge Check service when a message is not acknowledged within the required time. Valid values are Yes (default) and No. Required.
Expire messages in database(ACSExpireMessagesInDB)	Whether the EDIINT Acknowledge Check service updates the status of a transaction to Expired in the database when a message is not acknowledged within the required interval (if polling) or simply not acknowledged if checking only once. Valid values are Yes (default) and No. Required.



## Chapter 21. EDIINT Header Scanning Service

The following table provides an overview of the EDIINT Header Scanning service:

System name	None
Graphical Process Modeler (GPM) categories	All Services
Description	This service parses the header area of EDIINT messages and other MIME-based messages without loading or examining the entire message and outputs the header information to process data. This service is currently used by the EDIINTParse business process to scan message headers prior to parsing the message, to determine whether the sender of the message is an AS2 trading partner for which deferred extraction has been configured. The EDIINTParse business process currently uses this service to help determine whether the sender of the message is an AS2 trading partner for which deferred extraction has been configured. The value from the AS2-To header is needed to determine whether the sender is a trading partner in the AS2 Trading Partner Information database table, and, if so, whether the deferred extraction behavior has been enabled for the sender. This service can be used to scan the headers of MIME-based messages for any purpose.
Business usage	A user has implemented deferred extraction behavior for a trading partner that sends AS2 messages.
Usage example	An example of the usage of this service when you are using the AS2 Edition is as follows: <ol style="list-style-type: none"> <li>1. The application receives an AS2 message.</li> <li>2. A business process is invoked to process the message.</li> <li>3. The business process invokes the EDIINT Header Scanning service to determine whether deferred extraction of the payload should occur.</li> <li>4. The EDIINT Header Scanning parses the header information and outputs it to process data.</li> <li>5. Any situation in which you need to parse the header area of MIME-based messages and load that information to process data.</li> </ol>
Preconfigured?	Yes.
Requires third party files?	No.
Platform availability	All supported application platforms.

Related services	This service is used in conjunction with the EDIINT Pipeline service and the EDIINT MDN Building service.
Application requirements	No.
Initiates business processes?	No.
Invocation	Used in the EDIINTParse business process. Can be used in any business process for which this functionality is desired.
Restrictions	No.

## How the EDIINT Header Scanning Service Works

The following steps summarize how the EDIINT Header Scanning service works within a business process:

1. The EDIINT Header Scanning service parses the header area of each EDIINT message (without loading or examining the message).
2. The EDIINT Header Scanning service outputs the header information to process data.

## Implementing the EDIINT Header Scanning Service

To implement the EDIINT Header Scanning service, complete the following tasks:

1. Activate your license for the EDIINT Header Scanning service.
2. Create an EDIINT Header Scanning service configuration. You can also modify the **EDIINTHeaderScan** predefined service instance.
3. Configure the EDIINT Header Scanning service only once in the user interface and the GPM.
4. Use the EDIINT Header Scanning service in a business process.

## Configuring the EDIINT Header Scanning Service

To configure the EDIINT Header Scanning service, you must specify settings for the following fields in the application one time only.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul>

Field	Description
Show transcripts	Whether to enable extended logging to the advanced status shown in the process monitor. Valid values are Yes (default) and No. Required.
Protocol	The communications protocol through which the message was received. The HTTP Server adapter supplies the value for this parameter when it receives a message. <b>Note:</b> For AS2, this parameter should always be set to <b>http</b> .

## Process Data Example

This example shows the EDIINT Header Scanning service used to output header information to process data:

```
<MIMEEntity name="test2:00.000.00.00:1135fdf492d:72484">
  <EDIINT-Message-Sender>test1_as2</EDIINT-Message-Sender>
  <EDIINT-Message-Recipient>test2_as2</EDIINT-Message-Recipient>
  <content-type>application</content-type>
  <content-subtype>pkcs7-mime</content-subtype>
  <content-type-parameters>
    <content-type-parameter
name="smime-type">enveloped-data</content-type-parameter>
    <content-type-parameter name="name">
smime.p7m</content-type-parameter>
  </content-type-parameters>
  <content-transfer-encoding>
7bit</content-transfer-encoding>
  <content-location/>
  <content-id/>
  <OtherMIMEEntityHeaders>
    <OtherEntityHeader name="host">
<![CDATA[test2:16033]]></OtherEntityHeader>
    <OtherEntityHeader name="user-agent">
    <![CDATA[GIS/PsHttpClientAdapter]]>
</OtherEntityHeader>
    <OtherEntityHeader name="as2-version">
<![CDATA[1.1]]></OtherEntityHeader>
    <OtherEntityHeader name="as2-to">
<![CDATA[test2_as2]]></OtherEntityHeader>
    <OtherEntityHeader name="as2-from">
<![CDATA[test1_as2]]></OtherEntityHeader>
    <OtherEntityHeader name="message-id">
    <![CDATA[<MOKOyama-1477dadd-11363544df4-
-6514test10as2@test1>]]></OtherEntityHeader>
```

```

    <OtherEntityHeader name="date">
      <![CDATA[Mon, 25 Jun 2007 03:57:11 GMT]]>
    </OtherEntityHeader>

    <OtherEntityHeader name="subject">
      <![CDATA[Integrator Message]]>
    </OtherEntityHeader>

    <OtherEntityHeader name="disposition-notification-to">
      <![CDATA[test1_as2]]>
    </OtherEntityHeader>

    <OtherEntityHeader name="disposition-notification-options">
      <![CDATA[signed-receipt-protocol=optional,pkcs7-signature;
signed-receipt-
      micalg=optional,sha1]]>
    </OtherEntityHeader>

    <OtherEntityHeader name="content-length">
      <![CDATA[2305]]>
    </OtherEntityHeader>

    <OtherEntityHeader name="uri">
      <![CDATA[/b2bhttp/inbound/as2]]></OtherEntityHeader>

  </OtherMIMEEntityHeaders>
  <EntityBody
name="unknown">
    <Data/>

    <DocumentID/>
  </EntityBody>
</MIMEEntity>

```



## Chapter 22. EDIINT MDN Building Service

The following table provides an overview of the EDIINT MDN Building service:

System name	None
Graphical Process Modeler (GPM) categories	All Services
Description	This service builds a Message Disposition Notification (MDN) based on information in process data and a specified contract ID. This enables you to perform additional custom operations between message parsing and MDN generation so that you can consider the outcome of those operations before generating the MDN. When the EDIINT Pipeline service is configured to not build MDNs (the <b>Build MDNs</b> parameter is set to <b>No</b> ), the EDIINT Pipeline service propagates MDN building information to business processes launched to extract data. The EDIINT MDN Building service is used in the implementation of deferred extraction. The AS2Extract and MailboxAS2Add services use this service to generate an MDN if deferred extraction is enabled.
Business usage	A user wants exact control over the status reported in an MDN and when the MDN is sent.
Usage example	An example of the usage of this service is as follows: <ol style="list-style-type: none"> <li>1. The application receives an AS2 message.</li> <li>2. A business process is invoked to process the message.</li> <li>3. An attempt is made to archive the message and payload data to remote secure storage.</li> <li>4. The archival attempt fails.</li> <li>5. The MDN Generation service is instructed to generate an MDN with a disposition of Error: unexpected-processing-error because the archive attempt failed.</li> <li>6. The MDN is returned to the trading partner.</li> </ol>
Preconfigured?	Yes.
Requires third party files?	<ul style="list-style-type: none"> <li>• TrustpointAll.jar and TrustpointProviders.jar files from SecurityBuilder PKI-J 3.1</li> <li>• EccpressoAll.jar file from SecurityBuilder Crypto-J 2.3</li> </ul>
Platform availability	All supported application platforms

Related services	EDIINT Pipeline service and EDIINT Message service
Application requirements	No.
Initiates business processes?	No.
Invocation	Runs as part of a user-created (custom) business process.
Restrictions	No.

## How the EDIINT MDN Building Service Works

The following steps summarize how the EDIINT MDN Building service works within a business process:

1. The EDIINT MDN Building service is used to build an MDN based on information in process data and a specified contract ID.
2. The EDIINT MDN Building service uses the production profile in the specified contract ID as the originator information and the consumption profile in the specified contract ID as the recipient information.

**Note:** Contract usage can be reversed to allow you to reuse a contract that was identified for parsing, as long as the specified contract contains all required data.

## Implementing the EDIINT MDN Building Service

To implement the EDIINT MDN Building service, complete the following tasks:

1. Activate your license for the EDIINT MDN Building service.
2. Create an EDIINT MDN Building service configuration.
3. Configure the EDIINT MDN Building service only once in the User Interface and the GPM. You can also modify the **EDIINTMDNBuild** predefined service instance.
4. Use the EDIINT MDN Building service in a business process.

## Configuring the EDIINT MDN Building Service

To configure the EDIINT MDN Building service, you must specify settings for the following fields in the application one time only.

**Note:** The field name in parentheses represent the corresponding field name in the GPM. This information is provided for your reference.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.

Field	Description
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul>
Contract ID(b2b-contract-ID)	The ID of a contract containing information for building messages. Must be the ID of an existing contract. Optional, but must either be specified in the GPM or in the service configuration user interface. <b>Note:</b> Please see <i>Use of Contract ID</i> for more information.

## Parameters Passed from the Business Process to the Service

Use the field definitions in the following table to set up the business process correctly:

**Note:** When you parse a message with the EDIINT Message service or EDIINT Pipeline service and instruct that service not to build an MDN, the values for many fields will be placed in process data after the message is parsed. You can alter these values prior to calling the EDIINT MDN Building service, but any values you input must be acceptable within the EDIINT protocol and supported by the application.

Parameter	Description
b2b-contract-id	The ID of a contract containing information for building messages. Must be the ID of an existing contract. <b>Note:</b> This parameter must either be specified in the GPM or in the service configuration user interface. Please see <i>Use of Contract ID</i> for more information.
EDIINT-MDN-Protocol	The signature format requested for the MDN from the Disposition-Notification-Options header of the message from the EDIINT Message service (or EDIINT Pipeline service). The only acceptable value for this field is <b>pkcs7-signature</b> .
EDIINT-Receipt-Delivery- Protocol	The actual transport protocol that the EDIINT Message service (or EDIINT Pipeline service) determined that the MDN must use. The standard value for this field is <b>http</b> .

Parameter	Description
EDIINT-MIC-Alg	<p>The MIC algorithm requested for the MDN from the Disposition-Notification-Options header of the message from the EDIINT Message service (or EDIINT Pipeline service).</p> <p><b>Note:</b> You should not alter this value unless you plan to supply a received-content-MIC with the supplied algorithm.</p>
EDIINT-Received-Content-MIC	<p>The received-content-MIC that is computed by the EDIINT Message service (or the EDIINT Pipeline service) when processing the message.</p> <p><b>Note:</b> You can alter or add this value if you have a received-content-MIC from another source than the EDIINT Message service or EDIINT Pipeline service or if one of those services has been customized by services so that it does not produce a received-content-MIC.</p>
EDIINT-MDN-Recipient	<p>The receiver of the MDN message from the EDIINT Message service (or EDIINT Pipeline service).</p> <p><b>Note:</b> The EDIINT MDN Building service uses the consumption profile in the specified contract ID as the recipient information.</p>
EDIINT-MDN-Sender	<p>The sender of the MDN message from the EDIINT Message service (or EDIINT Pipeline service).</p> <p><b>Note:</b> The EDIINT MDN Building service uses the production profile in the specified contract ID as the originator (sender) information.</p>
EDIINT-MDN-Disposition	<p>The disposition of the MDN from the EDIINT Message service (or EDIINT Pipeline service).</p> <p><b>Note:</b> Generally this should be the string <b>processed</b>, although there are some situations noted in the AS2 specification in which you could set this parameter to <b>failure</b>. It is recommended that you do not change this parameter without careful thought.</p>

Parameter	Description
EDIINT-MDN-Disposition-Modifier	<p>The disposition modifier of the MDN from the EDIINT Message service (or EDIINT Pipeline service). This is only present if message parsing failed. Valid values are:</p> <ul style="list-style-type: none"> <li>• Error: authentication-failed</li> <li>• Error: decryption-failed</li> <li>• Error: decompression-failed</li> <li>• Error:unexpected-processing-error</li> </ul> <p><b>Note:</b> There are other modifiers permitted by the AS2 specification, but the valid values are the only ones supported by the application. If a message is processed without errors by the EDIINT Message service, the EDIINT-MDN-Disposition is <b>processed</b> and there is no EDIINT-MDN-Disposition-Modifier. In this situation, you can add the disposition modifier <b>Error: unexpected-processing-error</b> in the event that the safe-store operation fails by using an assign like the following:&lt;assign to="EDIINT-MDN-Disposition-Modifier"&gt;Error:unexpected-processing-error&lt;/assign&gt;</p>
EDIINT-MDN-Original-Message-Document-ID	The document identifier of the message the EDIINT Message service (or EDIINT Pipeline service) processed.
EDIINT-MDN-Transaction-Type	The transaction type as determined by the EDIINT Message service (or EDIINT Pipeline service). Valid values are AS1 or AS2.
EDIINT-Receipt-Delivery- Port	The port on which that the EDIINT Message service (or EDIINT Pipeline service) determined the MDN would be sent.
Use-Contract-In-Reverse	This parameter must be set to True if you need to use the same contract the EDIINT Message service or EDIINT Pipeline service uses to parse a message.

## Business Process Example

This example uses an instance of the EDIINT MDN Building Service named **EDIINTMDNBuild**. The default EDIINT parsing business process (**EDIINTParse**) was altered to replace the single step that calls the EDIINT Message Service for parsing the message with the following BPML containing three steps to build and then synchronously send the MDN later. If you are sending the MDN asynchronously or are unsure whether you would send the MDN synchronously and asynchronously, you will have to invoke a different sending process (for example, HTTPAsyncSend) or add logic for making the synchronous or asynchronous decision and invoking the appropriate process.

```
<operation name="One">
  <participant name="EDIINTPipelineParse"/>
  <output message="noopout">
```

```

        <assign
to="." from="*"></assign>
        <assign
to="ShowTranscripts">true</assign>

        <assign to="DontBuildMDN">true</assign>

</output>
    <input message="noopin">

        <assign to="." from="*"></assign>

</input>
</operation>

<operation name="Two">
    <participant
name="EDIINTMDNBuild"/>
    <output message="noopout">

        <assign to="." from="*"></assign>

        <assign to="ShowTranscripts">true</assign>

</output>
    <input message="noopin">

        <assign to="." from="*"></assign>

</input>
</operation>

<operation name="InvokeSendBP">
    <participant
name="InvokeBusinessProcessService" />

    <output message="Xout">
        <assign
to="INVOKE_MODE">INLINE</assign>
        <assign
to="NOTIFY_PARENT_ON_ERROR">ALL</assign>

        <assign to="WFD_NAME">HTTPSyncSend</assign>

        <assign to="." from="*"></assign>

</output>
    <input message="Xin"
>
        <assign to="." from="*"></assign>

</input>
</operation>

```

## External System Interaction

An external system is responsible for originating the AS2 message that the MDN is acknowledging. Many external systems employ an MDN timeout feature to determine whether an AS2 message was successfully sent. If the MDN is not received within a certain amount of time, the send is assumed to have failed. The length of this timeout is not standard; it is set by the external system. The external system also decides the actions that are taken if the MDN is not received in the specified length of time. For example, the external system might resend the

message or perform some sort of manual intervention. The application has no control over any actions taken by the external system.

**Note:** MDNs are intended to express the results of message handling within the context of the AS2 protocol. Therefore, including the result of operations outside of the AS2 protocol is not intended in the MDN functionality. You should use great care in determining which other operations, if any, are considered in the generation of MDNs.

## Use of Contract ID

You can specify a contract ID when you create an EDIINT MDN Building service instance, but it is not necessary. When the EDIINT Message Service parses a message, it typically looks up a contract and the contract ID is assigned to the BPML parameter **b2b-contract-id**. The EDIINT MDN Building service configuration can use the same contract ID (you set this in the **b2b-contract-id** parameter). However, you do need to set the Use-Contract-In-Reverse parameter to reuse the same contract.

**Note:** The Use-Contract-In-Reverse parameter is mandatory if no specific contract is configured in the service configuration

You only need to explicitly assign one contract ID unless you need to use a different contract ID from the contract ID previously defined or you are using the EDIINT Message Service or EDIINT Pipeline service in a manner in which the contract ID is not specified.





## Chapter 23. EDIINT Message Service

The following table provides an overview of the EDIINT Message service:

System name	None
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > EDIINT
Description	<p><i>Electronic Data Interchange-Internet Integration (EDIINT)</i> is a protocol developed by the Internet Engineering Task Force (IETF) for securely transporting messages containing business data over the Internet, using MIME packaging types. There are two types of EDIINT:</p> <ul style="list-style-type: none"> <li>• AS1, which uses SMTP, POP, and IMAP as a transport</li> <li>• AS2, which uses HTTP as a transport</li> </ul> <p>Within a business process in the application, the EDIINT Message service builds and parses EDIINT AS1 and AS2 messages, including plain text, signed, and encrypted data. Communications services, such as the B2B SMTP Client adapter or the B2B HTTP Client adapter, then send or receive the messages within the business process.</p> <p><b>Note:</b> Because of our continuing efforts to improve services and adapters to align with new technology and capabilities, the B2B HTTP Client adapter has entered the retirement process in the application and will be replaced with the HTTP Client adapter and its related services. For more information on the retirement process, see <i>Retiring and Removed Services and Adapters</i>. The EDIINT Message service can also generate and process a signed or unsigned Message Disposition Notification (MDN). Signed MDNs provide non-repudiation of receipt, which occurs when the original sender of a message verifies the signed receipt coming back from the receiver.</p> <p><b>Note:</b> The EDIINT Message service has limitations on message size. In general, the maximum size is less than the available memory. The exception is when parsing messages that are signed after compression using file system buffering. To handle messages of unlimited size or to handle several large messages concurrently, use the EDIINT Pipeline service.</p>
Preconfigured?	No
Requires third party files?	No
Platform availability	All supported application platforms

Related services	<ul style="list-style-type: none"> <li>• EDIINT Acknowledge Check service</li> <li>• EDIINT Pipeline service</li> <li>• EDIINT MDN Building service</li> </ul>
Application requirements	You must create appropriate profiles, communications adapter instances, and business processes for sending MDNs (parsing only).
Initiates business processes?	This service is not an adapter, but it does invoke business processes either after looking up an appropriate contract or being configured to expressly launch a specific process when contracts are not required.
Restrictions	This service requires that you have correctly set up profiles for communications. It also requires that the mail and HTTP server adapters that pass inbound messages to it are configured to retrieve raw messages. If you use the EDIINT Message service in a business process associated with a URL in the HTTP Server adapter, you must set the URL to use raw messages. Otherwise, the HTTP Server adapter will remove the message headers and the EDIINT Message service will not be able to process the message.

## How the EDIINT Message Service Works

The EDIINT Message service parses an EDIINT message in the following sequence of events. The EDIINT Message service:

1. Receives a business document.
2. Looks up profiles based on the contract assigned.
3. Uses the consumption profile in the contract to determine how to encapsulate the payload and what type of MDN (if any) to request.
4. Outputs the message to the primary document.

The EDIINT Message service builds an EDIINT message in the following sequence of events. The EDIINT Message service:

1. Receives the data payload (business document)
2. Builds the business document
3. Verifies any signatures
4. Evaluates the message components
5. Encrypts information
6. Tries to look up a contract to get profile information and keys in the database so it can build the message
7. Builds header information about the type of notification requested (if any)
8. Determines the sender of the message
9. Determines the recipient of the message
10. Sends a message

**Note:** The constructed message is the output primary document.

11. Outputs information to process data for any communications adapters that use profiles (outputs a Profile ID)
12. Outputs information for the EDIINT Acknowledge Check service to process data

## Implementing the EDIINT Message Service

To implement the EDIINT Message service for use in a business process, first determine whether you want to build or parse EDIINT messages (or both), and then complete the following processes, as appropriate.

### Implementing the EDIINT Message Service to Build Messages

To implement the EDIINT Message service to build EDIINT messages, complete the following tasks:

1. Activate your license for the EDIINT Message service. For information, see *Obtaining a License File*.
2. Create two trading profiles. That is, one to represent a consumption profile and one to represent a production profile:
  - One trading profile should include your IDs and keys.
  - The second trading profile should include the ID for the trading partner and certificates.
3. Create a contract for sending EDIINT messages to a trading partner. Assign the information for trading partner to the consumption profile, and assign your information to the production profile.

**Note:** You must supply a contract ID. The production profile in the contract is used for the originator information. The identifier in the identity of the production profile is used as the value of AS2-From. The consumption profile in the contract is used for the recipient information. The identifier from the identity in the consumption profile is used as the value of AS2-To.

4. Create an EDIINT Message service configuration (selecting the Build action), and assign it the appropriate contract.

**Note:** For every contract you create for sending EDIINT messages, you must create a configuration of the EDIINT Message service and assign the appropriate contract to it.

5. Activate your license for one of the following communications services:
  - B2B SMTP Client adapter
  - HTTP Client adapter

**Note:** AS2 business processes that used the retired B2B HTTP Client adapter (specifically the business processes used to send MDNs back to requester) have been updated to use the HTTP Client adapter. Previously the EDIINT Message Service was configured to use the HTTPAsynchSend process to send back the asynchronous MDN, but now the EDIINT Pipeline Service uses the HTTPClientSend process, which has been updated to have a default response timeout for this purpose, and which uses the HTTP Client adapters.

6. Create a configuration of the communications service and assign it the appropriate contract name.

**Note:** It is not necessary to configure the communications services for outbound transport. The EDIINT Message service communicates the

information about where to send the message to the appropriate communications service by providing the appropriate transport information from the trading profile.

7. Create a business process that:
  - Invokes the EDIINT Message service configuration that you created to build EDIINT messages.
  - Invokes the communications service you configured to send the messages.
  - Uses the EDIINT Acknowledge Check service to wait for any acknowledgement.
8. To indicate whether an MDN acknowledgement has been received for an EDIINT message within a specified time period, include the EDIINT Acknowledge Check service in your business process.

**Note:** If you are using AS2 with synchronous MDNs, this business process must also include a step that uses a configuration of the EDIINT Message service for parsing after the send action.

## Implementing the EDIINT Message Service to Parse

The following behavior is standard for message parsing with the EDIINT Message service:

- The service can check for duplicate messages based on message ID. When a duplicate message is found, the service sends back the existing MDN, if available, and does not send the data further.
- The service can either build and send an MDN or output information to process data for the MDN building service to use later. The default is to build the MDN.
- The service can either send an MDN (if it builds an MDN) or put the MDN into the current context as a document. The default is to send the MDN. There are 3 business processes that you must configure for each service instance for sending MDNs: one for asynchronous HTTP MDNs, one for synchronous HTTP MDNs, and one for SMTP MDNs (while this is not often used, it is permitted by the AS2 specification).

To implement the EDIINT Message service to parse EDIINT messages, complete the following tasks:

1. Activate your license for the EDIINT Message service.
2. Create business processes for sending SMTP MDNs, or synchronous or asynchronous HTTP MDNs. These simple business processes invoke configurations of the HTTP Server adapter, HTTP Client adapter, or SMTP Client adapter.
3. Create a business process for parsing that invokes the EDIINT Message service configuration that you create in step 5.5
4. Create a contract for receiving and parsing messages.
  - The consumption profile represents your organization.
  - The production profile represents your trading partner.

**Note:** Contract ID can either be looked up using AS2 identifiers in message or provided directly as a BPML parameter. Default is to look up the contract ID. This is done using AS2-To and AS2-From values to find a contract by extensions in the database. Production profile in contract ID is used as the originator information. The Consumption profile in contract ID is used as recipient information

5. Create a configuration of the EDIINT Message service for parsing.
6. Configure the EDIINT Message service.
7. Activate your license for one of the following communications services:
  - HTTP Server adapter
  - Mail Server adapter
8. Create a URL and set it up to retrieve raw messages.
9. Assign the business process you created in step 6 to the URL. The business process invokes the EDIINT Message (Parsing) service configuration that you created in step 5.
10. Create configurations for the communications services you want to use. Set them up to retrieve raw messages. Add them to the business process you created in step 6.
11. To determine whether an MDN acknowledgement has been received for an EDIINT message within a specified time period, include the EDIINT Acknowledge Check service in your business process.

## Configuring the EDIINT Message Service

To configure the EDIINT Message service, you must specify settings for the following fields in the application.

**Note:** Names in parentheses represent the corresponding field names in the GPM. This information is provided for your reference.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	<p>Select one of the options:</p> <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul> <p><b>Note:</b> See <i>Managing Services and Adapters</i>.</p>
Action (Action)	Valid value is Parse or Build. Required.
<b>Parse Parameters:</b>	

Field	Description
Don't Build MDN(DontBuildMDN)	Whether to build an MDN. If you set this value to True, the service outputs information that is needed by the EDIINT MDN Building service to process data, but does not actually build the MDN. This gives you the opportunity to change the information, if necessary, before actually calling the EDIINT MDN Building service. Valid values are True and False (default). Required. <b>Note:</b> If you set this value to True, it will take precedence over <b>Put MDNs In Business Document Context</b> . Do not set both Don't Build MDN an Put MDNs In WFC to True.
Require Addresses(RequireAddresses)	Valid value is Yes or No. Required. <ul style="list-style-type: none"> <li>• Yes – Service fails if a contract for an inbound message cannot be looked up.</li> <li>• No – Service attempts to parse a message, and if successful, tries to run the default business process.</li> </ul>
Default Data business process (if not requiring addresses)(DefaultDataWorkflow)	Name of a business process to run if No is selected for Require Addresses and no contract has been found for an inbound message. Required if not requiring addresses.
SMTP MDN business process (SMTPMDNWorkflow)	Name of the business process for sending SMTP MDNs. If you are using AS1, you must select a business process from the list. Required.
HTTP Synchronous MDN business process (HTTPSYNCDNWorkflow)	Name of the business process for sending synchronous HTTP MDNs. If you are doing AS2 with synchronous MDNs, you must select a business process from the list. Required.
HTTP Asynchronous MDN business process (HTTPASYNMDNWorkflow)	Name of a business process for sending asynchronous HTTP MDNs. If you are using AS2 with asynchronous MDNs, you must select a business process from the list. Required.
Put documents in business process context(PutDocsInWFC)	Put business documents extracted from messages after parsing into the current business process context. Valid values are Y and N (default). Optional.
Put MDNs in business process context(PutMDNsInWFC)	Put MDNs created after parsing messages into the current business process context. Valid values are Y and N (default). Optional.
Put MDNs in Business Process Context(PutMDNsInBusinessProcess Context)	Put MDNs created after parsing messages into the current business process context. Valid values are True and False. Default is False. Required.

Field	Description
Use supplied contract(UseSuppliedContract)	<p>Use if you do not want the service to look up a contract for parsing, but to expect one to be supplied in process data when the service runs. Valid values are Yes and No (default). Optional.</p> <p><b>Note:</b> If you must use duplicate contracts, you need to force the EDIINT Message service to not look up the contract information by choosing No for this parameter.</p>
Process duplicate messages(ProcessDuplicateMessages)	<p>Send business data from messages with duplicate message IDs into the application. Valid values are Yes and No (default). Optional.</p> <p><b>Note:</b> If you select Yes, you may receive duplicate data.</p>
Update expired messages(UpdateExpiredTransactions)	<p>Whether to update the status of EDIINT transactions that currently have a status of Expired when an acknowledgement is received. Valid values are Yes and No (default). Optional.</p> <p><b>Note:</b> If you select No, and an MDN arrives for an expired transaction, status information is placed in the additional data field of the transaction record.</p>
<b>Build Parameter:</b>	
Contract ID(b2b-contract-id)	<p>Identification of a contract containing information for building messages. This field is required to build EDIINT messages. It is not available if you select Parse.</p>





## Chapter 24. EDIINT Pipeline Service

The following table provides an overview of the EDIINT Pipeline service:

System name	EDIINTPipelineService
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > EDIINT
Description	<p><i>Electronic Data Interchange-Internet Integration (EDIINT)</i> includes a family of protocols developed by the Internet Engineering Task Force (IETF) for securely transporting messages containing business data over the Internet, using MIME packaging types. The EDIINT Pipeline service builds and parses EDIINT AS2 messages, including plain text, signed, compressed, and encrypted data. It also generates Message Disposition Notifications (MDNs) and launches a business process to send them. The EDIINT Pipeline service can build and parse large documents greater than 2 GB. The EDIINT Pipeline service also builds and parses messages that conform to the AS2 multiple attachment draft specification.</p> <p><b>Note:</b> The application also includes the EDIINT Message service, which builds and parses both EDIINT AS1 and EDIINT AS2 messages, but does not have the large file building and parsing capabilities of the EDIINT Pipeline service. EDIINT Pipeline service is the replacement for the EDI Message service if you need to handle unlimited file sizes or many large messages. Additionally, when this service is configured to not build MDNs (the <b>Build MDNs</b> parameter is set to <b>No</b>), this service propagates MDN building information to business processes launched to extract data. The following information is also propagated by this service: information needed by the EDIINT MDN Building service to build an MDN and the name of the business process configured for sending the MDN on the requested protocol.</p>
Business usage	You can use this service in two ways by embedding it in a business process: To parse EDIINT AS2 messages. To build EDIINT AS2 messages, which can then be sent to a trading partner.
Usage examples	See <i>Business Process Examples</i> .
Preconfigured?	Configurations of the EDIINT Pipeline service for building and parsing messages are installed with the application.

Requires third party files?	<ul style="list-style-type: none"> <li>TrustpointAll.jar and TrustpointProviders.jar files from SecurityBuilder PKI-J 3.1 provided by Certicom.</li> <li>EccpressoAll.jar file from SecurityBuilder Crypto-J 2.3 provided by Certicom.</li> </ul>
Platform availability	All supported application platforms
Related services	<ul style="list-style-type: none"> <li>EDIINT Acknowledge Check service. Use this service when MDNs are expected for messages built with the EDIINT Pipeline service.</li> <li>EDIINT Message service</li> <li>EDIINT MDN Building service</li> <li>EDIINT Header Scanning service</li> </ul>
Application requirements	You must create appropriate profiles, communications adapter configurations, and business processes for returning MDNs (parsing only).
Initiates business processes?	This service invokes business processes for building and sending messages and either after looking up an appropriate contract or being configured to launch a specific process when contracts are not required. When this service is used to build an EDIINT message, typically new business processes are not launched.
Invocation	This service is typically invoked within a business process which has been started by the HTTP Server adapter when configured to parse EDIINT messages. Typically, you would invoke the service directly from a business process when using it to build an EDIINT message.
Returned status values	Success, Failure
Restrictions	<ul style="list-style-type: none"> <li>You must have profiles and contracts configured for the trading partner.</li> <li>You must have the HTTP Server adapter configurations that pass inbound messages to this service configured to send raw messages.△</li> </ul>
Persistence level	None

Testing considerations	To test and verify that the EDIINT Pipeline service can parse an AS2 message: Capture a copy of the EDIINT messages being received from the trading partner. Do this by inserting a configuration of the File System adapter immediately before the EDIINT Pipeline service in the business process used to process AS2 messages received from a trading partner. Configure the File System adapter to save the captured EDIINT message to a file on disk, where you can examine it later. To replay this captured message, create a separate test business process and configure a File System adapter to retrieve the saved message from the previous step. As the next step in the business process, use the EDIINT Pipeline service to parse the replayed message. This mimics the process of receiving and processing an AS2 message. By default the EDIINT Pipeline service ignores duplicate messages (as identified by the embedded Message-ID header.) To change this behavior for testing, edit the Message-ID header in the message (using a binary editor) and change it to a unique value each time this message is replayed in step (2) above.
------------------------	--

## When to Use the EDIINT Pipeline Service

The EDIINT Pipeline service and EDIINT Message service perform similar functions, but there are some situations where one or the other should be used.

- Use the EDIINT Pipeline service if:
  - You need to build or parse large (greater than 2 GB) AS2 messages.
  - You need to build or parse messages with multiple AS2 attachments.
  - You need to build or parse many large messages concurrently.
- Use the *EDIINT Message service* if you are building or parsing AS1 messages. The EDIINT Pipeline service does not support AS1 messages.

## How the EDIINT Pipeline Service Works

Use the EDIINT Pipeline service to build or parse an EDIINT message. The EDIINT Pipeline service constructs a “pipeline” that consists of a set of nested data transformers, one for each MIME entity in the stack of messages. To build AS2 messages with multiple attachments, the EDIINT Pipeline service accepts an XML structure that describes the set of attachment documents and document properties. The EDIINT Pipeline service then uses the XML structure to build a multipart related MIME entity that contains the attachment documents.

When parsing messages, the EDIINT Pipeline service handles the payload documents in two ways, depending on how it is configured: it can launch the business process specified in the contract used to parse the message with the payload document as the primary document or it can add the payload document to the current business process context. The default behavior is to launch the business process that is specified in the contract used to parse the message with the payload document as the primary document. When parsing a message with

multiple attachments that conforms to the AS2 specification for multiple attachments, the EDIINT Pipeline service outputs enough information to ProcessData so the services or processes occurring after this service is run can determine whether the attachments are related and came from the same AS2 message.

Several predefined AS2 business processes use the EDIINT Pipeline service, as opposed to the older EDIINT Message service. The EDIINT Pipeline service has the same functional behavior as the older service, but it implements streaming to allow for the handling of larger documents. The following packaged business processes were updated:

- AS2SendASyncMDN
- AS2SendNoMDN
- AS2SendSyncMDN
- MailboxAS2SendAsyncMDN
- MailboxAS2SendNoMDN
- MailboxAS2SendSyncMDN
- EDIINTParse

**Note:** If you have customized any of the attached business processes, then you will need to add your modifications to the updated business processes after installing the build. Your customized business process is not lost, but the updated business processes are set to the default.

## How the EDIINT Pipeline Service Works In Parse Mode

The EDIINT Pipeline service parses an EDIINT message in the following sequence of events. The EDIINT Pipeline service:

1. Receives a message
2. Determines the origin of the message
3. Determines the intended receiver of the message
4. Extracts header information about the type of notification requested (if any)
5. Tries to look up a contract to get profile information and keys in the database so it can process the message
6. Evaluates the message components
7. Decrypts encrypted information
8. Verifies any signatures
9. Breaks the message down to the level of the data payload (business document)
10. Sends the data payload (business document) to the business process specified in the contract
11. Returns an MDN back to the sender of the EDIINT message, either synchronously or asynchronously as requested by the sender.

## How the EDIINT Pipeline Service Works In Build Mode

The EDIINT Pipeline service builds an EDIINT message in the following sequence of events.

**Note:** The EDIINT Message Service has limitations on message size. The maximum size is less than the available memory in most cases. The exception is when

parsing messages that are signed after compression when using file system buffering. To handle messages of unlimited size or to handle several large messages concurrently, use the EDIINT Pipeline service.

The EDIINT Pipeline service does the following:

**Note:** The EDIINT Pipeline Service uses the HTTPClientSend process to send back the asynchronous MDN, which has been updated to have a default response timeout for this purpose, and which uses the HTTP Client adapters.

1. Invoke the EDIINT Pipeline service with the EDI purchase order document.
2. EDIINT Pipeline service looks up the contract established with the trading partner to determine the security attributes to use when creating the EDIINT message.
3. EDIINT Pipeline service returns the newly created EDIINT message to the business process.
4. Business process invokes an HTTP client adapter to deliver the EDIINT message.
5. If a synchronous MDN is expected for this request, the HTTP response is parsed with the EDIINT Pipeline service.
6. Business process invokes the EDIINT Acknowledge Check service to confirm that the expected MDN has been received.

## Implementing the EDIINT Pipeline Service

To implement the EDIINT Pipeline service for use in a business process, first determine whether you want to build or parse EDIINT AS2 messages (or both), and then complete the following processes, as appropriate.

### Implementing the EDIINT Pipeline Service to Build Messages

To implement the EDIINT Pipeline service to build EDIINT messages, complete the following tasks:

1. Activate your license for the EDIINT Pipeline service. For information, see *Obtaining a License File*.
2. Create two trading profiles: one to represent a consumption profile and one to represent a production profile:
  - One trading profile should include your IDs and keys.
  - The second trading profile should include the ID for the trading partner and certificates.
3. Create a contract for sending EDIINT messages to a trading partner. Assign the information for trading partner to the consumption profile, and assign your information to the production profile.
4. Create an EDIINT Pipeline service configuration (selecting the Build action), and assign it the appropriate contract. You can also modify the **EDIINTPipelineBuild** predefined service instance.

**Note:** For every contract you create for sending EDIINT messages, you can assign the contract in BPML. The AS2 Edition reuses the same service instances for many contracts. However, you can assign the contract as a service instance parameter if you want to use a dedicated service instance for a specific contract.

5. Activate your license for the HTTP Client adapter.

6. Create a configuration of the HTTP Client adapter and assign it the appropriate contract name.

**Note:** It is not necessary to configure the communications services for outbound transport. The EDIINT Pipeline service communicates the information about where to send the message to the appropriate communications service by providing the appropriate transport information from the trading profile.

7. Create a business process that:
  - Invokes the EDIINT Pipeline service configuration that you created to build EDIINT messages.
  - Invokes the communications service you configured to send the messages.
  - Uses the EDIINT Acknowledge Check service to wait for any acknowledgement.
8. To indicate whether an MDN acknowledgement has been received for an EDIINT message within a specified time period, include the EDIINT Acknowledge Check service in your business process.

**Note:** If you are using AS2 with synchronous MDNs, this business process must also include a step that uses a configuration of the EDIINT Pipeline service for parsing after the send action.

## Implementing the EDIINT Pipeline Service to Parse Messages

To implement the EDIINT Pipeline service to parse EDIINT messages, complete the following tasks:

1. Activate your license for the EDIINT Pipeline service. For information, see *Obtaining a License File*.
2. Create business processes for sending synchronous or asynchronous HTTP MDNs. These simple business processes invoke configurations of the HTTP Server adapter or HTTP Client adapter.
3. Create a contract for receiving and parsing messages.
  - The consumption profile represents your organization.
  - The production profile represents your trading partner.
4. Create a configuration of the EDIINT Pipeline service for parsing.
5. Configure the EDIINT Pipeline service. You can also modify the **EDIINTPipelineParse** predefined service instance.
6. Activate your license for the HTTP Server adapter.
7. Create a business process for parsing that invokes the EDIINT Pipeline service configuration that you created in step 4.
8. Create a URL and set it up to retrieve raw messages.
9. Assign the business process you created in step 7 to the URL. The business process invokes the EDIINT Pipeline (Parsing) service configuration that you created in step 4.
10. Create configurations for the HTTP Server adapter. Set them up to retrieve raw messages. Add them to the business process you created in step 7.
11. To determine whether an MDN acknowledgement has been received for an EDIINT Pipeline within a specified time period, include the EDIINT Acknowledge Check service in your business process.

## Configuring the EDIINT Pipeline Service

To configure the EDIINT Pipeline service, you must complete the following steps:

1. Select **Deployment > Services > Configuration**.
2. Search for the EDIINT Pipeline service or select it from the list and click **Go!**
3. Click **Edit**.
4. Specify field settings in the Admin Console (Creating or Setting up a Service Configuration in the Admin Console).
5. On the Confirm page, verify that the **Enable Service for Business Processes** check box is selected and click **Finish**.

## Creating or Setting Up a Service Configuration in the Admin Console

To configure the EDIINT Pipeline service, you must specify settings for the following fields in the application user interface one time only.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – You do not want to include this configuration in a group at this time.</li> <li>• Create New Group – You can enter a name for a new group in this field, which will then be created along with this configuration.</li> <li>• Select Group – If you have already created one or more groups for this service type, they are displayed in the list. Select a group from the list.</li> </ul> Optional. <b>Note:</b> See <i>Managing Services and Adapters</i> for more information.
Action	Valid values are Parse or Build. Required.
<b>Parameters for Parse action:</b>	
Require Addresses	Required. Valid values are: <input type="checkbox"/> <ul style="list-style-type: none"> <li>• Yes – Service fails if a contract for an inbound message cannot be looked up (default)</li> <li>• No – Service attempts to parse a message, and if successful, tries to run the default business process.</li> </ul>
Default Data Business Process (If not requiring addresses)	Name of a business process to run if No is selected for <b>Require Addresses</b> and no contract has been found for an inbound message. Must be the name of a business process. Required if <b>Require Addresses</b> is set to No.

Field	Description
SMTP MDN Business Process	The name of the business process for sending SMTP MDNs. If you are using AS1, you must select a business process from the list. Required.
HTTP Synchronous MDN Business Process	Name of the business process for sending synchronous HTTP MDNs. If you are using AS2 with synchronous MDNs, you must select a business process from the list. Required.
HTTP Asynchronous MDN Business Process	Name of a business process for sending asynchronous HTTP MDNs. If you are using AS2 with asynchronous MDNs, you must select a business process from the list. Required.
Build MDNs	Whether to build an MDN. If you set this value to <b>Yes</b> , the service outputs information that is needed by the EDIINT MDN Building service to process data, but does not actually build the MDN. This gives you the opportunity to change the information, if necessary, before actually calling the EDIINT MDN Building service. Valid values are Yes and No (default). Required. <b>Note:</b> If you set <b>Build MDNs</b> to No, it will take precedence over <b>Put MDNs in business process context</b> . Do not set both Build MDNs and Put MDNs In business process context to No.
Put documents in business process context	Put business documents extracted from messages after parsing into the current business process context. Valid values are Yes and No. Default is No. Required.
Put MDNs in business process context	Put MDNs created after parsing messages into the current business process context. Valid values are Yes and No. Default is No. Required. <b>Note:</b> If you set Build MDNs to No, it will take precedence over Put MDNs in business process context. Do not set both Build MDNs and Put MDNs In business process context to No.
Use supplied contract	Use if you do not want the service to look up a contract for parsing, but to expect one to be supplied in process data when the service is invoked. Valid values are Yes and No. Default is No. Required.



Field	Description
Process duplicate messages	<p>Send business data from messages with duplicate message IDs into the application. Valid values are Yes and No. Default is No. Required.</p> <p><b>Note:</b> If you select Yes, you may receive duplicate data.</p> <p><b>Note:</b> Deferred extraction must not be enabled if duplicate suppression (Process Duplicate Messages) is enabled. Conversely, if deferred extraction is enabled, duplicate suppression (Process Duplicate Messages) must not be enabled. These two features are mutually exclusive. If you want to enable duplication suppression, locate the appropriate entry in the ediint.properties file. Copy the entry and paste it into the customer_overrides.properties file, and set the property to YES. After you change the rule and save the customer_overrides.properties file, you must restart the application for the changes to take effect.</p>
Update expired messages	<p>Whether to update the status of EDIINT transactions that currently have a status of Expired when an acknowledgement is received. Valid values are Yes and No. Default is No. Required.</p> <p><b>Note:</b> If you select No, and an MDN arrives for an expired transaction, status information is placed in the additional data field of the transaction record.</p>
Pipeline timeout (Seconds)	<p>Maximum processing time for a request before it is timed out. Default is 1800 (seconds). Required.</p>
<b>Parameter for Build action:</b>	
Contract ID	<p>The ID of a contract containing information for building messages. Must be the ID of an existing contract. Optional.</p>

## Output Messages

The parameters that can be assigned by the service in the business process context (when building messages or MDNs) are listed below:

- B2B-message-mode: (always send for now)
- B2B-raw-message: (always to true)
- B2B-contract-id: (ID of the contract used to build the message)
- B2B-want-response: (always true)
- B2B-raw-response: (true for HTTP synchronous MDNs only)
- xport-B2B-mode: on

## Business Process Examples

The following example business processes illustrate using the EDIINT Pipeline service:

Example 1: Using the EDIINT Pipeline service to build messages:

```
<operation>△  
    <participant name="EDIINTBuild"/>△  
    <output message="noopout">△  
        <assign to="." from="*" />△  
        <assign  
to="Action">build</assign>△  
        </output>△  
    <input message="noopin">△  
        <assign  
to="." from="*" />△  
        </input>△  
</operation>△
```

Example 2: Using the EDIINT Pipeline service to parse messages. This example enables the processing of duplicate messages and assumes that the service instance has been configured for parsing when created:

```
<process  
name="EDIINTParsePipelineAS2">  
    <sequence>  
        <operation name="Parse">  
            <participant  
name="EDIINTPipelineService"/>  
            <output  
message="noopout">  
                <assign to="."  
from="*"></assign>  
                <assign to="ProcessDuplicateMessages">true</assign>  
            </output>  
            <input message="noopin">  
                <assign to="." from="*"></assign>  
            </input>  
        </operation>  
    </sequence>  
</process>
```

## Chapter 25. Generic Deenvelope Service

### CAUTION:

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the Generic Deenvelope service:

System name	GenericDeenvelope
Graphical Process Modeler (GPM) categories	All Services, EDI
Description	<p>Parses a SWIFT, CHIPS, Fedwire, TRADACOMS, VDA, or RND envelope; checks it for compliance; and extracts the transaction sets that it contains so they can be further processed by the application.</p> <p><b>Note:</b> In previous releases, the document lifespan default was zero so that when the workflow expired, all associated documents were purged/archived with the workflow. Now the lifespan is configurable for documents awaiting acknowledgement (the default is 30 days) and standards that use the Generic Deenvelope and Generic Envelope services. You can change the default lifespan by editing the <b>document.lifespan</b> property in the <b>enveloping.properties</b> file. The document lifespan of the outbound document is automatically reset to zero after the acknowledgement for the document is received or if the user manually accepts the acknowledgement.</p>
Business usage	<p>Used within the SWIFT Deenvelope, CHIPS Deenvelope, Fedwire Deenvelope, TRADACOMS Deenvelope, VDADeenvelope, and RNDDeenvelope business processes to deenvelope the data.</p> <p><b>Note:</b> The Generic Deenvelope service assumes that SWIFT expects a starting CRLF (carriage return/line feed) but not an ending CRLF.</p>
Usage example	<p>A typical scenario is one where SWIFT, CHIPS, Fedwire, TRADACOMS, VDA, or RND data must be received from a trading partner. The data must be deenveloped in order to extract identifying batch and interchange data. The SWIFT Deenvelope, CHIPS Deenvelope, Fedwire Deenvelope, TRADACOMS Deenvelope, RNDDeenvelope, or VDADeenvelope business process calls GenericDeenvelope to provide these enveloping services.</p>

Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms
Invocation	Currently run only by the SWIFT Deenvelope, CHIPS Deenvelope, Fedwire Deenvelope, TRADACOMS Deenvelope, RNDDeenvelope, and VDADeenvelope business processes.
Business process context considerations	Translation occurs using the SWIFT_Deenvelope, CHIPS_Deenvelope, Fedwire_Deenvelope, TRADACOMS_Deenvelope, RNDDeenvelope, or VDA_Deenvelope map. That map updates ProcessData with the SenderId, ReceiverId, MessageType, EnvelopeType, and ControlNumber. The service extracts this information from ProcessData using the business process context and finds the associated envelope definition. The envelope definition provides the name of another map used for compliance checking and translation.

## Document Tracking Levels and Performance

You can improve EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance improvement is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search

- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM; it must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

Setting	Description
none	Provides the best EDI performance with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking, and EDI Reporting are nonfunctional.
basic	Provides a good EDI performance while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking, and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking, and EDI Reporting are fully functional.

## Adding Translation Map Name to Process Data

The Generic Deenvelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The Generic Deenvelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.

## Splitting Documents in a Delimited EDI File

When you use the Document Extraction service to split documents in a delimited EDI file, only one set of delimiters is supported. The input file cannot contain multiple documents that each use a different set of delimiters. Therefore, if you have multiple documents using different delimiters, you must first send the data through the EDI Deenvelope service (in Document mode) to split out the individual documents that use differing delimiters. When you invoke the EDI Deenvelope service, the Mode service parameter must be set to Document. This instructs the EDI Deenvelope service not to bootstrap a business process after it finishes splitting the input file. You must also update the customer\_overrides.properties file to include the START and END tag of the documents to be extracted, as well as the delimiters (or the location of delimiters) to use.



## Chapter 26. Generic Envelope Service

**CAUTION:**

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the Generic Envelope service:

System name	GenericEnvelope
Graphical Process Modeler (GPM) categories	All Services, EDI
Description	<p>Applies a SWIFT, CHIPS, Fedwire, TRADACOMS, RND, or VDA envelope to the set of transaction set messages that it receives as input, according to the envelope definition contained in the EDI State extension of the messages that were originally placed on the enveloping business process.</p> <p><b>Note:</b> In previous releases, the document lifespan default was zero so that when the workflow expired, all associated documents were purged/archived with the workflow. Now the lifespan is configurable for documents awaiting acknowledgement (the default is 30 days) and standards that use the Generic Deenvelope and Generic Envelope services. You can change the default lifespan by editing the <b>document.lifespan</b> property in the <b>enveloping.properties</b> file. The document lifespan of the outbound document is automatically reset to zero after the acknowledgement for the document is received or if the user manually accepts the acknowledgement.</p>
Business usage	<p>Used within the SWIFT Envelope, CHIPS Envelope, Fedwire Envelope, TRADACOMS Envelope, RNDEnvelope, and VDAEnvelope business processes to envelope the EDI transactions contained in the business process context with SWIFT/CHIPS/Fedwire/TRADACOMS/VDA/RND envelopes that have been preconfigured and saved as translation maps in the application.</p> <p><b>Note:</b> The Generic envelope service assumes that SWIFT expects a starting CRLF (carriage return/line feed) but not an ending CRLF.</p>

Usage example	A typical scenario is one where SWIFT, CHIPS, Fedwire, TRADACOMS, RND, or VDA data must be sent to a trading partner. In preparation of this, the data must be enveloped in order to provide identifying batch and interchange data. The SWIFT Envelope, CHIPS Envelope, Fedwire Envelope, TRADACOMS Envelope, RNDEnvelope, or VDAEnvelope business process calls GenericEnvelope to provide these enveloping services.
Preconfigured?	Yes
Platform availability	All supported application platforms
Related services	No
Application requirements	No
Invocation	Currently run only by the SWIFT Envelope, CHIPS Envelope, Fedwire Envelope, TRADACOMS Envelope, RNDEnvelope, and VDAEnvelope business processes.
Notes	<p>Output messages:</p> <ul style="list-style-type: none"> <li>• No_Documents_To_Envelope – If EDIEncoder is not run prior to running EnvelopeGeneric service△</li> <li>• EnvelopeMapName cannot be null – If EnvelopeMapName parameter is not defined in the envelope definition△</li> <li>• No_Envelope_Defined – If the envelope defined has a Sender ID, Receiver ID, or AcceptorLookupAlias different from that in the EDIEncoder step of the business process△</li> <li>• Document Translation Error – If the document translation step produced errors△</li> <li>• Envelope Translation Error – If the envelope translation step produced errors</li> </ul>

## Document Tracking Levels and Performance

You can improve EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service



- X12 Develope service
- Generic Develope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance improvement is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM; it must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

Setting	Description
none	Provides the best EDI performance with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking, and EDI Reporting are nonfunctional.
basic	Provides a good EDI performance while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking, and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking, and EDI Reporting are fully functional.

### Adding Translation Map Name to Process Data

The Generic Envelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The Generic Envelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.



## Chapter 27. Image Cash Letter Join Service (Build 7002 or higher)

The Image Cash Letter Join service inserts variable length binary images (Type52 records) into Image Cash Letter documents. This service is typically used after translation to join the binary images previously split out by the Image Cash Letter Split service.

**Note:** This service does not enforce or validate the standard.

The following table provides an overview of the Image Cash Letter Join service:

System name	ImageCashLetterSplit
Graphical Process Modeler (GPM) categories	<ul style="list-style-type: none"> <li>• All Services</li> <li>• EDI</li> </ul>
Description	The default behavior is to read through the PrimaryDocument looking for binary image data placeholder records (inserted by the Image Cash Letter Split service) and, if found, join the corresponding binary image back into the Image Cash Letter document. Alternatively, this service can be used to insert binary images based on truncated Type52 records (only contains fields 1-17).
Preconfigured?	A configuration of this service is created when the product is installed.
Requires third party files?	No
Platform availability	All supported application platforms
Related services	Image Cash Letter Split service
Application requirements	No
Initiates business processes?	None
Invocation	Runs as part of a business process.
Returned status values	<ul style="list-style-type: none"> <li>• Success (this status value is returned if the service successfully joined the binary images into the document)</li> <li>• Error (see below for list of errors) <ol style="list-style-type: none"> <li>1. Not licensed for use</li> <li>2. No Primary Document or zero bytes</li> <li>3. No Placeholder records found</li> <li>4. No Type52 records found</li> <li>5. Missing Type52 record for image</li> <li>6. Missing Type52 record after Type50</li> <li>7. Invalid record length</li> <li>8. Image document not found</li> </ol> </li> </ul>
Restrictions	No

## Implementing the Image Cash Letter Join Service

To implement the Image Cash Letter Join service, you just need to use the Image Cash Letter Join service in a business process.

### Parameters Passed From Business Process to Service

The following table contains the parameters passed from the business process to the Image Cash Letter Join service:

Parameter	Description
join_bufferSize	This is the size of the buffer for reading data. If not specified, the default is 2048.
join_encoding	This specifies the encoding of the input PrimaryDocument. If not specified, the PrimaryDocument encoding is used (if set) or the system default is used.
join_inputRecLen	This indicates whether the input PrimaryDocument contains a record length for each record. If not specified, it defaults to true (indicating that the input contains the record length). <b>Note:</b> The record length is always written to the output PrimaryDocument.
join_inputRecSepNum	This indicates whether the input PrimaryDocument contains record separators. If not specified, the default is zero (no separators). This behavior can be overridden by changing the value of this parameter (for example, if the file uses CR/LF record separator, set this parameter to <b>split_inputRecSepNum=2</b> ).
join_outputRecSeps	This indicates whether each record written to the output PrimaryDocument will be followed by a record separator. If not specified, the default is none (no record separators). If you want each record written with a record separator, you must do so using two-byte hex characters (for example, <b>split_outputRecSeps='0D0A'</b> indicates the separator used is the carriage return/line feed).

Parameter	Description
join52	<p>This indicates whether the service looks for placeholder records or Type52 records for image replacement. If not specified, it defaults to false, which instructs the service to look for placeholder records previously inserted by the Image Cash Letter Split service. If you want the service to use Type52 records, you must set this parameter to true. Furthermore, the Type52 records must contain fields 1-17 only and this service will fill in field18 (length of image data) and field19 (the image data). The images to be joined using the "join52=true" option must be in the ProcessData tree as follows:&lt;ProcessData&gt; &lt;Images&gt; &lt;Image1/&gt; &lt;ImageN/&gt; &lt;/Images&gt;&lt;/ProcessData&gt;</p>
errorIfNothingToJoin	<p>This determines whether an error is returned if no placeholder or Type52 records are found in the PrimaryDocument. The default is true (return an error if no placeholder or Type52 records are found), unless otherwise specified.</p>



## Chapter 28. Image Cash Letter Split Service (Build 7002 or higher)

The Image Cash Letter Split service removes variable length binary images (Type52 records) from Image Cash Letter documents. This service is typically used before translation.

**Note:** This service does not enforce or validate the standard.

The following table provides an overview of the Image Cash Letter Split service:

System name	ImageCashLetterSplit
Graphical Process Modeler (GPM) categories	<ul style="list-style-type: none"> <li>• All Services</li> <li>• EDI</li> </ul>
Description	Removes variable length binary images (Type52 records) from Image Cash Letter documents. This service is typically used before translation. and it replaces the Type52 record with a fixed length placeholder record that begins with   <b>CheckImage</b> followed by a number of the image file created in ProcessData (for example,  CheckImage1,  CheckImage2, and so forth). This default behavior can be changed to strip Type52 records and the image data separately by setting the workflow parameter <b>split52=true</b> . In this case, the textual portion of the record is added as an attribute of the image document.
Preconfigured?	A configuration of this service is created when the product is installed.
Requires third party files?	No
Platform availability	All supported application platforms
Related services	Image Cash Letter Join service
Application requirements	No
Initiates business processes?	None
Invocation	Runs as part of a business process.
Returned status values	<ul style="list-style-type: none"> <li>• Success (this status value is returned if the service successfully splits the binary images from the document)</li> <li>• Error (see below for list of errors)               <ol style="list-style-type: none"> <li>1. Not licensed for use</li> <li>2. No Primary Document or zero bytes</li> <li>3. Unexpected record type</li> <li>4. Invalid record length</li> <li>5. No Type52 records found</li> </ol> </li> </ul>
Restrictions	No

## Implementing the Image Cash Letter Split Service

To implement the Image Cash Letter Split service, you just need to use the Image Cash Letter Split service in a business process.

### Parameters Passed From Business Process to Service

The following table contains the parameters passed from the business process to the Image Cash Letter Split service:

Parameter	Description
split_bufferSize	This is the size of the buffer for reading data. If not specified, the default is 2048.
split_encoding	This specifies the encoding of the input PrimaryDocument. If not specified, the PrimaryDocument encoding is used (if set) or the system default is used.
split_inputRecSepNum	This indicates whether the input PrimaryDocument contains record separators. If not specified, the default is zero (no separators). This behavior can be overridden by changing the value of this parameter (for example, if the file uses CR/LF record separator, set this parameter to <b>split_inputRecSepNum=2</b> ).
split_outputRecSeps	This indicates whether each record written to the output PrimaryDocument will be followed by a record separator. The default record separator used is hex 0x0A(line feed). If you want to specify a different record separator, you must do so using two-byte hex characters (for example, <b>split_outputRecSeps="0D0A"</b> indicates the separator used is carriage return/line feed).
split_outputRecLen	This indicates whether the record lengths will be written to the output PrimaryDocument for each record. If not specified, it defaults to true (the record lengths are written to the output PrimaryDocument for each record).
split52	This specifies whether the Type52 record is removed from the PrimaryDocument as one entity. If not specified, it defaults to false. If you want the Type 52 record and image data removed separately, you must set this parameter to true (which will add the textual portion of the record as an attribute of the image document).
errorIfNothingToSplit	This determines whether an error is returned if no Type52 records are found in the PrimaryDocument. The default is true (return an error if no Type52 records are found), unless otherwise specified.



---

## Chapter 29. Map Test Service

The Map Test service runs only in concert with the Map Editor to enable you to remotely test a compiled map (.txo file) from a client machine prior to checking the map in to the application server.

The following table provides an overview of the Map Test service:

System Name	MapTestService
Graphical Process Modeler (GPM) categories)	All Services
Description	This service accepts a message containing a compiled map and associated data, runs the data through translation, and returns a translation report and output data to the user.
Business usage	This service is used by the Map Test feature of the Map Editor to allow testing of a map from the client machine without having to check the map into the application. <b>Note:</b> Although you can run this service from the client, the actual map is executed on the application Server. We recommend testing maps only against Test of Development systems, not against the application production system.
Usage example	A User runs the Map Test feature to remotely test a compiled map (.txo) from a client machine prior to checking the map in to the application server, to ensure that the map translates data correctly and efficiently, prior to executing the map in a Production environment.
Preconfigured?	Yes
Requires third party files?	None
Platform availability	All supported application platforms.
Related services	SOAP service, HTTP Server adapter, Translation service
Application requirements	None
Initiates business processes?	No
Invocation	Used internally by the application; invoked by the Map Editor Map Test feature on the client.
Business process context considerations	None
Returned status values	SuccessError

Restrictions	If you have not configured your operating system to specify default programs to open .txt and .xml files, the test result files may not be automatically displayed. If this is the case, however, you can locate and open the files using the file/path location.
Persistence level	System default
Testing considerations	None

## How the Map Test Service Works

When you use the Map Test service, the Map Editor takes a compiled map (translation object) and a data file to run with the map, and loads both the translation object and the data file into an XML SOAP message. When the Map Test service runs, it is visible in the application current processes interface.

Using HTTP, the Map Test client posts the XML SOAP message to the Map Test service. The Map Test service submits the SOAP message to the SOAP service (inbound or outbound), and the SOAP service disassembles the message and returns the translation object and associated data back to the Map Test service.

**Note:** You need to enable or start the Map Test service; by default it is disabled.

The Map Test service then submits the translation object and the data to the Translation service, which runs translation using the supplied translation object and data, and returns the output data and a translation report to the Map Test service.

The Map Test service loads the translation report and output data into another XML SOAP message and sends it to the client using the HTTP Server adapter. Then the Map Editor disassembles the SOAP message and presents the user with the translation report (in XML format) and the output translation data. If there is no translation report, the Map Editor returns a file stating that no translation report is available.

## Implementing the Map Test Service

**Note:** You do not need to create a configuration of the Map Test service. However, since the default is for the service to be disabled, you do need to enable it to use the Map Test feature. That is, in the **translator.properties** file the **maptest.MaptestServiceEnabled** property is set to False by default. If you do not set the value to True using the Customer Override feature (explained below), the service will not accept any map test requests.

You can turn off the Map Test service to prevent users who have access to a trading partner's application system from attempting to use the Map Test feature to run translation on that trading partner's system. Turning off the Map Test service prevents the possible execution of JDBC maps which could access production data. The application supports the use of a customer override property file to override property settings in the property files. The customer override property file is not changed during installation of the application upgrades or patches. To prevent having your customized settings overwritten, you should use the customer override property file whenever possible rather than editing the application property files directly.

To enable the Map Test service, complete the following tasks:

**Note:** If the Map Test service is disabled by the **translator.properties** entry, an error message returned to the client and is presented to the user in a format like the translator report.

1. In the **install\_dir/properties** directory, locate (or create, if necessary) the **customer\_overrides.properties** file.
2. Open the **customer\_overrides.properties** file in a text editor.
3. Add the property you want to override, using the following format:  
`translator.maptest.MaptestServiceEnabled=true`
4. Save and close the **customer\_overrides.properties** file.
5. Stop and restart the application to use the new values.
6. Test your changes to ensure that the overrides give the desired results. If you have problems, contact Sterling Commerce Customer Support for assistance.



---

## Chapter 30. RosettaNet Message Builder Service

**CAUTION:**

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the RosettaNet Message Builder service:

System name	RNIF20MessageBuilder
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > RosettaNet
Description	Constructs RNIF 1.1 and 2.0 message from documents and data stored in the business process context. Uses standard trading profile objects to encrypt or sign message if necessary.
Business usage	Required to build properly formatted RosettaNet RNIF 1.1 or 2.0 messages.
Usage example	Runs by RosettaNet PIP business processes.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	No
Initiates business processes?	No
Invocation	Runs by RosettaNet PIP business processes.
Business process context considerations	No
Returned status values	<ul style="list-style-type: none"><li>• Success – RosettaNet RNIF message built without error△</li><li>• △Error – RosettaNet RNIF message build error. Status report will contain error details.△</li></ul>
Restrictions	No
Testing considerations	Turn on the RosettaNet logger to see debug messages.



---

## Chapter 31. RosettaNet Message Parser Service

### CAUTION:

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the RosettaNet Message Parser service:

System name	RNIF20MessageParser
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > RosettaNet
Description	Parses an RNIF 1.1 or 2.0 message, storing the extracted headers in process data, and places service content and attachments as the body of the application documents. This service uses standard trading profile objects to decrypt or verify the message if necessary.
Business usage	Required to parse incoming RosettaNet RNIF 1.1 or 2.0 messages.
Usage example	Runs as part of a RosettaNet PIP business process.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a RosettaNet PIP business process.
Business process context considerations	No
Returned status values	<ul style="list-style-type: none"><li>• Success – RosettaNet message parsing was successful.△</li><li>• △Error△ – RosettaNet message parsing failed. The status report and the process data will contain the error details.△</li></ul>
Restrictions	No
Testing considerations	Turn on the RosettaNet logger to see debug messages.

### How the RosettaNet Message Parser Service Works

The RosettaNet Message Parser service parses an RNIF 1.1 or 2.0 message, storing the extracted headers in process data, and places service content and attachments as the body of the application documents. Uses standard trading profile objects to decrypt or verify signatures, if necessary. Operates on the primary document.

## Implementing the RosettaNet Message Parser Service

There is no implementation required for this service.

### Parameters Passed from Service to Business Process<sup>△</sup>

The following table contains the parameters passed from the RosettaNet Message Parser service to a business process:

Field	Description
ResponseType	Response type contained in message. Valid values are sync and async.
Preamble	XML containing the Preamble
DeliveryHeader	XML containing the Delivery Header (RNIF 2.0 only)
ServiceHeader	XML containing the Service Header
RNIFReceivedMIMEHeaders	XML containing the RNIF 1.1 or 2.0 MIME message headers
RNIFReceivedMessageInfo	Information about the received message (was it encrypted and/or signed?)
ParseResultCode	Was the parsing successful? Valid values are SUCCESS and ERROR.
ParseResultDescription	Description of the parsing result.



## Chapter 32. RosettaNet Message Sending Service

**CAUTION:**

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the RosettaNet Message Sending service:

System name	RNIF20Send
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > RosettaNet
Description	Sends RosettaNet RNIF 1.1 or 2.0 messages.
Business usage	Required to send RosettaNet RNIF 1.1 or 2.0 messages to a trading partner.
Usage example	Runs as part of a RosettaNet PIP business process.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms
Related services	RNHTTPAsyncSend business process used to perform an asynchronous HTTP send. <b>Note:</b> Because of our continuing efforts to improve services and adapters to align with new technology and capabilities, the HTTP Send adapter used in this business process has entered the retirement process in the application and will be replaced with the HTTP Client Adapter with related services. For more information on the retirement process, see <i>Retiring and Removed Services and Adapters</i> .
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a RosettaNet PIP business process.
Business process context considerations	No
Returned status values	<ul style="list-style-type: none"> <li>• Success – RosettaNet message successfully sent<sup>△</sup></li> <li>• <sup>△</sup>Error – RosettaNet message delivery failed<sup>△</sup></li> </ul>
Restrictions	No
Testing considerations	Turn on the RosettaNet logger to see debug messages.



---

## Chapter 33. RosettaNet PIP Tracking Service

**CAUTION:**

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and its use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the RosettaNet PIP Tracking service:

System name	RNPIPTracking
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > RosettaNet
Description	Manages RosettaNet PIP tracking data
Business usage	Used by other RosettaNet business process to ensure that RosettaNet PIPs execute in the proper order and to add correlation data for RosettaNet messages.
Usage example	Runs as part of a RosettaNet PIP business process.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a RosettaNet PIP business process.
Business process context considerations	No
Returned status values	N/A
Restrictions	No
Testing considerations	N/A



---

## Chapter 34. RosettaNet Profile Service

### CAUTION:

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and its use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the RosettaNet Profile service:

System name	RNProfile
Graphical Process Modeler (GPM) categories	All Services, Internet B2B > RosettaNet
Description	Loads the standard trading profile data as well as RosettaNet Profile data from the configured application contracts. The data is retrieved from the database and serialized as XML. The XML is returned and can be mapped into process data. The RosettaNet Message Sending service locates the business process definition name of the HTTP Async transport business process, based on the production trading profile.
Business usage	Used by a RosettaNet PIP process to load the RosettaNet PIP contract profile information.
Usage example	Runs as part of a RosettaNet PIP business process.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	No
Initiates business processes?	No
Invocation	Runs as part of a RosettaNet PIP business process.
Business process context considerations	No
Returned status values	<ul style="list-style-type: none"><li>• Success△ – RosettaNet PIP contract lookup succeeded△</li><li>• △Error△ – RosettaNet PIP contract lookup failed, the contract could not be found△</li></ul>
Restrictions	No
Testing considerations	Turn on the RosettaNet logger to see debug messages.



## Chapter 35. Standards Translation Service

The following table provides an overview of the Standards Translation service:

System name	Standards Translation Type
Graphical Process Modeler (GPM) categories	All Services, Translation
Description	Performs translation of the primary document using a specified map, and replaces the primary document with the result of the translation. This service sets correlations to enable EDI tracking and document repair for the standards used with the application.
Business usage	Performs translation of the primary document within business processes.
Usage example	You want to take positional data from your order system and translate it to variable-length-delimited data so that it can be read by your billing system. Use the Map Editor to create a map that will translate the incoming data from positional data to variable-length-delimited data. Write a business process that will put the data into the primary document, then start the Standards Translation service. Using the map you created, the service translates the data from positional data to variable-length-delimited, and replaces the old data with newly translated data in the primary document.
Preconfigured?	There is a configuration of the service delivered with the application, but you must configure parameters for it in the GPM.
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	The map specified in the map_name parameter must be registered with the application and activated. If either of these conditions is not met then the translation will not be performed.
Initiates business processes?	No
Invocation	Runs as part of a business process.

Business process context considerations	<p>The Standards Translation service looks for the following parameters in the business process context. If the service finds them, it uses them during translations where either the input or output is EDI:</p> <ul style="list-style-type: none"> <li>• edi_output_tag_delimiter</li> <li>• edi_output_segment_delimiter</li> <li>• edi_output_element_delimiter</li> <li>• edi_output_sub_element_delimiter</li> <li>• edi_output_repeating_element_delimiter</li> <li>• edi_output_release_character</li> <li>• edi_output_decimal_separator</li> <li>• edi_input_tag_delimiter</li> <li>• edi_input_segment_delimiter</li> <li>• edi_input_element_delimiter</li> <li>• edi_input_sub_element_delimiter</li> <li>• edi_input_repeating_element_delimiter</li> <li>• edi_input_release_character</li> <li>• edi_input_decimal_separator</li> </ul>
Returned status values	<ul style="list-style-type: none"> <li>• Success – Translation was successful.</li> <li>• Error – Errors were encountered during translation or translation could not be performed. See the Translator report contained in the Business Process Context Status report for further detail.</li> </ul>
Restrictions	No
Persistence level	None
Testing considerations	<p>The best way to test is within a simple business process where the Standards Translation service is the only operation. After the business process runs, verify the output in the application, and review the translator report for detail on what occurred during the translation.</p>

## How the Standards Translation Service Works

The Standards Translation service translates data in the following file formats and sets correlations to enable EDI tracking and document repair:

- Electronic data interchange (EDI)
- Positional
- Variable-length-delimited
- Extensible Markup Language (XML)
- Structured Query Language (SQL)
- Japanese Center for Informatization of Industry (CII)

**Note:** If the input document character encoding is specified in the application, it overrides the encoding specified in the map. The output document content type and character encoding are set based on the information contained in the map.



The Standards Translation service creates a translation report.

## Implementing the Standards Translation Service

To implement the Standards Translation service, complete the following tasks:

1. Activate your license for the Standards Translation service. See *An Overview of Implementing Services*.
2. If you are using a map that has a database on the output side, you must set up a connection to the database that contains the tables you want to access. See *Setting Up a Connection to an External Database*.
3. Create a Standards Translation service configuration. See *Creating a Service Configuration*.
4. Configure the Standards Translation service. See *Configuring the Standards Translation Service*.
5. Use the Standards Translation service in a business process.

## Configuring the Standards Translation Service

To configure the Standards Translation service, you must specify settings for the following fields in the GPM:

Field	Description
Config	Name of the service configuration.
edi_input_decimal_separator	Character used to indicate the decimal point on the input side.
edi_input_element_delimiter	Character used to delimit elements (fields) on the input side.
edi_input_release_character	Character used to quote elements (fields) that contain the delimiter on the input side.
edi_input_repeating_element_delimiter	Character used to delimit repeating elements on the input side.
edi_input_segment_delimiter	Character used to delimit segments on the input side.
edi_input_sub_element_delimiter	Character used to delimit sub-elements on the input side.
edi_input_tag_delimiter	Character used to delimit tags on the input side.
edi_output_decimal_separator	Character used to indicate the decimal point on the output side.
edi_output_element_delimiter	Character used to delimit elements (fields) on the output side.
edi_output_release_character	Character used to quote elements (fields) that contain the delimiter on the output side.
edi_output_repeating_element_delimiter	Character used to delimit repeating elements on the output side.
edi_output_segment_delimiter	Character used to delimit segments on the output side.
edi_output_sub_element_delimiter	Character used to delimit sub-elements on the output side.
edi_output_tag_delimiter	Character used to delimit tags on the output side.

Field	Description
exhaust_input	<p>Whether to execute the map until the Standards Translation service has translated all of the input. Valid values are Yes and No.</p> <p><b>Note:</b> If your map design is faulty (that is, if the data structure does not match the layout of the map), the data in the input file cannot be properly processed. If a segment is present in the input file it must be defined and active in the map and in the proper sequence. When the translator reads a segment, it tries to match it to the records in the map based on their tag values. If exhaust_input is set to "Yes" the translator attempts to match each segment in the input file to a segment in the map, until it reaches the end of the input file. Conversely, if exhaust_input is set to "No," the translator does not re-invoke the map to continue processing the remaining data in the input file.</p>
map_name	<p>Name of the map used for translation. The map must already be checked in to the application and enabled.</p> <p><b>Note:</b> The Standards Translation service must be configured so the map_name parameter does not override what is specified in process data when performing document repair. See <i>Business Process Example for Resending a Document</i> for more information. Additionally, you must specify an existing map that is already checked in to the application, or the Standards Translation service generates a null pointer exception when the map it attempts to load doesn't exist.</p>
output_report_to_process_data	<p>Whether to output the report to process data. Valid values are:</p> <ul style="list-style-type: none"> <li>• Yes: Output the report to process data.</li> <li>• No: Do no output the report to process data.</li> </ul>
output_to_process_data	<p>Whether the output of the translation should be placed in the process data tree. The output must be XML. Valid values are Yes and No.</p>
useStreams	<p>Whether to support large files (streaming mode). Valid values are Yes (default), No, and blank (which uses default). The default was changed with release 4.1.1, patch 1973. In versions previous to that, the service did not use document streaming by default.</p>

Field	Description
validate_input	<p>Validates the input to the input side of the map. Valid values are Yes and No.</p> <p><b>Note:</b> During translation, two types of validation are performed:</p> <ul style="list-style-type: none"> <li>• <b>Semantic validation</b> - controlled by the validate_input parameter. This type of validation checks the data against the definition of a selected industry standard (if applicable), field definitions in the map, and definitions in the envelope. The different types of semantic validation vary based on the specification of any given industry standard that is selected.</li> <li>• <b>Structural validation</b> - always performed, regardless of the validate_input parameter. It is a very basic check that ensures a function can be performed against the data. It checks whether a required field is blank or whether there is a mismatch between the expected data and what is actually in the field, for example.</li> </ul>
validate_input_against_dtd	<p>Validates the input to the DTD specified in the input document. Valid values are No validation, Validate using a DTD, and Validate using an XML schema.</p>
validate_output	<p>Validates the output to the output side of the map. Valid values are Yes and No.</p>

## Parameters Passed Through the Business Process Only

The following parameters can be passed through the business process using an Assign statement. Note that these parameters are not available through the GPM.

Parameter	Description
FromSchema	<p>Used to enable manipulation of a database schema prefix within the SQL Table/View or SQL Statement of a map. This parameter is required when overriding schema names within one or more SQL Statement fields. If the FromSchema and ToSchema parameters are not supplied, then no schema name substitution is performed.</p> <p><b>Note:</b> The schema search/replace is case-sensitive.</p>

Parameter	Description
ToSchema	<p>Used to enable manipulation of a database schema prefix within the SQL Table/View or SQL Statement of a map.</p> <p><b>Note:</b> The schema search/replace is case-sensitive. If the FromSchema and ToSchema parameters are not supplied, then no schema name substitution is performed. If the ToSchema parameter is supplied and contains a non-empty value, then any matching schema names are changed at translation time to use the supplied ToSchema schema value as follows:</p> <ul style="list-style-type: none"> <li>For a SQL Statement, only schema names that match the FromSchema value will be substituted. The FromSchema parameter is required—otherwise, no schema values are substituted. To match and substitute more than one value pair, the FromSchema and ToSchema parameter strings can be delimited with an @ sign. For example:  FromSchema="from1@from2"  ToSchema="to1@to2"  In this example, any schema names matching "from1" are changed to "to1," and any schema names matching "from2" are changed to "to2."</li> </ul> <p>For convenience, you can supply fewer ToSchema fragments than FromSchema fragments, and when there is no corresponding ToSchema fragment, the last fragment in the ToSchema string is used. For example:  FromSchema="from1@from2@from3"  ToSchema="to" In this example, any schema names matching "from1," "from2," or "from3" will be changed to "to."</p> <ul style="list-style-type: none"> <li>For a SQL Table/View, the FromSchema parameter is optional. If it is not supplied, all schema names are changed to the supplied ToSchema value. If it is supplied, the substitution occurs in the same way as it does for a SQL Statement. If the translator property <code>sql.driver.useIdentifierQuoteString</code> is set to True within <code>customer_overrides.properties</code>, then matching and substitution occurs with quoted schema names.</li> <li>If the ToSchema parameter is supplied but is empty (equal to "" (two double quotation marks) or " (two single quotation marks)), then any matching schema names contained in the map are removed at translation time.</li> </ul>

## Business Process Example for Resending a Document

When you will be repairing/resending a document, you must configure the Standards Translation service so the **map\_name** parameter does not override what is specified in process data when performing document repair. Therefore, you should configure your Standards Translation service business process to look like the following for this scenario:

```
...
<output message="Xout">
    <assign to="map_name">PosToTransactionReportRequest</assign>
    <assign to="." from="*"></assign>
</output>
...
```

The resend functionality will then add a `map_name` parameter to process data, which will reference the pass-through map. This map name must be picked up by the Standards Translation service so it is important to amend your BPML so process data can override the parameter that is defined for the service.



---

## Chapter 36. SWIFTNet Client Service

The SWIFTNet Client service is responsible for sending SWIFT InterAct or FileAct messages (both requests and responses) to SWIFTNet, which are initiated by the application. The SWIFTNet Client service enables you to use InterAct or FileAct messaging with a Store and Forward option. Additionally, the SWIFTNet Client service enables you to use either synchronous or asynchronous messaging for InterAct and either put or get messaging for FileAct.

**Note:** Each instance of the SWIFTNet Client service is configured for a pair of requestor/responder DNs and the SWIFTNet Client service name. This service may also be used by a third party to send authorization and refusal messages.

The following table provides an overview of the SWIFTNet Client service:

System Name	SWIFTNetClientService
Graphical Process Modeler (GPM) categories)	All Services.
Description	This service is responsible for sending SWIFT InterAct and FileAct messages (both requests and responses) to SWIFTNet, which are initiated by the application. The SWIFTNet Client service is also executed during system processing to create the SWIFTNet message header based on the configuration set in the CHIPS adapter. The request type is either chips.payment (if the transaction code is 10) or chips.message (for all transaction codes except 10). You do not need to specifically configure the SWIFTNet Client service for use with CHIPS.
Business usage	Use this service to send financial information based on SWIFT InterAct and FileAct messages to another participant in the SWIFTNet Central network. The business value of this service is inherent in utilizing the benefits of the SWIFTNet Central network to exchange financial messages.
Usage example	You wish to cancel a Customer Credit Transfer (MT192). The user uses the service to send out the Cancel Request, and wait for confirmation (MT 196). In this case, the request is sent out synchronously, with the service remaining open until the response is received.
Preconfigured?	Yes.
Requires third party files?	No third party files are required.
Platform availability	All supported application platforms.

Related services	This service works with the SWIFTNet Server adapter, the SWIFTNet HTTP Server adapter, the SWIFTNet MEFG Server, and the Command Line Adapter 2.
Application requirements	The SWIFTNet MEFG Server must be installed and configured in order to use this service. SSL can be implemented between the application and the SWIFTNet MEFG Server. You must also configure the SWIFTNet HTTP Server adapter.
Initiates business processes?	No.
Invocation	A user who has permission to perform this activity must execute the business process that invokes this service.
Business process context considerations	None
Returned status values	<ul style="list-style-type: none"> <li>• Fatal—non-recoverable error</li> <li>• Transient—recoverable error</li> <li>• Logic—recoverable error</li> <li>• Success—Success</li> <li>• Warning—Success with warning</li> </ul>
Restrictions	Only one SWIFTNet MEFG Server can be configured to talk to one SWIFTNet Server Adapter instance in the application.
Persistence level	N/A
Testing considerations	To test this adapter, run the SWIFTNet Client business process and verify that it completes successfully. Debug information for this service is located at:Operations > System > Logs > SWIFTNet

## How the SWIFTNet Client Service Works

The SWIFTNet Client service prepares the request and sends it to the SWIFTNet MEFG Server. The client application on the SWIFTNet MEFG Server processes this request, performs the necessary communication exchange with the SWIFTNet SAG/SNL instance, and sends the request to the SWIFTNet Central network. The SWIFTNet Client service can operate in either synchronous or asynchronous mode for InterAct. In synchronous mode, the request is sent to the SWIFTNet Central network using the SwInt:Exchange primitive. In asynchronous mode, the request is sent to the SWIFTNet Central network using the SwInt:Send primitive.

In synchronous mode, the SWIFTNet MEFG Server client application is blocked until a response is received from the responder through the SAG/SNL instance. Once a response is received, it is sent back to the application by the client application on the SWIFTNet MEFG Server, and the response payload is placed in the primary document.

In asynchronous mode, the SWIFTNet MEFG Server client application receives a response handle from the SAG/SNL instance. Using this response handle, the SWIFTNet MEFG Server client application periodically checks with SWIFTNet (using SwInt:Wait primitive) to determine if a response is available. Once a response is received, the response payload is placed in the primary document.



With release SWIFT 6.1, the use of a messaging interface that uses input channels is optional but is recommended. SWIFT plans to make it mandatory in a future release. Currently, the use of input channels is available for SWIFTNet InterAct Store and Forward (SnF) only.

If you configure the Input Channel in the SWIFTNet Server adapter, the SWIFTNet MEFG Server opens the Input Channel automatically during startup (when the SWIFTNet Server adapter is enabled). This Input Channel remains open until the SWIFTNet MEFG Server is shut down (when the SWIFTNet Server adapter is disabled). During this time, you have the option to send message using the input channel or without the input channel if you indicate this using the Input Channel parameter in the SWIFTNet Client service.

When SWIFTNet Copy is used, the Copied Request is sent to a third party in one of two modes: T-Copy (third party copy is for information only) or Y-Copy (third party needs to do authorization). When SWIFTNet Copy is used, the message/file copy is queued for delivery to the configured third party as soon as the third party is ready to receive. The third party has to acknowledge the receipt of the copy like any other message or file delivered from a queue. When the mode is Y-copy, then the third party must authorize or refuse the message or file, which requires a system message to be sent to SWIFT. When authorizing the message or file, the authorization may contain information destined to the sender and information destined to the receiver of the message or file.

When using SWIFT 6.1, the application may act as a third party. If the Copy Mode is Y-Copy, the application sends an authorization message, which is like sending an Interact store-and-forward request. The SWIFTNet Client service is used, but you must set the you must set the **thirdPartyAuth** parameter to TRUE, and provide the authorization decision (either Authorised or Refused) for the **AuthDecision** parameter. Additionally, the third party may provides information destined to the sender or receiver of the message or file.

## Implementing the SWIFTNet Client Service

To implement the SWIFTNet Client service, complete the following tasks:

1. Create a configuration of the SWIFTNet Client service. See *Managing Services and Services*. For information about the fields specific to this service, see *Configuring the SWIFTNet Client Service*.

**Note:** For specific instructions on configuring an input channel, see *SWIFT Input Channel*.

**Note:** If you create a new configuration of the SWIFTNet Client service, you must also create a new business process or edit a copy of the appropriate predefined business process, SWIFTNetClient.bp or SWIFTNetClientFA.bp, to update it to use your service configuration. You do not need to create an instance of the SWIFTNet Client service for every Requestor or Responder DN; you can simply reuse the SWIFTNet Client service instance and pass the parameters that differ from the sample service through the business process.

2. Specify field settings for the service configuration in the application Admin Console and in the GPM as necessary. See *Configuring the SWIFTNet Client Service*.

**Note:** When you create the configuration, you will configure it differently depending on whether you are using InterAct or FileAct messaging. Either can be used with or without the store-and-forward option.

## Configuring the SWIFTNet Client Service

1. Select **Deployment > Services > Configuration**.
2. Search for SWIFTNet Client service or select it from the list and click **Go!**.
3. Click **Edit**.
4. Specify field settings in the Admin Console or Business Process (*Creating or Setting Up a Service Configuration in the Admin Console or Business Process*), or the GPM (*Setting Up the Service in the GPM*).

**Note:** Each instance of the SWIFTNet Client service is configured for a pair of requestor/responder DNs and the SWIFTNet Client service name.

5. On the Confirm page, verify that the **Enable Service for Business Processes** check box is selected.

## Creating or Setting Up a Service Configuration in the Admin Console or Business Process

Use the field definitions in the following table to create a new configuration of the SWIFTNet Client service, or to set up the configuration provided with the application. Some fields are available in both the Admin Console and in the GPM.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – Do not include the configuration in a service group at this time.</li> <li>• Create New Group – Enter a unique name for a new group, which will be created with this configuration. (You can then add other services to the group as well.)</li> <li>• Select Group – If service groups already exist for this service type, they are displayed in the list. Select a group from the list.</li> </ul> <p><b>Note:</b> See <i>Managing Services and Services</i>.</p>
SWIFTNet Interface	SWIFTNet message type. Valid values are <b>InterAct</b> or <b>FileAct</b> . Required.
Store and Forward	Indicates if the file transfer is done using the store-and-forward method. Valid values are True (use Store-and-Forward) and False (default—do not use Store-and-Forward). Required. BPML element value is <b>SnF</b> .

Field	Description
SWIFTNet Operation	<p>The SWIFTNet operation to send an InterAct or FileAct message. Possible values are:</p> <ul style="list-style-type: none"> <li>• Synchronous (default)—InterAct</li> <li>• Asynchronous—InterAct</li> <li>• Put—FileAct</li> <li>• Get—FileAct</li> </ul> <p>Required. When SWIFTNet Operation is FileAct, you must select either Put or Get. If you do not select an operation, the service uses Synchronous as the default value. BPML element value is <b>sync</b> (default) or <b>async for InterAct, or Put or Get for FileAct</b>.</p>
Requestor DN	<p>Distinguished name of the requestor. Required. BPML element value is <b>requestorDN</b>.</p> <p><b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation.</p>
Responder DN	<p>Distinguished name of the responder. Required. BPML element value is <b>responderDN</b>.</p> <p><b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation.</p>
Service Name	<p>Name of the service to which both SWIFT correspondents have subscribed. Required. BPML element value is <b>serviceName</b>.</p> <p><b>Note:</b> This must be a SWIFTNet service to which you are subscribed.</p>
Authoriser DN	<p>The distinguished name of the authorizing party. Optional.</p>
This service allows Third Party Copy	<p>Whether this service uses T-Copy or Y-Copy (check your service agreement with SWIFT). BPML element value is <b>thirdPartyCopy</b>. Valid values are TRUE or FALSE. This parameter is displayed only if you selected <b>File Act</b> and <b>True</b> for <b>Store and Forward</b> on SWIFTNet Client Service Interface page.</p> <p><b>Note:</b> If the Copy Mode is Y-Copy, the application sends an authorization message, which is like sending an Interact store-and-forward request. The SWIFTNet Client service is used, but you must set the <b>This service allows Third Party Copy</b> parameter to TRUE, and provide the authorization decision (either Authorised or Refused) for the AuthDecision parameter.</p>

Field	Description
Request for Third Party Copy	<p>Whether you are requesting third party copy. When the Copy feature is defined as Optional in the service agreement, you can choose whether you want the Third Party Copy to occur. BPML element value is <b>copyIndicator</b>. Valid values are TRUE or FALSE. Displayed only if you select True for <b>This service allows Third Party Copy</b>.</p> <p><b>Note:</b> This parameter is displayed only if you selected <b>True</b> for <b>This service allows Third Party Copy</b>.</p>
Request for Notification from Third Party	<p>In T-Copy mode, this setting is not applicable, the value should always be set to FALSE. In Y-Copy mode, when the <b>Authorisation Notification Indicator</b> feature is available and defined as Optional in the service agreement, you can choose whether you want to receive the Authorisation Notification messages. BPML element value is <b>authNotifIndicator</b>. Valid values are TRUE or FALSE. Displayed only if you selected <b>True</b> for <b>This service allows Third Party Copy</b>.</p> <p><b>Note:</b> This parameter is displayed only if you selected <b>True</b> for <b>This service allows Third Party Copy</b>.</p>
Request Type	<p>Request type supported by the message exchange. Optional for InterAct and required for FileAct in SWIFTNet 6.0. BPML element value is <b>requestType</b>.</p> <p><b>Note:</b> In SWIFTNet 6.0 FileAct the format convention is as follows:</p> <pre data-bbox="933 1186 1331 1239">&lt;business_area&gt;.&lt;type_of_syntax&gt;.&lt;detailed_syntax_and_format&gt;</pre> <p>This format starts with a four-character business area code, followed by a period (dot), followed by a three-character code that designates the type of syntax (which can be &lt;nnn&gt; , FIN, or xxx), followed by another period (dot), and then followed by a more detailed indication of syntax and format.</p>
Request Reference	<p>User reference of the request. Optional. BPML element value is <b>requestReference</b>.</p>
Non Repudiation Required	<p>Indicates whether non-repudiation is required. Possible values are True (when enabled this means that trading partners cannot deny that they sent a request) or False (default—when enabled this indicates that non-repudiation is not required). Optional. BPML element value is <b>nonRepudiation</b>.</p>

Field	Description
End-to-End Signature Required	Whether an end-to-end signature is required. Valid values are False (default) and True. Optional. <b>Note:</b> You can use an end-to-end signature regardless of whether you are using non-repudiation (for example, for SWIFT SCORE messages).
Number of Retries	Number of retries to connect to SAG. Default value is 3. Optional. BPML element value is <b>numOfRetries</b> .
Retry Delay (in seconds)	Time that will elapse before the next retry. Default value is 60 (seconds). Optional. BPML element value is <b>secInRetryDelay</b> .
Trace	Trace for logging purposes in the SWIFTNet MEFG Server. Valid values are True and False (default). Required. BPML element value is <b>trace</b> .
Use Signature List	Whether to use a signature list. This enables you to select your own signatures. If you do not use a signature list then normal Crypto is used. Valid values are False and True. Required. <b>Note:</b> This parameter is displayed only if you selected <b>True</b> for <b>End-to-End Signature Required</b> .
Return Signature List	Whether to return a signature list. Valid values are False and True. Required. If you want a signature list returned, the SWIFTNet MEFG Server receives the requestor's own signature in the response message. This returned signature will be extracted and saved as a separate message. This message is stored in the database and is made available for Correlation search. <b>Note:</b> This parameter is displayed only if you selected <b>True</b> for <b>End-to-End Signature Required</b> .
Use RND	Whether to use RND (digest reference values that terminate on "and RND"). Valid values are False (default) and True. Required. <b>Note:</b> This parameter is displayed only if you selected <b>True</b> for <b>End-to-End Signature Required</b> .

Field	Description
Delivery Notification (Del. Notifn)	Indicates that the sender asked the receiver to send a delivery notification. Possible values are True or False (default). Optional. BPML element value is <b>deliveryNotification</b> . <b>Note:</b> This parameter is only displayed when you select <b>True</b> for Store and Forward or are performing a FileAct Put. If you are performing a Put operation, you can request the responder to send you a delivery notification and specify a different Delivery Notification DN and Request Type of Delivery Notification, if desired. If you are performing a Get operation, the responder can request Delivery Notification from the requestor after receiving the file. That setting for delivery notification is configured through the SWIFTNet Server adapter.
Request Type of Delivery Notification	Used to request a specific delivery notification message from the remote receiving server application when it returns the delivery notification (when Delivery Notification is set to True). Optional. BPML element value is <b>requestTypeDelNotifn</b> . <b>Note:</b> This parameter is only displayed when you select <b>True</b> for Store and Forward or a FileAct Put.
Message Priority	Indicates priority handling in the queue for store-and-forward only. Valid values are Normal (default) and Urgent. Optional. BPML element value is <b>messagePriority</b> . <b>Note:</b> This value is used as a selection criterion when delivering messages from a queue, and in SWIFTNet FileAct to influence the pace of the FileAct flow.
Use Input Channel	Whether to use the input channel. Valid values are False (default) and True. Required. This parameter is displayed only if you selected <b>True</b> for <b>Store and Forward</b> and <b>InterAct</b> for <b>SWIFTNet interface</b> . <b>Note:</b> Used for InterAct store-and-forward only. Select <b>True</b> if you are using an input channel. If you configure this parameter, the SWIFTNet MEFG Server opens the Input Channel automatically during the startup (when the SWIFTNet Server Adapter is enabled). This Input Channel remains open until the SWIFTNet MEFG Server is shut down (or the SWIFTNet Server Adapter is disabled). During this time, you still have an option to send message using the input channel or without the input channel. All you need to do is to indicate this by using this parameter in SWIFTNet Client service.
MEFG SWIFTNet IP	The IP address for the SWIFTNet MEFG Server. Required.
MEFG SWIFTNet Port	The port for the SWIFTNet MEFG Server. Default is NULL. Optional.

Field	Description
Response Timeout	The timeout interval (in seconds) in which a response must be received or the message operation fails. Optional. Default is 60 seconds.
Use SSL	Whether to enable Secure Socket Layer (SSL) over HTTP communication between the application and the SWIFTNet MEFG Server. Valid values are None and Must. <b>Note:</b> Regardless of the value you select for <b>Use SSL</b> , you must also update the business processes associated with the SWIFTNet Client service. See <i>Upgrading the SWIFTNetClient Business Process to Use the Integrated SWIFTNet Client Service</i> for more information.
Cipher Strength	Indicates the strength of the cipher. Possible values are ALL (default), WEAK, and STRONG. Optional.
CA Certificate	The CA certificate of the SWIFTNet MEFG Server. <b>Note:</b> This is the public key certificate that must be configured to set up the outbound SSL channel. This page is only displayed if you set <b>Use SSL</b> to <b>Must</b> . <b>Note:</b> The SWIFTNet Client service Configuration page allows you to select the same CA Certificate for SSL processing a second time, and continues to allow additional selections of the same certificate in subsequent edits. If you have already selected a CA Certificate once for a configuration of the SWIFTNet Client service, do not select the same CA Certificate again, as this will result in an error when you execute the relevant business process.
Switch to SnF mode when real-time transmission failed	Whether to switch to store-and-forward mode when real-time transmission fails. Select True if you want to switch to Store and Forward mode when the real-time transmission (InterAct and FileAct Put) has failed. Valid values are True and False.

## Setting Up the Service in the GPM

Use the field definitions in the following table to set up the service configuration in the GPM:

Parameter	Description
deliveryNotification	Indicates that the sender asked the receiver to send a delivery notification. Possible values are TRUE or FALSE. Optional. <b>Note:</b> This parameter is only displayed when you select <b>True</b> for SnF or are performing a FileAct Put. If you are performing a Put operation, you can request the responder to send you a delivery notification and specify a different Delivery Notification DN and Request Type of Delivery Notification, if desired. If you are performing a Get operation, the responder can request Delivery Notification from the requestor after receiving the file. That setting for delivery notification is configured through the SWIFTNet Server adapter.
deliverynotification DN	Distinguished name of the Responder of the delivery notification. Optional.
deliverynotificationRT	Request type of the delivery notification. This is used for a FileAct Get. Required.
fileDesc	User description about the file transfer. Only for FileAct Put. Optional.
fileInfo	Specify whether the file will be compressed or not. Only for FileAct Put. Optional. In SWIFTNet 6.0 FileAct, the format convention is as follows:SwCompression=<value> Valid values are SwCompression=None (default) or SwCompression=ZIP. <b>Note:</b> If you specify to use compression, you must have compressed the file before sending it to the SWIFTNet Server adapter.
interfaceMode	SWIFTNet message type. Valid values are InterAct or FileAct. The default value is InterAct. Required.
logicalFilename	This name is communicated to the server application. By default, this name is the physical name without the file path. Optional. Only for FileAct.For a FileAct Put, this is the logical name of the file to be retrieved based on the <reception_dir>/<responder_dn>/<requestor_dn>.For a FileAct Get, this is the logical name of the file to send based on the <download_dir>/<responser_dn>/<requestor_dn>.
messagePriority	Indicates priority handling in the queue for store-and-forward only. Optional. <b>Note:</b> This value is used as a selection criterion when delivering messages from a queue, and in SWIFTNet FileAct to influence the pace of the FileAct flow.



Parameter	Description
nonRepudiation	Indicates whether non-repudiation is required. Possible values are TRUE (when enabled, trading partners cannot deny that they sent a request) or FALSE (default, indicating that non-repudiation is not required). Optional.
numOfRetries	Number of retries to connect to SAG. Default value is 3. Optional.
physicalFilename	Optional. Only for FileAct.For a FileAct Put, this is the full path and the physical name of the file to send.For a FileAct Get, this is the full path and the physical name of the file to save after the Get is completed.
possibleDuplicate	Indicates whether to include a trailer specifying that this message may be a duplicate.This is an optional component of the envelope that indicates that this message may already have been sent. For example, if the system crashes during the delivery of a message, another copy of the message could be sent, with this trailer included to indicate that it may be a duplicate.Possible values are TRUE and FALSE (default).Optional.
requestorDN	Distinguished name of the requestor. Required. <b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation.
requestReference	User reference of the request. Optional.
requestType	Request type supported by the message exchange. Optional for InterAct and required for FileAct in SWIFTNet 6.0. <b>Note:</b> In SWIFTNet 6.0 FileAct the format convention is as follows: <business_area>.<type_of_syntax>.<detailed_syntax_and_format>  This format starts with a four-character business area code, followed by a period (dot), followed by a three-character code that designates the type of syntax (which can be <nnn> , FIN, or xxx), followed by another period (dot), and then followed by a more detailed indication of syntax and format
requestTypeDelNotifn	Used to request a specific delivery notification message from the remote receiving server application when it returns the delivery notification (when Non Repudiation required and/or Delivery Notification are set to TRUE). Optional
responderDN	Distinguished name of the responder. Required. <b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation.

Parameter	Description
secInRetryDelay	Number of delays before the next retry. Default value is 60 (seconds). Optional.
serviceName	Name of the service to which both SWIFT correspondents have subscribed. Required. This must be a SWIFTNet service to which you have already subscribed.
SnF	Indicates if the file transfer is done using the store-and-forward method. Valid values are True (use Store-and-Forward) and False (default—do not use Store-and-Forward). Required.
swiftOp	The SWIFTNet operation to send an InterAct or FileAct message Possible values are: <ul style="list-style-type: none"> <li>• sync (default)—InterAct</li> <li>• async—InterAct</li> <li>• put—FileAct</li> <li>• get—FileAct</li> </ul> Required.
trace	Trace for logging purposes in the SWIFTNet MEFG Server. Possible values are 0 (no logging; this is the default) or 4 (logging is enabled). Optional.
transferDesc	User description about the transfer. Only for FileAct. Optional.
transferInfo	User information about the transfer. Only for FileAct. Optional.
switchToSnF	Indicates whether you want to switch to store-and-forward mode if a real-time transmission (InterAct or a FileAct Put) has failed. Possible values are True or False (default). Required.
SnFServiceName	The name of the store-and-forward service. Required when Switch to SnF mode when real-time transmission failed is set to True.
HTTPClientAdapter	HTTP Client Adapter instance that is used to communicate with the SWIFTNet MEFG Server. Optional. Default value is SWIFTNetHTTPClientAdapter.
MEFGServerHost	The IP address of the SWIFTNet MEFG Server. Required.
MEFGServerPort	The port of the SWIFTNet MEFG Server. Optional. Default value is 80.
MEFGServerResponse Timeout	Timeout period (in seconds) for the SWIFTNet MEFG Server to respond. Optional. Default value is 60.
UseSSL	Flag to indicate whether to secure communication between the application and the SWIFTNet MEFG Server with SSL. Possible values are TRUE or FALSE (default). Optional.

Parameter	Description
CipherStrength	The level of encryption to be applied on the data channel. Possible values are All (default), Weak, or Strong. Optional.
CACertId	The public key certificates for the SWIFTNet MEFG Server. Required if SSL is set to TRUE.
sign	Whether an end-to-end signature is required.
useSignatureList	Whether to use a signature list. This enables you to select your own signatures. If you do not use a signature list then normal Crypto is used.
returnSignatureList	Whether to return a signature list.
useRND	Whether to use RND (digest reference values that terminate on "and RND"). Valid values are False (default) and True.
useInputChannel	Whether to use the input channel.
messageID	The message identifier of the payload. This is used only when <b>Possible Duplicate</b> is set to <b>True</b> .
renewDN	The distinguished name of the user. This is used for the renewal of Security Context.
authoriserDN	The distinguished name of the authorizing party.
thirdPartyCopy	Flag to indicate whether this service should use T-Copy or Y-Copy. Only available for FileAct SnF. Valid values are TRUE or FALSE.
copyIndicator	When the Third Party Copy feature is defined as Optional in the service agreement, you can choose whether you want Third Party Copy to occur. Valid values are TRUE or FALSE.
authNotifIndicator	In T-Copy mode this setting is not applicable and the value should always be set to FALSE. In Y-Copy mode, when the <b>Authorisation Notification Indicator</b> feature is available and defined as Optional in the service agreement, you can choose whether you want to receive back the Authorisation Notification messages. Valid values are TRUE or FALSE.
HeaderInfo	Your Enhanced Header Info. Since Enhanced Header Info is usually an XML structure, you should specify it as CDATA.
thirdPartyAuth	Flag to indicate that the application is acting as Third Party who are going to send notification message (in Y-Copy mode). Valid values are TRUE or FALSE. This parameter should be used together with the required <b>AuthDecision</b> and <b>MessageName</b> parameters, and optional <b>ToSndrInfo</b> , <b>ToRcvrInfo</b> and <b>RefuseReason</b> parameters.

Parameter	Description
AuthDecision	Specify the third party decision here. Valid values are Authorised or Refused. Use this parameter with <b>thirdPartyAuth</b> parameter.
MessageName	Specify the name of the HeaderInfo message as inserted into the mailbox. The format is ThirdParty_[CopySnFRef]. When you use Mailbox Extract service to extract the HeaderInfo message from mailbox, by default the name is available in Process Data.
ToSndrInfo	If you are the third party and decide to Authorised a request or file notifications, this parameter enables you to specify "Third Party to Sender Information." The information can be any structure (plain text or XML). If you use XML structure, this parameter should be CDATA.
ToRcvrInfo	If you are the third party and decide to Authorised a request or file notifications, this parameter enables you to specify Third Party to Receiver Information. The information can be any structure (plain text or XML). If you use XML structure, this parameter should be CDATA.
RefuseReason	If you are the third party and you refuse a request or file notifications, you can specify Refusal Reason in this parameter. The information can be any structure (plain text or XML). If you use XML structure, this parameter should be CDATA.

## Business Process Example

To construct a message you need to perform the following tasks:

- Create a configuration of the SWIFTNet Client service.
- Edit the SWIFTNetClient business process (or create a new business process) in the following manner:
  - Match the name of the business process that you create or modify.
  - If necessary, modify the SWIFTNet MEFG Server IP and port to point to your installation of the SWIFTNet MEFG Server.
  - Configure the business process for the Requestor DN/Responder DN pair and the SWIFTNet service name.
  - Specify the request type and request reference for use in SWIFTNet.
  - If required, select non-repudiation and possible duplicate (which enables the resending of the file in case of an error in transmission) parameters.
  - Specify the number of retries to the SAG connection and the retry interval.
  - Enable Document Tracking for AFT Tracking.

**Note:** You do not need to create an instance of the SWIFTNet Client service for every requestor or responder DN; you can reuse the SWIFTNet Client

service instance and pass in the requestorDN, responderDN, and any other parameters that differ from the configuration of the sample service through the SWIFTNetClient business process.

This is the BPML for the example business process:

```
<operation>
  <participant
    name="SWIFTNetClientService"/>
  <output
    message="handleClientRequest">
    <assign
      to="." from="*" />
    <assign to="swiftOp">async</assign>
  </output>
  <input
    message="testing">
    <assign to="."
      from="*" />
  </input>
</operation>
```

This is the complete BPML to execute the SWIFTNet Client service:

**Note:** The **bold** lines indicate information that you need to modify to match the business process you are using.

```
<process name="SWIFTNetClient">
  <sequence name="SWIFTNetClientService">
    <operation name="set user token">
      <participant name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign to="USER_TOKEN">admin</assign>
        <assign to="." from="*"></assign>
      </output>
      <input message="inmsg">
        <assign to="." from="*"></assign>
      </input>
    </operation>

    <operation>
      <participant name="SWIFTNetClientService" />
      <output message="handleClientRequest">
        <assign to="." from="*"></assign>
        <assign to="interfaceMode">interact</assign>
        <assign to="swiftOp">sync</assign>
        <assign to="requestorDN">o=swiftbic,o=swift</assign>
        <assign to="responderDN">o=swiftbic,o=swift</assign>
        <assign to="serviceName">swift.generic.ia!x</assign>
```

```

    <assign to="SnF">FALSE</assign>
    <assign to="nonRepudiation">FALSE</assign>
    <assign to="possibleDuplicate">FALSE</assign>
    <assign to="deliveryNotification">FALSE</assign>
  </output>
  <input message="testing">
    <assign to="." from="*"></assign>
  </input>
</operation>

</sequence>
</process>

```

This is the complete BPML to execute the SWIFTNet Client service for FileAct for a Put:

```

<process name="SWIFTNet-FA-Put-NonSnF-DN">
  <sequence name="SWIFTNetClientService">
    <operation name="set user token">
      <participant name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign to="USER_TOKEN">admin</assign>
        <assign to="." from="*"></assign>
      </output>
      <input message="inmsg">
        <assign to="." from="*"></assign>
      </input>
    </operation>

    <operation>
      <participant name="SWIFTNetClientService"/>
      <output message="handleClientRequest">
        <assign to="." from="*"></assign>
        <assign to="physicalFilename">
/local/share/measle/swiftdata/payload.txt</assign>
        <assign to="logicalFilename">payload.txt</assign>
        <assign to="transferInfo">payload</assign>
        <assign to="transferDesc">payload</assign>
        <assign to="fileDesc">payload</assign>
        <assign to="interfaceMode">fileact</assign>
      </output>
    </operation>
  </sequence>
</process>

```

```

    <assign to="swiftOp">put</assign>
    <assign to="requestorDN">o=swiftbic,o=swift</assign>
    <assign to="responderDN">o=swiftbic,o=swift</assign>
    <assign to="serviceName">swift.generic.fa!x</assign>
    <assign to="requestType">Type.GIS.Server1</assign>
    <assign to="SnF">FALSE</assign>
    <assign to="nonRepudiation">FALSE</assign>
    <assign to="possibleDuplicate">FALSE</assign>
    <assign to="deliveryNotification">TRUE</assign>
  </output>
  <input message="testing">
    <assign to="." from="*"></assign>
  </input>
</operation>

</sequence>
</process>

```

This is the complete BPML to execute the SWIFTNet Client service for FileAct for a Get:

```

<process name="SWIFTNet-FA-Get-NonSnF-DN">
  <sequence name="SWIFTNetClientService">
    <operation name="set user token">
      <participant name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign to="USER_TOKEN">admin</assign>
        <assign to="." from="*"></assign>
      </output>
      <input message="inmsg">
        <assign to="." from="*"></assign>
      </input>
    </operation>

    <operation>
      <participant name="SWIFTNetClientService"/>
      <output message="handleClientRequest">
        <assign to="." from="*"></assign>
        <assign to="physicalFilename">
/local/share/measle/swiftdata/payload-receive.txt</assign>
        <assign to="logicalFilename">payload.txt</assign>

```

```

    <assign to="interfaceMode">fileact</assign>
    <assign to="swiftOp">get</assign>
    <assign to="requestorDN">o=swiftbic,o=swift</assign>
    <assign to="responderDN">o=swiftbic,o=swift</assign>
    <assign to="serviceName">swift.generic.falx</assign>
    <assign to="SnF">FALSE</assign>
    <assign to="nonRepudiation">FALSE</assign>
    <assign to="possibleDuplicate">FALSE</assign>
    <assign to="deliveryNotification">TRUE</assign>
  </output>
  <input message="testing">
    <assign to="." from="*"></assign>
  </input>
</operation>

</sequence>
</process>

```

This is a sample business process for a third party to send an authorised notification message:

```

<process name="SWIFTNetClient">
  <sequence name="SWIFTNetClientService">
    <operation name="set user token">
      <participant name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign to="USER_TOKEN">admin</assign>
        <assign to="." from="*" />
      </output>
      <input message="inmsg">
        <assign
to="." from="*" />
      </input>
    </operation>
  </operation>
  <participant name="SWIFTNetClientService"/>
  <output message="handleClientRequest">
    <assign to="." from="*" />
  </output>
  <assign to="thirdPartyAuth">TRUE</assign>
  <assign to="AuthDecision">Authorised</assign>

```



```

    <assign to="MessageName">ThirdParty_snp892349710118</assign>
    <assign to="ToSndrInfo">Plain text example</assign>
    <assign to="ToRcvrInfo"><![CDATA[<info><abc>XML
example</abc></info>]]></assign>
  </output>
  <input message="testing">

  <assign to="." from="*" />
</input>

</operation>
</sequence>
</process>

```

This is a sample business process for a third party Third Party to send a refused notification message.

```

<process name="SWIFTNetClient">
<sequence name="SWIFTNetClientService">
  <operation name="set user token">
    <participant name="SetUserToken"/>
    <output message="SetUserTokenMessage">
      <assign to="USER_TOKEN">admin</assign>
      <assign to="." from="*" />
    </output>
    <input message="inmsg">
      <assign
to="." from="*" />
    </input>
  </operation>
  <operation>
    <participant name="SWIFTNetClientService"/>
    <output message="handleClientRequest">
      <assign to="." from="*" />
    <assign to="thirdPartyAuth">TRUE</assign>
    <assign to="AuthDecision">Refused</assign>
    <assign to="MessageName">ThirdParty_snp892349710118</assign>
    <assign to="RefuseReason">Plain text example</assign>
  </output>
  <input message="testing">
    <assign to="." from="*" />
  </input>
</operation>
</sequence>
</process>

```

This is a sample business process to send a FileAct store-and-forward with Header Info:

```

<process name="SWIFTNetClient">
  <sequence name="SWIFTNetClientService">
    <operation name="set user token">
      <participant name="SetUserToken"/>
        <output message="SetUserTokenMessage">
          <assign to="USER_TOKEN">admin</assign>
          <assign to="." from="*" />
        </output>
        <input message="inmsg">
          <assign
to="." from="*" />
        </input>
      </operation>
    </operation>
    <participant name="SWIFTNetClientService"/>
    <output message="handleClientRequest">
      <assign to="." from="*" />
    </output>
    <assign to="interfaceMode">fileact</assign>
    <assign to="swiftOp">put</assign>
    <assign to="SnF">TRUE</assign>
    <assign to="requestorDN">o=swiftbic,o=swift</assign>
    <assign to="responderDN">o=swiftbic,o=swift</assign>
    <assign to="serviceName">swift.generic.falx</assign>
    <assign to="requestType">pain.001.001.01</assign>
    <assign to="physicalFilename">
/local/share/measle/swiftdata/payload.txt</assign>
    <assign to="logicalFilename">payload.txt</assign>
    <assign to="transferInfo">Date=29082008</assign>
    <assign to="transferDesc">transfer desc</assign>
    <assign to="fileInfo">SwCompression=None</assign>
    <assign to="fileDesc">file desc</assign>
    <assign to="HeaderInfo">
    <![CDATA[<App1Spcfc xmlns="urn:swift:xsd:App1Spcfc.TxsCtr.01">
<TxsCtr><Tt1NbOfTxs>5</Tt1NbOfTxs></TxsCtr></App1Spcfc]]></assign>
    <assign to="nonRepudiation">FALSE</assign>
    <assign to="possibleDuplicate">FALSE</assign>
  </sequence>
</process>

```

```

        <assign to="deliveryNotification">FALSE</assign>

    </output>
    <input message="testing">

        <assign to="." from="*" />
    </input>

</operation>
</sequence>
</process>

```

This is the complete business process to open the input channel:

```

<process
name="SWIFTNetOpenInputChannel">
  <sequence
name="SWIFTNetOpenInputChannel">
  <operation
name="set user token">
  <participant
name="SetUserToken"/>
  <output message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*" />
  </output>
  <input
message="inmsg">
  <assign to="." from="*" />
  </input>
  </operation>

  <!-- build Open request -->

  <operation name="Service">
  <participant
name="SWIFTNetClientService"/>
  <output
message="openInputChannelRequest">
  <assign
to="." from="*" />
  </output>
  <input
message="inmsg">
  <assign to="." from="*" />
  </input>

  </operation>
  </sequence>
</process>

```

This is the complete business process to close the input channel:

```

<process
name="SWIFTNetCloseInputChannel">
  <sequence
name="SWIFTNetCloseInputChannel">
  <operation
name="set user token">
  <participant
name="SetUserToken"/>
  <output message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*" />

```

```

        </output>
        <input
message="inmsg">
        <assign to="." from="*" />
        </input>
    </operation>
    <!--
build Close request -->
    <operation name="Service">
        <participant
name="SWIFTNetClientService"/>
        <output
message="closeInputChannelRequest">
        <assign
to="." from="*" />
        </output>
        <input
message="inmsg">
        <assign to="." from="*" />
        </input>
    </operation>
</sequence>
</process>

```

This is the complete business process to create the input channel:

```

<process
name="SWIFTNetCreateInputChannel">
    <sequence
name="SWIFTNetCreateInputChannel">
        <operation
name="set user token">
            <participant
name="SetUserToken"/>
            <output message="SetUserTokenMessage">
                <assign
to="USER_TOKEN">admin</assign>
                <assign
to="." from="*"></assign>
            </output>
            <input
message="inmsg">
                <assign to="." from="*"></assign>
            </input>
        </operation>
    <!--
build Create request -->
        <operation>
            <participant
name="SWIFTNetClientService"/>
            <output
message="createInputChannelRequest">
                <assign
to="." from="*"></assign>
                <assign
to="authoriserDN">Put a value here</assign>
                <assign
to="inputChannelName">Put a value here</assign>
            </output>
            <input
message="inmsg">
                <assign to="." from="*"></assign>
            </input>
        </operation>
    </sequence>
</process>

```

This is the complete business process to delete the input channel:

```
<process
name="SWIFTNetDeleteInputChannel">
  <sequence
name="SWIFTNetDeleteInputChannel">
  <operation
name="set user token">
  <participant
name="SetUserToken"/>
  <output message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*"></assign>
  </output>
  <input
message="inmsg">
  <assign to="." from="*"></assign>
  </input>
  </operation>
  <!--
build Delete request -->
  <!-- W A R
N I N G   N O T E -->
  <!-- Once deleted,
the input channel cannot be re-created or used anymore -->
  <operation>
  <participant
name="SWIFTNetClientService"/>
  <output
message="deleteInputChannelRequest">
  <assign
to="." from="*"></assign>
  <assign
to="authoriserDN">Put a value here</assign>
  <assign
to="inputChannelName">Put a value here</assign>
  </output>
  <input
message="inmsg">
  <assign to="." from="*"></assign>
  </input>
  </operation>
  </sequence>
</process>
```

This is the complete business process to renew the Security Context:

```
<process
name="SWIFTNetClientRenewSecContext">
  <sequence
name="SWIFTNetClientService">
  <operation
name="set user token">
  <participant
name="SetUserToken"/>
  <output message="SetUserTokenMessage">
  <assign
to="USER_TOKEN">admin</assign>
  <assign
to="." from="*" />
  </output>
  <input
message="inmsg">
  <assign to="." from="*" />
  </input>
  </operation>
```

```

<!--
build SWIFTNET request -->
  <operation>
    <participant
name="SWIFTNetClientService"/>
    <output
message="renewSecurityContext">
      <assign
to="renewDN">o=swiftbic,o=swift
</assign>
      <assign
to="." from="*"/>
    </output>
    <input
message="testing">
      <assign to="." from="*"/>
    </input>
    </operation>
  </sequence>
</process>

```

## Parameters Passed From Business Process to Service

The following table contains the parameters passed from the business process to the SWIFTNet Client service:

Parameter	Description
swiftOp	The SWIFTNet operation to send an InterAct or FileAct message Possible values are: <ul style="list-style-type: none"> <li>• sync (default)—InterAct</li> <li>• async—InterAct</li> <li>• put—FileAct</li> <li>• get—FileAct</li> </ul> Required.
trace	Trace for logging purposes in the SWIFTNet MEFG Server. Possible values are 0 (no logging; this is the default) or 4 (logging is enabled). Optional.
numOfRetries	Number of retries to connect to SAG. Default value is 3. Optional.
secInRetryDelay	Time that will elapse before the next retry. Default value is 60 (seconds). Optional.
requestorDN	Distinguished name of the requestor. Required. <b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation.
responderDN	Distinguished name of the responder. Required. <b>Note:</b> This DN must be registered with the SAG instance using SWIFTNet Alliance Webstation.
serviceName	Name of the service to which both SWIFT correspondents have subscribed. Required. <b>Note:</b> This must be SWIFTNet service to which you are subscribed.

Parameter	Description
requestType	Request type supported by the message exchange. Optional.
requestReference	User reference of the request. Optional.
SnF	Indicates if the file transfer is done using the store-and-forward method. Valid values are True (use Store-and-Forward) and False (default—do not use Store-and-Forward). Required.
physicalFilename	Optional. Only for FileAct. For a FileAct Put, this is the full path and the physical name of the file to send. For a FileAct Get, this is the full path and the physical name of the file to save after the Get is completed.
logicalFilename	This name is communicated to the server application. By default, this name is the physical name without the file path. Optional. Only for FileAct. For a FileAct Put, this is the logical name of the file to be saved based on the <reception_dir>/<responder_dn>/<requestor_dn>. For a FileAct Get, this is the logical name of the file to send based on the <download_dir>/<responder_dn>/<requestor_dn>.
fileInfo	User information about the file transfer. Only for FileAct. Optional.
fileDesc	User description about the file transfer. Only for FileAct. Optional.
transferInfo	User information about the transfer. Only for FileAct. Optional.
transferDesc	User description about the transfer. Only for FileAct. Optional.
possibleDuplicate	Indicates whether to include a trailer specifying that this message may be a duplicate. This is an optional component of the envelope that indicates that this message may already have been sent. For example, if the system crashes during the delivery of a message, another copy of the message could be sent, with this trailer included to indicate that it may be a duplicate. Possible values are TRUE and FALSE (default). Optional.
messageID	Message identifier for resending a message if Possible Duplicate is set to TRUE. Optional.

Parameter	Description
deliveryNotification	Indicates that the sender asked the receiver to send a delivery notification. Possible values are TRUE or FALSE. Optional. <b>Note:</b> This parameter is only displayed when you select <b>True</b> for SnF or are performing a FileAct Put. If you are performing a Put operation, you can request the responder to send you a delivery notification and specify a different Delivery Notification DN and Request Type of Delivery Notification, if desired. If you are performing a Get operation, the responder can request Delivery Notification from the requestor after receiving the file. That setting for delivery notification is configured through the SWIFTNet Server adapter.
requestTypeDelNotifn	Used to request a specific delivery notification message from the remote receiving server application when it returns the delivery notification (when Non Repudiation required and/or Delivery Notification are set to TRUE). Optional
messagePriority	Indicates priority handling in the queue for store-and-forward only. Optional. <b>Note:</b> This value is used as a selection criterion when delivering messages from a queue, and in SWIFTNet FileAct to influence the pace of the FileAct flow.
nonRepudiation	Indicates whether non-repudiation is required. Possible values are TRUE (when enabled, trading partners cannot deny that they sent a request) or FALSE (default, indicating that non-repudiation is not required). Optional.
HeaderInfo	The Enhanced Header information. Since Enhanced Header Info is usually an XML structure, you should specify it as CDATA.
thirdPartyAuth	Flag to indicate that the application is acting as a Third Party that will send a notification message (in Y-Copy mode). Valid values are TRUE or FALSE. This parameter should be used together with the required AuthDecision and MessageName parameters, and optional ToSndrInfo, ToRcvrInfo and RefuseReason parameters.
AuthDecision	The third party decision. Valid values are Authorised or Refused. Use this parameter with the thirdPartyAuth parameter.
MessageName	The name of the HeaderInfo message, as inserted into the mailbox. The format is ThirdParty_[CopySnFRef]. When you use Mailbox Extract service to extract the HeaderInfo message from mailbox, by default the name is available in Process Data.



Parameter	Description
ToSndrInfo	If you are the third party and decide to authorize a request or file notification, this parameter enables you to specify Third Party to Sender Information. The information can be in any format (plain text or XML). If you use XML format, this parameter should be CDATA.
ToRcvrInfo	If you are the third party and decide to authorize a request or file notification, this parameter enables you to specify Third Party to Receiver Information. The information can be any format (plain text or XML). If you use XML format, this parameter should be CDATA.
RefuseReason	If you are the third party and you refuse a request or file notification, you can specify a Refusal Reason in this parameter. The information can be any format (plain text or XML). If you use XML format, this parameter should be CDATA.
sign	Whether an end-to-end signature is required.
useSignatureList	Whether to use a signature list. This enables you to select your own signatures. If you do not use a signature list then normal Crypto is used.
returnSignatureList	Whether to return a signature list.
useRND	Whether to use RND (digest reference values that terminate on "and RND"). Valid values are False (default) and True.
renewDN	The distinguished name of the user. This is used for the renewal of Security Context.
MEFGServerHost	The IP address of the SWIFTNet MEFG Server. Required.
MEFGServerPort	The port of the SWIFTNet MEFG Server. Optional. Default value is NULL.
MEFGServerResponse Timeout	Timeout period (in seconds) for the SWIFTNet MEFG Server to respond. Optional. Default value is 60.
UseSSL	Flag to indicate whether to secure communication between the application and the SWIFTNet MEFG Server with SSL. Possible values are TRUE or FALSE (default). Optional.
CACertId	The public key certificates for the SWIFTNet MEFG Server. Required if SSL is set to TRUE.
CipherStrength	The level of encryption to be applied on the data channel. Possible values are All (default), Weak, or Strong. Optional.
useInputChannel	Whether to use the input channel.
forceOpen	Whether to force the channel open.

Parameter	Description
switchToSnF	Whether to switch to store-and-forward mode when real-time transmission fails. Select True if you want to switch to Store and Forward mode when the real-time transmission (InterAct and FileAct Put) has failed. Valid values are True and False.
SnFServiceName	The name of the store-and-forward service. Required when Switch to SnF mode when real-time transmission failed is set to True.
HTTPClientAdapter	HTTP Client Adapter instance that is used to communicate with the SWIFTNet MEFG Server. Optional. Default value is SWIFTNetHTTPClientAdapter.

## Upgrading the SWIFTNetClient Business Process to Use the Integrated SWIFTNet Client Service

Now that the SWIFTNet Client service has been enhanced to support SSL, the SWIFTNet Client service has also been improved by integrating all the outbound services internally. To use the SWIFTNet Client service, you must upgrade the SWIFTNetClient business process. The upgraded BPML differs based on whether you are using InterAct or FileAct.

**Note:** If you previously installed an earlier version of the application Standards Library, you do not need to upgrade the SWIFTNetClient business process again. However, you will need to reinstall the SWIFTNet MEFG Server (see *Using SWIFTNet* for more information).

## Upgrading the SWIFTNetClient Business Process for InterAct

If you are using InterAct, this is the complete BPML to execute the SWIFTNet Client service for InterAct:

```
<process
name="SWIFTNetClient">
  <sequence name="SWIFTNetClientService">
    <operation
name="set user token">
      <participant
name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign
to="USER_TOKEN">admin</assign>
        <assign
to="." from="*"/>
      </output>
      <input
message="inmsg">
        <assign to="." from="*"/>
      </input>
    </operation>
    <!--
build SWIFTNET request -->
    <operation>
      <participant
name="SWIFTNetClientService"/>
      <output
message="handleClientRequest">
        <assign
```

```

to="." from="*" />
</output>
<input
message="testing">
<assign to="." from="*" />
</input>
</operation>
</sequence>
</process>

```

## Upgrading the SWIFTNetClient Business Process for FileAct

If you are using FileAct, this is the complete BPML to execute the SWIFTNet Client service for FileAct:

```

<process
name="SWIFTNetClientFA">
<sequence name="SWIFTNetClientService">
<operation
name="set user token">
<participant
name="SetUserToken"/>
<output message="SetUserTokenMessage">
<assign
to="USER_TOKEN">admin</assign>
<assign
to="." from="*" />
</output>
<input
message="inmsg">
<assign to="." from="*" />
</input>
</operation>
<!--
build SWIFTNET request -->
<operation>
<participant
name="SWIFTNetClientService"/>
<output
message="handleClientRequest">
<assign
to="." from="*" />
<assign to="physicalFilename"
from=""/>
<assign to="logicalFilename"
from=""/>
<assign to="transferInfo"
from=""/>
<assign to="transferDesc"
from=""/>
<assign to="fileInfo" from="'SwCompression=None'"/>
<assign
to="fileDesc" from=""/>
</output>
<input
message="testing">
<assign to="." from="*" />
</input>
</operation>
</sequence>
</process>

```

## Enabling SWIFTNet Document Tracking

When you are creating or editing your SWIFTNet Client business process in the business process text editor, you can easily enable SWIFTNet document tracking in

the application by selecting the **Document Tracking** check box on the Process Levels page. Set the following options as needed and leave the rest of the business process parameters as the defaults:

- On the **Deadline Settings** page, set the deadline and notification options, if necessary.
- On the **Life Span** page, set the life span, if necessary.

## Chapter 37. SWIFTNet Server Adapter (Build 5100 - 5103)

The SWIFTNet Server adapter communicates to the SWIFTNet Network through the SWIFTNet MEFG server. It responds to and accepts InterAct and FileAct messages that are sent by remote SWIFTNet correspondents. The following table provides an overview of the SWIFTNet Server adapter:

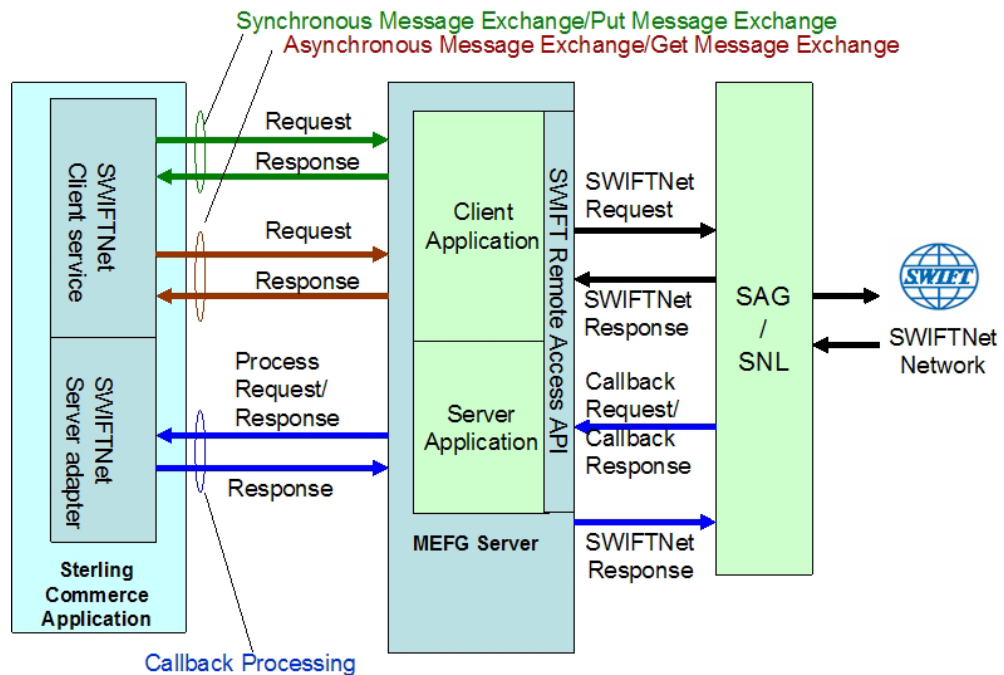
System Name	SWIFTNet Server Adapter
Graphical Process Modeler (GPM) categories)	All services
Description	This adapter is responsible for receiving and responding to SWIFTNet InterAct and FileAct messages using the SWIFTNet MEFG Server.
Business usage	A business would use this adapter in order to exchange SWIFTNet InterAct and FileAct messages with its trading partners over the SWIFTNet system. <b>Note:</b> You will also need to configure this adapter if you are transporting CHIPS messages using the SWIFTNet transport mode. This adapter also sends acknowledgements to CHIPS.
Usage example	When an InterAct or FileAct message is received, a business process is executed to process the message and, when required, to generate the SWIFTNet response.
Preconfigured?	This adapter is preconfigured as part of the application installation.
Requires third party files?	No third party files are required.
Platform availability	All supported application platforms.
Related services	This is designed to work in conjunction with the SWIFTNet MEFG Server and the Command Line Adapter 2. This service also works with the SWIFTNet HTTP Server adapter to provide SSL support.
Application requirements	SSL can be implemented between the application and the MEFG Server if the SWIFTNet HTTP Server adapter is configured for that setup.
Initiates business processes?	Initiates system business processes.
Invocation	By the Multi-Enterprise Financial Gateway for SWIFTNet application.
Business process context considerations	None.
Returned status values	<ul style="list-style-type: none"> <li>• Fatal—non-recoverable error</li> <li>• Transient—recoverable error</li> <li>• Logic—recoverable error</li> <li>• Success—Success</li> <li>• Warning—Success with warning</li> </ul>

Restrictions	Only one SWIFTNet MEFG Server can be configured to talk to one SWIFTNet Server adapter instance in the application.
Persistence level	N/A
Testing considerations	N/A

## How the SWIFTNet Server Adapter Works

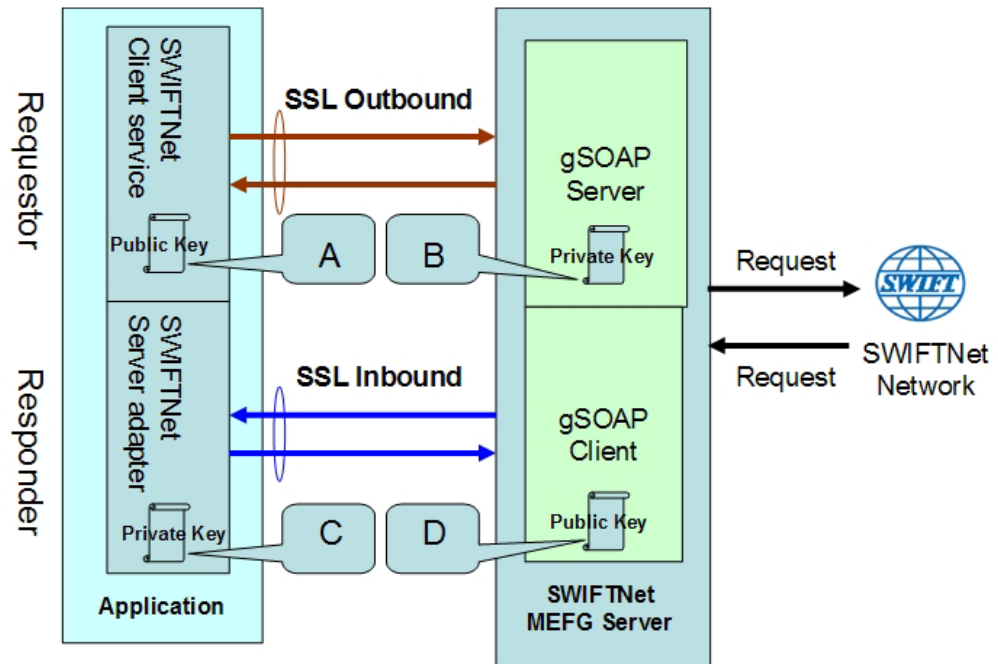
The SWIFTNet Server adapter is comprised of two parts: the service part and the adapter part. The service part is used in a business process that does not require configuration except for enabling it for document tracking. The adapter part is configured through the Admin Console or the GPM, and this adapter is responsible for starting and stopping the SWIFTNet MEFG Server from the application using the Command Line Adapter 2 (CLA2), which is built into the SWIFTNet Server adapter. Starting and stopping the operation of the SWIFTNet MEFG Server will only work correctly if the CLA2Client.jar is deployed in the same machine where the SWIFTNet MEFG Server is installed. The CLA2Client.jar file must also be started by a user who has permission to access the SWIFTNet MEFG Server home directory.

This diagram illustrates the process flow between the application and the SWIFTNet network through the SWIFTNet MEFG Server (not using SSL):



The SWIFTNet Server adapter (in conjunction with the SWIFTNet HTTP Server adapter) enables you to use Secure Sockets Layer (SSL) to provide secure authentication, using the SWIFTNet HTTP Server adapter to accept the forwarded request from the SWIFTNet MEFG Server. When you use SSL with the application, two channels are secured: an Outbound channel (the application acting as the Requestor) and an Inbound channel (the application acting as the Responder).

This diagram illustrates the configuration necessary between the application and the SWIFTNet network through the SWIFTNet MEFG Server to set up the Outbound and Inbound channels (using the SWIFTNet HTTP Server adapter for SSL):



You will need 2 pairs of certificates. The first pair belongs to the SWIFTNet MEFG Server (A and B in the diagram above) and is used to secure the outbound channel. The second pair of certificates belongs to the application (C and D in the diagram above) and is used to secure the inbound channel. In the above diagram, the callouts signify the following:

- A — A public key certificate file belongs to the SWIFTNet MEFG Server that is configured on the SWIFTNet Client service (the certificate is specified for the CA Certificate parameter).
- B — A private key certificate file that is stored on the SWIFTNet MEFG Server as a key file (which you configure through the SSL Configuration utility named `sslUtil.jar` in the SWIFTNet MEFG Server installation bin subdirectory). The `sslUtil.jar` file is located in the bin subdirectory of the SWIFTNet MEFG Server installation directory.
- C — A private key certificate file that is configured on the SWIFTNet HTTP Server adapter (the certificate is specified for the System Cert parameter).
- D — A public key file that belongs to the application and is stored for the SWIFTNet MEFG Server as a CA Cert file or trusted list (that you configure through the SSL Configuration utility named `sslUtil.jar` in the SWIFTNet MEFG Server installation directory).

**Note:** To configure SSL on the SWIFTNet MEFG Server, run the following command in the bin directory of the MEFG SWIFTNet Server installation bin sub-directory:

```
java -jar sslUtil.jar
```

**Note:** The application enables you easily renew certificates. See *Renewing a Certificate* for more information.

## Implementing the SWIFTNet Server Adapter

To implement the SWIFTNet Server adapter, complete the following tasks:

1. Create a configuration of the Command Line Adapter 2.
  - a. Locate the client jar (CLA2Client.jar) that contains the necessary classes.
  - b. Move the client jar to the machine where you will be running the remote adapter.
  - c. Start the remote adapter using the following command:

```
java -jar CLA2Client.jar <port> [debug]
```

**Note:** The [debug] option is not required, but is provided for your convenience. If you upgrade the application, you may need to obtain a new CLA2Client.jar file to avoid a Class Conflict error.
2. Create a configuration of the SWIFTNet Server adapter. See *Managing Adapters and Services*. For information about the fields specific to this adapter, see *Configuring the SWIFTNet Server Adapter*.
3. Specify field settings for the adapter configuration in the application Admin Console and in the GPM as necessary. See *Creating or Setting Up a Adapter Configuration in the Admin Console* or *Setting Up the Adapter in the GPM*.
4. Configure the business process you are using for the SWIFTNet Server adapter. The business processes that work with SWIFTNet Server adapter include the following:
  - handleSWIFTNetServerSnFRequest
  - handleSWIFTNetInboundCorrelation
  - handleSWIFTNetOutboundCorrelation
  - handleSWIFTNetServerFADelNotif
  - handleSWIFTNetServerFAEvent
  - handleSWIFTNetServerFARequest
  - handleSWIFTNetServerFASnFDelNotif
  - handleSWIFTNetServerFASnFRequest
  - handleSWIFTNetServerRequest
  - handleSWIFTNetServerSnFDelNotif
  - handleSWIFTNetServerFASnFEvent
  - handleSWIFTNetSnFinboundCorrelation
  - handleSWIFTNetSnFOutboundCorrelation
5. Define the **HTTP Listen Port** in the SWIFTNet HTTP Server adapter instance, which should have the same value as the **GIS HTTP Sever Adapter Port** defined in the SWIFTNet Server adapter configuration.
6. Specify field settings in the business process. See *Business Process Example*.

## Configuring the SWIFTNet Server Adapter

1. Select **Deployment > Services > Configuration**.
2. Search for SWIFTNet Server adapter or select it from the list and click **Go!**
3. Click **Edit**.
4. Specify field settings in the Admin Console (*Creating or Setting Up a Adapter Configuration in the Admin Console*)—alternatively you can specify field settings in the GPM (*Setting Up the Adapter in the GPM*), but you will need to access the adapter instance through the Admin console to enable the instance (as described in step 5).



**Note:** Specify failover processing to ensure that failover is supported if a SAG connection fails by configuring **Active-Active Configuration**.

**Note:** For specific instructions on configuring an input channel, see *SWIFT Input Channel*.

5. After configuring the SWIFTNet Server adapter in the Admin Console, click the **Enable Service for Business Process** check box on the Confirm page to enable the instance.
6. Once the SWIFTNet Server adapter is configured and saved, click the **Enabled** check box on the Services Configuration page. This starts the SWIFTNet MEFG Server.
7. On the Confirm page, verify that the **Enable Service for Business Processes** check box is selected to enable the adapter instance.

You must specify field settings in the application, using the Admin Console and the GPM.

## Creating or Setting Up a Adapter Configuration in the Admin Console

Use the field definitions in the following table to create a new configuration of the SWIFTNet Server adapter, or to set up the configuration provided with the application. Some fields are available in both the Admin Console and in the GPM.

**Note:** The business entities (accessible through the Business Entities wizard as part of the SWIFTNet Server adapter configuration) are shared by both RA1 and RA2. The Business Entities wizard enables you to add multiple entities.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	Select one of the options: <ul style="list-style-type: none"> <li>• None – Do not include the configuration in a service group at this time.</li> <li>• Create New Group – Enter a unique name for a new group, which will be created with this configuration. (You can then add other services to the group as well.)</li> <li>• Select Group – If service groups already exist for this service type, they are displayed in the list. Select a group from the list.</li> </ul> <p><b>Note:</b> Only select group if this adapter is clustered in a group. See <i>Managing Services and Services</i>.</p>
GIS Server IP	The callback IP of the application for the SWIFTNet MEFG Server. Required. <b>Note:</b> The default value is the IP address of the machine where the application is installed.

Field	Description
GIS HTTP Server Adapter Port	<p>This is the listening port for the SWIFTNet HTTP Server Adapter. Required. The default populated value is the instance port number of the application instance plus 53. For example, if the application instance port is 34600, the listening port populated by default is 34653.</p> <p><b>Note:</b> The HTTP Server adapter functions between the SWIFTNet Server adapter and the SWIFTNet MEFG Server. For an SSL connection, this value should be server name because the certificate is made with the server name.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
MEFG SWIFTNet IP	The IP address of the SWIFTNet MEFG Server. Required.
MEFG SWIFTNet Port	The port of the SWIFTNet MEFG Server. Required.
CLA2Client Listening Port	<p>The listening port used by the client command adapter (CLA2Client) running along the SWIFTNet MEFG Server. Required.</p> <p><b>Note:</b> This port listens for requests to start and stop the SWIFTNet MEFG Server.</p>
MEFG SWIFTNet Home	The home directory of the SWIFTNet MEFG Server. Required.
Use SSL	Whether to enable Secure Sockets Layer (SSL) over HTTP communication between the application and the SWIFTNet MEFG Server. Valid values are False (default) and True.
Cipher Strength	<p>Specifies the strength of the algorithms (cipher suites) used to encrypt data. Valid values are:</p> <ul style="list-style-type: none"> <li>• STRONG - Required if Use SSL is Must</li> <li>• ALL - All cipher strengths are supported</li> <li>• WEAK - Often required for international trade, because government regulations prohibit STRONG encryption from being exported</li> </ul> <p>Default is ALL. Required if SSL is checked.</p>
CA Certificate	Move one or more CA Certificates to the use column. These are the digital security certificates that the SSL server will use to authenticate the client. Optional.

Field	Description
Message Partner Client Name	The client message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server client application. <b>Note:</b> The Message Partner Client Name must correspond to the Application Interface Message Partner that is defined on the SAG as the client interface for the SWIFTNet MEFG Server.
Message Partner Server Name	The server message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server server application. <b>Note:</b> The Message Partner Server Name must correspond to the Application Interface Message Partner that is defined on the SAG as the client interface for the SWIFTNet MEFG Server.
Active-Active Configuration	Enables you to set up active-active configuration using two separate instances of the Remote API (RA), RA1 and RA2. Each RA should be configured to point to a different SAG to support failover processing. Possible values are True and False (default). Required. <b>Note:</b> This parameter specifies whether to support failover if one SAG fails. When this parameter is set to True, you are presented with parameters for both an RA1 Profile and an RA2 Profile. When you are operating in an environment with multiple SAGs configured in active-active mode, setting this parameter enables you to define an alternate RA connection to a secondary SAG for failover support.
SNL Endpoint (for Store and Forward only)	The SNL endpoint used to receive data from SnF queues (for example, <code>snl_sft</code> ). Optional—complete only if using store and forward processing. <b>Note:</b> You must define endpoints on the SAG to route the InterAct messages to the correct application interface. If you are using store-and-forward, an extra endpoint is required to route messages coming from the store-and-forward queue (you can use the default endpoint for store-and-forward, <code>snl_sft</code> ).
SnF Monitoring Interval (in seconds)	The store and forward monitoring interval (in seconds). Optional. <b>Note:</b> This parameter enables you to indicate the interval that you want the SWIFTNet MEFG Server to check on the queue status. The SWIFTNet MEFG Server sets a timer to send the <code>GetSnFStatusRequest</code> message based on the value you enter.
Return Signature List	Whether you want your own signature returned. Valid values are False (default - normal Crypto is used) and True. Optional for T-Copy and Y-Copy implementation.

Field	Description
Use Input Channel (for InterAct Store and Forward only)	<p>Whether to use the input channel with this adapter. Valid values are False (default) and True. You have to select True if you are using an input channel. Required.</p> <p><b>Note:</b> Used for InterAct store-and-forward only. If you configure this parameter, the SWIFTNet MEFG Server opens the Input Channel automatically during the startup (when the SWIFTNet Server Adapter is enabled). This Input Channel remains open until the SWIFTNet MEFG Server is shut down (or the SWIFTNet Server Adapter is disabled). During this time, you still have an option to send message using the input channel or without the input channel. All you need to do is to indicate this by using this parameter in SWIFTNet Client service.</p>
SWIFTNet RA	<p>The absolute path of the RA1 installation directory for RA1 SWIFTNet. Required. For example, <b>/SWIFTAlliance/RA</b>.</p> <p><b>Note:</b> This parameter specifies where to pick up the remote API and execute to SAG.</p>
Config	<p>The relative path of the RA1 instance configuration directory (relative to the RA installation directory). Required. For example, <b>RA1/cfg</b>.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Bin	<p>This is added to the PATH environment variable to contain the SWIFTNet MEFG Server binaries. Possible value is bin. Required.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>

Field	Description
Lib	<p>This is added to the library path environment variable. Possible value is lib. Required.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Category	<p>This is the category of RA. Possible values are:</p> <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> <p>Required.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Delivery Notification	<p>Determines whether the RA1 server is handling a delivery notification. Possible values are True and False (default). Optional. This is used for a FileAct get.</p>
Delivery Notification DN	<p>Distinguished name of the responder of the delivery notification. Optional.</p>
Request Type of Del. Notifn	<p>Request type of the delivery notification. This is used for a FileAct Get. Required.</p>
Send Del. Notifn before Backend Processing	<p>Indicates if the server will send a delivery notification before the internal process is executed. Required.</p>
Event Status Tracking	<p>Indicates if the server requires all the FileAct Event statuses to be returned. Valid values are:</p> <ul style="list-style-type: none"> <li>• Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>• Full (all statuses are returned)</li> </ul> <p>Required.</p>

Field	Description
SWIFTNet RA	<p>The absolute path of the RA2 installation directory for RA2 SWIFTNet. Required (based on Active-Active configuration). For example, <code>/SWIFTAlliance/RA</code>.</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p>
Config	<p>The relative path of the RA2 instance configuration directory (relative to the RA2 installation directory). Required (based on Active-Active configuration). For example, <code>/RA2/cfg</code>.</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Bin	<p>This is added to the PATH environment variable to contain the SWIFTNet MEFG Server binaries. Required (based on Active-Active configuration).</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Lib	<p>This is added to the library path environment variable. Required (based on Active-Active configuration).</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>

Field	Description
Category	<p>This is the category of RA2. Possible values are:</p> <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> <p>Required (based on Active-Active configuration).</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Delivery Notification	Determines whether the RA2 server is handling a delivery notification. Possible values are True and False (default). Optional. This is used for a FileAct get.
Delivery Notification DN	Distinguished name of the responder of the delivery notification. Optional.
Request Type of Del. Notifn	Request type of the delivery notification. This is used for a FileAct Get. Required.
Send Del. Notifn before Backend Processing	Indicates if the server will send a delivery notification before the internal process is executed. Required.
Event Status Tracking	<p>Indicates if the server requires all the FileAct Event statuses to be returned. Valid values are:</p> <ul style="list-style-type: none"> <li>• Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>• Full (all statuses are returned)</li> </ul> <p>Required.</p>
Input Channel Name	The name of the input channel. Required only if you specified <b>True</b> for <b>Use Input Channel</b> .
Authoriser DN	The authorized distinguished name that will be used to open the input channel. Required only if you specified <b>True</b> for <b>Use Input Channel</b> .
Force Open the Input Channel	Whether to force open the input channel or use normal mode. Valid values are False (use normal mode, which is the default) and True (force the input channel). Required only if you specified <b>True</b> for <b>Use Input Channel</b> .

Field	Description
Max. Resend Attempts	The maximum number of resend attempts allowed before the application automatically sends a Resolve Gap request to SWIFT. The default is 3. Required only if you specified <b>True</b> for <b>Use Input Channel</b> .
Run As User	Identify a user who has permission to run the scheduled activity. You can type the user ID, click the button to select the user ID from the list, and click <b>Save</b> . Optional. <b>Note:</b> You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Use 24 Hour Clock Display	By default, the scheduling wizard displays times using a 12-hour clock (which designates the time in hours as a.m. or p.m.). Use this option to display times using a 24-hour clock. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Do not use schedule	Removes all references to a schedule from the service. If you select this option, you cannot enable the schedule in the future. You must recreate the schedule instead. Use this option only when you do not need a schedule for the service. This is the default option. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Run based on timer	Run the service at a certain time or time interval, such as every 2 hours. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Select Time	Type the time at which you want the Resend Scheduler to run. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.



Field	Description
Run daily	Run the service one or more times every day. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Run based on days of the week	Run the service on certain days of the week, such as every Monday. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Run based on days of the month	Run the service on certain days of the month, such as the 1st or 15th of every month. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Schedule Exclusions	Allows you to add any schedule anomalies (when the Resend Scheduler should not run). <b>Note:</b> We recommend you leave this parameter blank (that is, do not create any schedule exclusions).
Date Exclusions	Allows you to add any date anomalies (any date on which the Resend Scheduler should not run). <b>Note:</b> We recommend you leave this parameter blank (that is, do not create any date exclusions).
New Business Entity	Click <b>add</b> to create a new business entity or click <b>edit</b> to modify an existing entity. <b>Note:</b> You must have at least one business entity created to proceed.

Field	Description
Entity	<p>Identifies the security context to be used. For the client, the business entity is the requester. For the server, the business entity is the responder. Required for each configured entity to access a proprietary SWIFTNet PK1 certificate to set up a valid security context.</p> <p><b>Note:</b> This is the distinguished name created by SWIFT. This parameter is only displayed if you edit an existing Business Entity or add a new Business Entity. The business entities are shared by both the RA1 and RA2 profiles.</p>
UserId	<p>The user identifier for this business entity (to log in to SWIFTNet). Required for each configured entity.</p> <p><b>Note:</b> The UserName is created in SAG (in the Users Module) and must also have a certificate created for it in the SAG. This parameter is only displayed if you edit an existing Business Entity or add a new Business Entity.</p>
Password	<p>The user password for this business entity (to log in to SWIFTNet). Required for each configured entity.</p> <p><b>Note:</b> This password is automatically encrypted. This parameter is only displayed if you edit an existing Business Entity or add a new Business Entity.</p>
Message Queue	<p>The name of the store and forward queue from which to receive messages. Required in Store and Forward mode.</p>
Notification Queue	<p>The Name of the store-and-forward queue to retrieve delivery notifications (optional; if empty, same as Message Queue). Required in Store and Forward mode.</p>
Acquire queue by force	<p>Whether to acquire the queue by force. Valid values are False (default) and True. Required.</p>
Use Default Delivery Notification	<p>Indicates whether to use the default delivery notification configuration on the RA1 page. Required.</p>
Delivery Notification (Del. Notifn)	<p>Indicates whether the sender asked the receiver to send a delivery notification. Optional. Valid values are True (default) or False.</p> <p><b>Note:</b> This parameter is only available when <b>Use Default Delivery Notification</b> is not selected.</p>

Field	Description
Request Type of Del. Notifn	If <b>Delivery Notification (Del. Notifn)</b> is set to True, the value of this parameter is used to request a specific delivery notification message from the remote receiving server application when it returns the delivery notification. Optional. <b>Note:</b> This parameter is only available when <b>Use Default Delivery Notification</b> is not selected.
Reception Directory	The full directory path where the file is received and stored during FileAct Put mode. Required for FileAct. Optional.
Download Directory	The full directory path where the file is picked up and sent to the requestor during FileAct Get mode. Required for FileAct. Optional.
Success Directory	The full directory path that must be specified when using the FileAct #OLDEST_FILE feature. Required for FileAct. Optional.

## Setting Up the Adapter in the GPM

Use the field definitions in the following table to set up the adapter configuration in the GPM:

Field	Description
Active-Active Configuration	Enables you to set up active-active configuration using two separate instances of the Remote API (RA), RA1 and RA2. Each RA should be configured to point to a different SAG to support failover processing. Possible values are True and False (default). Required. <b>Note:</b> This parameter specifies whether to support failover if one SAG fails. When this parameter is set to True, you are presented with parameters for both an RA1 Profile and an RA2 Profile. When you are operating in an environment with multiple SAGs configured in active-active mode, setting this parameter enables you to define an alternate RA connection to a secondary SAG for failover support.
commandLinePort	The listening port used by the client command adapter (CLA2Client) running along the SWIFTNet MEFG Server. Required. <b>Note:</b> This port listens for requests to stop the SWIFTNet MEFG Server.

Field	Description
deliveryNotification	Determines whether the server is handling a delivery notification. Possible values are True and False (default). Optional. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
Description	Error description for the rejected response. Optional. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
downloadDir	The full directory path where the file is picked up and sent to the requestor during FileAct Get mode. Required for FileAct.
Info	Error information for the rejected response. Optional. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
interfaceMode	SWIFTNet message type. Valid values are InterAct or FileAct. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
localServerAddress	The callback IP of the application for the SWIFTNet MEFG Server. Required.
localServerPort	The is the listening port for the SWIFTNet HTTP Server adapter. Required. <b>Note:</b> The HTTP Server adapter functions in between the SWIFTNet Server adapter and the SWIFTNet MEFG Server.
messageID	Message identifier for the incoming message. Required. <b>Note:</b> This is a unique identifier. This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
messagePartnerClient Name	The client message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server client application.
messagePartnerServer Name	The server message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server server application. <b>Note:</b> The Message Partner Server Name must correspond to the Application Interface Message Partner that is defined on the SAG as the client interface for the SWIFTNet MEFG Server.
RA1Bin	This is added to the PATH environment variable. Possible value is bin. Required.

Field	Description
RA1Category	This is the category of RA. Possible values are: <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> Required.
RA1Config	The relative path of the RA1 instance configuration directory (relative to the RA installation directory). Required.
RA1deliveryNotification	Determines whether the RA1 server is handling a delivery notification. Possible values are True and False (default). Optional.
RA1deliveryNotification DN	Distinguished name of the responder of the delivery notification. Optional.
RA1deliverynotification RT	Request type of the delivery notification. This is used for a FileAct Get. Required.
RA1eventstatusTracking	Indicates if the server requires all the FileAct Event statuses to be return. Valid values are: <ul style="list-style-type: none"> <li>• Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>• Full (all statuses are returned)</li> </ul> Required.
RA1Lib	This is added to the library path environment variable. Possible value is lib. Required.
RA1sendDNb4bkend Process	Indicates if the server will send a delivery notification before the internal process is executed. Required.
RA1Swiftnethome	The home directory of the SWIFTNet MEFG Server. Required. <b>Note:</b> This is an absolute path location. This parameter specifies where to pick up the remote API and execute to SAG.
RA2Bin	This is added to the PATH environment variable. Possible value is bin. Required. <b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.
RA2Category	This is the category of RA. Possible values are: <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> Required. <b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.
RA2Config	The relative path of the RA2 instance configuration directory (relative to the RA installation directory). Required.

Field	Description
RA2deliveryNotification	Determines whether the RA2 server is handling a delivery notification. Possible values are True and False (default). Optional.
RA2deliveryNotification DN	Distinguished name of the responder of the delivery notification. Optional.
RA2deliverynotification RT	Request type of the delivery notification. This is used for a FileAct Get. Required.
RA2eventstatusTracking	Indicates if the server requires all the FileAct Event statuses to be return. Valid values are: <ul style="list-style-type: none"> <li>Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>Full (all statuses are returned)</li> </ul> Required.
RA2Lib	This is added to the library path environment variable. Possible value is lib. Required.
RA2sendDNb4bkend Process	Indicates if the server will send a delivery notification before the internal process is executed. Required.
RA2Swiftnethome	The home directory of the RA2. Required. <b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True. <b>Note:</b> This is an absolute path location. This parameter specifies where to pick up the remote API and execute to SAG.
receptionDir	The full directory path where the file is received and stored during FileAct Put mode. Required for FileAct.
remoteServerAddress	The IP address of the SWIFTNet MEFG Server. Required.
remoteServerPort	The port of the SWIFTNet MEFG Server. Required.
SnF	Indicates if you are using the store-and-forward method. Valid values are True (use Store-and-Forward) and False (default—do not use Store-and-Forward). Required. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
snlEndPoint	The SNL endpoint used to receive data from SnF queues (for example, snl_sft). Optional—complete only if using store and forward processing.

Field	Description
Status	<p>The status of the message. Possible values are:</p> <ul style="list-style-type: none"> <li>• Accepted</li> <li>• Rejected</li> <li>• Failed</li> <li>• Duplicated</li> </ul> <p>Required.  <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.</p>
successDir	<p>The full directory path that must be specified when using the FileAct #OLDEST_FILE feature. Required for FileAct.</p>
swiftnetServerHome Directory	<p>The home directory where the SWIFTNet MEFG Server is installed. Required.</p>

## Business Process Example

The service part of the SWIFTNet Server adapter that is used in the business process is bootstrapped when the SWIFTNet MEFG Server posts the request through the URI defined in the HTTP Server adapter. For more information about the HTTP Server adapter, see *HTTP Server Adapter*.

## Renewing a Certificate

You can create a business process and schedule it to be executed at an interval of three months. You only need to pass in the distinguished name that is specified in the SWIFTNet Server Adapter Business Entities. When the request is passed to the SWIFTNet MEFG Server, it looks up the user identifier and the encrypted password in the configuration file. The SWIFTNet MEFG Server then performs an `initRequest` and `CreateSecurityContext` to open the certificate.

```

<operation>

  <participant
name="SWIFTNetClientService"/>

  <output
message="renewSecurityContext">

    <assign
to="renewDN">o=yourDN,o=swift</assign>

    <assign
to="." from="*"/>

  </output>

  <input
message="testing">

    <assign
to="." from="*"/>

  </input></operation>

```

## Interact Business Process Without Store-and-Forward Processing

The following business process example (in which the service part of the SWIFTNet Server adapter as part of InterAct processing) is used if you are not using store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerRequest business process.

```
<process name="handleSWIFTNetServerRequest">
  <sequence>
    <operation
name="set user token">
      <participant
name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign
to="USER_TOKEN">admin</assign>
        <assign
to="." from="*"/>
      </output>
      <input
message="inmsg">
        <assign to="." from="*"/>
      </input>
    </operation>
    <operation
name="SoapIn">
      <participant name="SOAPInbound"/>
      <output
message="output">
        <assign to="." from="*"/>
        <assign
to="bootstrap">>false</assign>
        <assign
to="SOAP_INTERMEDIATE_NODE">>false</assign>
      </output>
      <input
message="input">
        <assign to="." from="*"/>
      </input>
    </operation>
    <operation>
      <participant
name="SWIFTNetServerAdapter"/>
      <output
message="handleServerRequest">
        <assign
to="." from="*"/>
      </output>
      <input
message="testing">
        <assign to="." from="*"/>
      </input>
    </operation>
    <!--
internal processing by invoking a subprocess -->
    <!--
business-specific processing that will return a response for InterAct
-->
    <operation>
      <participant
name="InvokeSubProcessService"/>
      <output
message="Xout">
```



```

        <assign to="INVOKE_MODE">SYNC</assign>
        <assign
to="." from="*" />
    </output>
    <input
message="Xin">
        <assign to="." from="*" />
    </input>
</operation>
<!--
this is to construct the server response message back to GIS Server
application -->
    <operation>
        <participant
name="SWIFTNetServerAdapter" />
        <output
message="handleServerResponse">
            <assign
to="." from="*" />
            <assign to="interface"
from="SwiftServerRequest/interface/text()" />
            <assign
to="messageID" from="SwiftServerRequest/messageID/text()" />
            <assign
to="Status">Accepted</assign>
            <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()" />
            <assign
to="SnF" from="SwiftServerRequest/SnF/text()" />
        </output>
        <input
message="testing">
            <assign to="." from="*" />
        </input>
    </operation>
    <operation
name="SoapOut">
        <participant name="SOAPOutbound" />
        <output
message="output">
            <assign to="." from="*" />
            <assign
to="SOAP_MODE">respond</assign>
        </output>
        <input
message="input">
            <assign to="." from="*" />
        </input>
    </operation>
    <assign
to="doc-has-headers">>true</assign>
    <operation
name="HttpResponse">
        <participant name="HttpRespond" />
        <output
message="Xout">
            <assign to="." from="*" />
        </output>
        <input
message="Xin">
            <assign to="." from="*" />
        </input>
    </operation>
</onFault>
<!--
On Fault, we will clear PrimDoc, construct Rejected response and
soap-envelope it -->

```

```

    <sequence>
      <operation
name="ReleasePrimDoc">
        <participant
name="ReleaseService"/>
        <output message="outmsg">
          <assign
to="TARGET"/>/ProcessData/PrimaryDocument</assign>
          <assign
to="." from="*"/>
        </output>
        <input
message="inmsg"/>
      </operation>
      <operation>
        <participant
name="SWIFTNetServerAdapter"/>
        <output
message="handleServerResponse">
          <assign
to="." from="*"/>
          <assign to="interface"
from="SwiftServerRequest/interface/text()"/>
          <assign
to="messageID" from="SwiftServerRequest/messageID/text()"/>
          <assign
to="Status">Rejected</assign>
          <assign
to="Description">Unable to get the Server Response</assign>
          <assign
to="Info">Failure in getting the Server Response</assign>
          <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()"/>
          <assign
to="SnF" from="SwiftServerRequest/SnF/text()"/>
        </output>
        <input
message="testing">
          <assign to="."
from="*"/>
        </input>
      </operation>
      <operation
name="SoapOut">
        <participant name="SOAPOutbound"/>
        <output
message="output">
          <assign to="."
from="*"/>
          <assign to="SOAP_MODE">respond</assign>
        </output>
        <input
message="input">
          <assign to="." from="*"/>
        </input>
      </operation>
      <assign
to="doc-has-headers">true</assign>
      <operation
name="HttpResponse">
        <participant
name="HttpRespond"/>
        <output message="Xout">
          <assign
to="." from="*"/>
        </output>
      </input>

```

```

message="Xin">
  <assign to="." from="*" />
  </input>
  </operation>
</sequence>
</onFault>
</sequence>
</process>

```

## InterAct Business Process With Store-and-Forward Processing

The following business process example demonstrates the service part of the SWIFTNet Server adapter being used as part of InterAct processing if you are using store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerSnFRequest business process.

```

<process name="handleSWIFTNetServerSnFRequest">

  <rule name="IsAuthNotification">
    <condition>SwiftServerRequest/AuthResponse
= 'TRUE'</condition>
  </rule>

  <sequence>
    <operation name="set
user token">
      <participant name="SetUserToken" />

      <output message="SetUserTokenMessage">

        <assign to="USER_TOKEN">admin</assign>

        <assign to="." from="*" />
      </output>

      <input message="inmsg">
        <assign
to="." from="*" />
      </input>

    </operation>
    <operation name="SoapIn">

      <participant name="SOAPInbound" />

      <output message="output">
        <assign
to="." from="*" />
        <assign to="bootstrap">>false</assign>

        <assign to="SOAP_INTERMEDIATE_NODE">>false</assign>

      </output>
      <input message="input">

        <assign to="." from="*" />
      </input>

    </operation>
  </operation>

  <participant name="SWIFTNetServerAdapter" />

  <output message="handleServerRequest">

```

```

        <assign to="." from="*" />
    </output>

    <input message="testing">
        <assign
to="." from="*" />
    </input>

</operation>
<choice name="AddToMailbox">

    <select>
        <case ref="IsAuthNotification"
negative="true" activity="Mailbox Add Service" />

    </select>

    <!-- internal processing
for SnF is to put into a Mailbox so that it can bootstrap internal
business process later-->
    <!-- Mailbox
path is based on SwiftServerRequest/responderDN/requestorDN/for
InterAct -->
    <operation name="Mailbox
Add Service">
        <participant name="MailboxAdd" />

        <output message="AddRequest">

            <assign to="." from="*" />

            <assign to="MailboxPath" from="concat('/',
SwiftServerRequest/responderDN/text(), '/',
SwiftServerRequest/requestorDN/text())" />

            <assign to="ContentType">ascii</assign>

        </output>
        <input message="inmsg">

            <assign to="AddResults" from="*" />

        </input>
    </operation>

</choice>
<operation>

    <participant name="SWIFTNetServerAdapter" />

    <output message="handleServerResponse">

        <assign to="." from="*" />

    <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

        <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

        <assign to="Status">Accepted</assign>

        <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()" />

        <assign to="SnF" from="SwiftServerRequest/SnF/text()" />

    </output>
    <input message="testing">

```

```

        <assign to="." from="*" />
    </input>

</operation>
<operation name="SoapOut">

    <participant name="SOAPOutbound" />

    <output message="output">
        <assign
to="." from="*" />
        <assign to="SOAP_MODE">respond</assign>

    </output>
    <input message="input">

        <assign to="." from="*" />
    </input>

</operation>
    <assign to="doc-has-headers">true</assign>

<operation name="HttpResponse">
<participant name="HttpRespond" />

    <output message="Xout">
        <assign
to="." from="*" />
    </output>

    <input message="Xin">
        <assign
to="." from="*" />
    </input>

</operation>
<onFault>

<sequence>
    <operation name="ReleasePrimDoc">

        <participant name="ReleaseService" />

        <output message="outmsg">

            <assign to="TARGET">/ProcessData/PrimaryDocument</assign>

            <assign to="." from="*" />

        </output>
        <input message="inmsg" />

    </operation>
    <operation>

        <participant name="SWIFTNetServerAdapter" />

        <output message="handleServerResponse">

            <assign to="." from="*" />

            <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

            <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

```

```

        <assign to="Status">Rejected</assign>
        <assign to="Description">Unable to get the Server Response</assign>
        <assign to="Info">Failure in getting the Server Response</assign>
        <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()"/>
        <assign to="SnF" from="SwiftServerRequest/SnF/text()"/>
    </output>
    <input message="testing">
        <assign to="." from="*" />
    </input>
</operation>
<operation name="SoapOut">
<participant name="SOAPOutbound"/>
    <output message="output">
        <assign to="." from="*" />
    <assign to="SOAP_MODE">respond</assign>
    </output>
    <input message="input">
        <assign to="." from="*" />
    </input>
</operation>
<assign to="doc-has-headers">>true</assign>
<operation name="HttpResponse">
    <participant name="HttpRespond"/>
    <output message="Xout">
        <assign to="." from="*" />
    </output>
    <input message="Xin">
        <assign to="." from="*" />
    </input>
</operation>
</sequence>
</onFault>
</sequence>
</process>

```

## Fileact Business Process Without Store-and-Forward Processing

The following business process example shows the service part of the SWIFTNet Server adapter as part of FileAct processing without using store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerRequest business process.

```

<process name="handleSWIFTNetServerFARequest">
  <sequence>
    <operation
name="set user token">
      <participant
name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign
to="USER_TOKEN">admin</assign>
        <assign
to="." from="*"/>
      </output>
      <input
message="inmsg">
        <assign to="." from="*"/>
      </input>
    </operation>
    <operation
name="SoapIn">
      <participant name="SOAPInbound"/>
      <output
message="output">
        <assign to="." from="*"/>
        <assign
to="bootstrap">>false</assign>
        <assign
to="SOAP_INTERMEDIATE_NODE">>false</assign>
      </output>
      <input
message="input">
        <assign to="." from="*"/>
      </input>
    </operation>
    <operation>
      <participant
name="SWIFTNetServerAdapter"/>
      <output
message="handleServerRequest">
        <assign
to="." from="*"/>
      </output>
      <input
message="testing">
        <assign to="." from="*"/>
      </input>
    </operation>
    <!--
this is to construct the server response message back to GIS Server
application -->
    <operation>
      <participant
name="SWIFTNetServerAdapter"/>
      <output
message="handleServerResponse">
        <assign
to="." from="*"/>
        <assign to="interfaceMode"
from="SwiftServerRequest/interfaceMode/text()"/>
        <assign
to="messageID" from="SwiftServerRequest/messageID/text()"/>
        <assign
to="Status">Accepted</assign>
        <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()"/>

```

```

    <assign
to="SnF" from="SwiftServerRequest/SnF/text()"/>
    </output>
    <input
message="testing">
    <assign to="." from="*" />
    </input>
    </operation>
    <operation
name="SoapOut">
    <participant name="SOAPOutbound"/>
    <output
message="output">
    <assign to="." from="*" />
    <assign
to="SOAP_MODE">respond</assign>
    </output>
    <input
message="input">
    <assign to="." from="*" />
    </input>
    </operation>
    <assign
to="doc-has-headers">true</assign>
    <operation
name="HttpResponse">
    <participant name="HttpRespond"/>
    <output
message="Xout">
    <assign to="." from="*" />
    </output>
    <input
message="Xin">
    <assign to="." from="*" />
    </input>
    </operation>
    <onFault>
    <!--
On Fault, we will clear PrimDoc, construct Rejected response and
soap-envelope it -->
    <sequence>
    <operation
name="ReleasePrimDoc">
    <participant
name="ReleaseService"/>
    <output message="outmsg">
    <assign
to="TARGET"/>/ProcessData/PrimaryDocument</assign>
    <assign
to="." from="*" />
    </output>
    <input
message="inmsg"/>
    </operation>
    <operation>
    <participant
name="SWIFTNetServerAdapter"/>
    <output
message="handleServerResponse">
    <assign
to="." from="*" />
    <assign to="interfaceMode"
from="SwiftServerRequest/interfaceMode/text()"/>
    <assign
to="messageID" from="SwiftServerRequest/messageID/text()"/>
    <assign
to="Status">Rejected</assign>

```



```

    <assign
to="Description">Unable to get the Server Response</assign>
    <assign
to="Info">Failure in getting the Server Response</assign>
    <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()"/>
    <assign
to="SnF" from="SwiftServerRequest/SnF/text()"/>
    </output>
    <input
message="testing">
    <assign to="."
from="*" />
    </input>
    </operation>
    <operation
name="SoapOut">
    <participant name="SOAPOutbound" />
    <output
message="output">
    <assign to="."
from="*" />
    <assign to="SOAP_MODE">respond</assign>
    </output>
    <input
message="input">
    <assign to="." from="*" />
    </input>
    </operation>
    <assign
to="doc-has-headers">true</assign>
    <operation
name="HttpResponse">
    <participant
name="HttpRespond" />
    <output message="Xout">
    <assign
to="." from="*" />
    </output>
    <input
message="Xin">
    <assign to="." from="*" />
    </input>
    </operation>
    </sequence>
    </onFault>
    </sequence>
</process>

```

## FileAct Business Process With Store-and-Forward Processing

The following business process example shows the service part of the SWIFTNet Server adapter used as part of FileAct processing with store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerFASnFRequest business process.

```

<process name="handleSWIFTNetServerFASnFRequest">
<rule name="UndefinedCopyOrForceReject">
    <condition>SwiftServerRequest/AuthRequest = 'N' or
SwiftServerRequest/FileInfoForceMode =
'Rejected'</condition>
    </rule>

```

```

<rule name="AuthorizationNeeded">
  <condition>SwiftServerRequest/AuthRequest
  = 'Y' and SwiftServerRequest/FileInfoForceMode != 'Refused'</condition>
</rule>
<rule name="ForceRefusal">
  <condition>SwiftServerRequest/FileInfoForceMode = 'Refused'</condition>
</rule>
<sequence>
  <operation name="set user token">
    <participant name="SetUserToken"/>
    <output message="SetUserTokenMessage">
      <assign to="USER_TOKEN">admin</assign>
      <assign to="." from="*" />
    </output>
    <input message="inmsg">
      <assign
to="." from="*" />
      </input>
    </operation>
    <operation name="SoapIn">
      <participant name="SOAPInbound"/>
      <output message="output">
        <assign
to="." from="*" />
        <assign to="bootstrap">>false</assign>
        <assign to="SOAP_INTERMEDIATE_NODE">>false</assign>
      </output>
      <input message="input">
        <assign to="." from="*" />
      </input>
    </operation>
    <operation>
      <participant name="SWIFTNetServerAdapter"/>
      <output message="handleServerRequest">
        <assign to="." from="*" />
      </output>
      <input message="testing">
        <assign
to="." from="*" />
        </input>
      </operation>
      <choice name="NeedAuthorization">
        <select>
          <case ref="AuthorizationNeeded"

```

```

activity="Mailbox Add Service"/>
</select>

<!-- Put into a Mailbox so that it can bootstrap internal
authorization business process later -->

<!-- Mailbox path is based on SwiftServerRequest/recipientDN/requestorDN/
-->
<operation name="Mailbox Add
Service">
  <participant name="MailboxAdd"/>

  <output message="AddRequest">

    <assign to="." from="*" />

    <assign to="PrimaryDocument" from="HeaderInfo/@SCIObjectID"/>

    <assign to="MessageName" from="concat('ThirdParty_',
SwiftServerRequest/copySnFReference/text())"/>

    <assign to="MailboxPath" from="concat('/',
SwiftServerRequest/recipientDN/text(), '/',
SwiftServerRequest/requestorDN/text())"/>

    <assign to="ExtractableCount">1</assign>

    <assign to="ContentType">ascii</assign>

  </output>
  <input message="inmsg">

    <assign to="AddResults" from="*" />

  </input>
</operation>

</choice>
<choice name="IsUndefinedCopyOrForceReject">

  <select>
    <case ref="UndefinedCopyOrForceReject"
negative="true" activity="AcceptRequest"/>

    <case ref="UndefinedCopyOrForceReject" activity="RejectRequest"/>
  </select>

  <operation name="AcceptRequest">

    <participant name="SWIFTNetServerAdapter"/>

    <output message="handleServerResponse">

      <assign to="." from="*" />

      <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()"/>

      <assign to="messageID" from="SwiftServerRequest/messageID/text()"/>

      <assign to="Status">Accepted</assign>

      <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()"/>

      <assign to="SnF" from="SwiftServerRequest/SnF/text()"/>

    </output>

```

```

        <input message="testing">
        <assign to="." from="*" />
    </input>
    </operation>

    <sequence name="RejectRequest">

        <operation name="ReleasePrimDoc">
            <participant name="ReleaseService" />
            <output message="outmsg">

                <assign to="TARGET">/ProcessData/PrimaryDocument</assign>

                <assign to="." from="*" />

            </output>
            <input message="inmsg" />

        </operation>
        <operation
name="Form Reject Response">
            <participant
name="SWIFTNetServerAdapter" />
            <output
message="handleServerResponse">

                <assign to="." from="*" />

                <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

                <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

                <assign to="Status">Rejected</assign>

                <assign to="Description">Copy Profile is undefined
or Responder forced to reject</assign>

                <assign to="Info">Unable to determine copy mode or
FileInfo force responder's rejection</assign>

                <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()" />

                <assign to="SnF" from="SwiftServerRequest/SnF/text()" />

            </output>
            <input message="testing">

                <assign to="." from="*" />

        </input>
    </operation>

    </sequence>
</choice>

    <operation name="SoapOut">
        <participant
name="SOAPOutbound" />
        <output message="output">

            <assign to="." from="*" />

```

```

<assign to="SOAP_MODE">respond</assign>

  </output>
  <input message="input">

    <assign to="." from="*" />
  </input>

</operation>
  <assign to="doc-has-headers">>true</assign>

<operation name="HttpResponse">
  <participant name="HttpRespond" />

  <output message="Xout">
    <assign
to="." from="*" />
  </output>

  <input message="Xin">
    <assign
to="." from="*" />
  </input>

</operation>
  <choice name="IsThirdPartyForceRefusal">

    <select>
      <case ref="ForceRefusal"
activity="InvokeForceRefusalProcess" />
    </select>
    <operation name="InvokeForceRefusalProcess">

      <participant name="InvokeBusinessProcessService" />

      <output message="Invoke_In">

        <assign to="." from="*" />

      <assign to="INVOKE_MODE">ASYNC</assign>

      <assign to="WFD_NAME">SWIFTNet3rdPartyClientForceRefusal</assign>

    </output>
    <input message="Invoke_Out">

      <assign to="." from="*" />

    </input>
  </operation>

</choice>
  <onFault>

    <sequence>
      <operation name="ReleasePrimDoc">

        <participant name="ReleaseService" />

        <output message="outmsg">

          <assign to="TARGET">/ProcessData/PrimaryDocument</assign>

          <assign to="." from="*" />

```

```

</output>
  <input message="inmsg"/>
</operation>
<operation>

  <participant name="SWIFTNetServerAdapter"/>

  <output message="handleServerResponse">

    <assign to="." from="*" />

    <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

    <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

    <assign to="Status">Rejected</assign>

    <assign to="Description">Unable to get the Server Response</assign>

    <assign to="Info">Failure in getting the Server Response</assign>

    <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()" />

    <assign to="SnF" from="SwiftServerRequest/SnF/text()" />

  </output>
  <input message="testing">

    <assign to="." from="*" />

  </input>
</operation>

<operation name="SoapOut">
<participant name="SOAPOutbound"/>

  <output message="output">

    <assign to="." from="*" />

    <assign to="SOAP_MODE">respond</assign>

  </output>
  <input message="input">

    <assign to="." from="*" />

  </input>
</operation>

<assign to="doc-has-headers">>true</assign>

<operation name="HttpResponse">

  <participant name="HttpRespond"/>

  <output message="Xout">

    <assign to="." from="*" />

  </output>
  <input message="Xin">

    <assign to="." from="*" />

```

```

        </input>
        </operation>

    </sequence>
    </onFault>

</sequence>
</process>

```

## Parameters Passed From Business Process to Adapter

The following table contains the parameters passed from the business process to the SWIFTNet Server adapter:

Parameter	Description
messageID	Message identifier for the incoming message. Required.
interfaceMode	This is the SWIFTNet interface. Possible values are InterAct (default) or FileAct. Required.
deliveryNotification	Determines whether the server is handling a delivery notification. Valid values are True and False. Required.
SnF	Indicates whether you are using the store-and-forward method. Valid values are True (use store-and-forward) and False (do not use store-and-forward—this is the default). Required.
Status	The status of the message. Possible values are: <ul style="list-style-type: none"> <li>• Accepted</li> <li>• Rejected</li> <li>• Failed</li> <li>• Duplicated</li> </ul> Required.
Description	Description of the message. Optional. <b>Note:</b> Only necessary when there is an error message.
Information	Information about the message. Optional. <b>Note:</b> Only necessary when there is an error message.

## Enabling SWIFTNet Document Tracking

You need to enable document tracking in the system business process you are using for the SWIFTNet Server adapter—`handleSWIFTNetServerRequest` (if you are not using store-and-forward processing) or `handleSWIFTNetServerSnFRequest` (if you are using store-and-forward processing)—so the system can track the document during the process. In the business process text editor, you can easily enable SWIFTNet document tracking in the application by selecting the **Document Tracking** check box on the Process Levels page. Set the following options as needed and leave the rest of the business process parameters as the defaults:

- On the **Deadline Settings** page, set the deadline and notification options, if necessary.
- On the **Life Span** page, set the life span, if necessary.

## SWIFTNet Header Info Support

With SWIFTNet version 6.1, SWIFT introduced a new FileAct header field (HeaderInfo) to contain key summary information related to the file. The presence of Sw:HeaderInfo within the request is an indication to invoke the feature to validate the Sw:HeaderInfo. With SWIFTNet version 6.3, SWIFT will perform stronger central validation on the HeaderInfo fields for FileAct.

Once a service is activated for the validation, SWIFT checks the HeaderInfo contents (that is, presence, syntax, and semantic). SWIFT rejects files with HeaderInfo contents that either do not pass this validation, or that do not use the HeaderInfo field according to the rules defined for the service.

### Header Info Support on the Client (the Application as Requestor)

As a requestor, the HeaderInfo is only allowed on the FileAct Put Request message. To specify the Header Info information, you must set the HeaderInfo parameter in the SWIFTNet Client Service accordingly. Please refer to *SWIFTNet Client Service* documentation for more details.

The application then validates the HeaderInfo if there is a matching **Request Type** profile in the SWIFTNet Service Profile before sending out the HeaderInfo. Therefore, when you specify HeaderInfo during sending, you must configure the **Request Type** profile in the SWIFTNet Service Profile. You can configure the HeaderInfo as a mandatory or an optional parameter; however, if there is no matching profile, this request is forbidden.

**Note:** Please do not include <Sw:HeaderInfo> tag when specifying the value of HeaderInfo parameter. This <Sw:HeaderInfo> tag is automatically included during the process.

### Header Info Support on the Server (the Application as Responder)

As a responder, the HeaderInfo is only allowed on the FileAct Get Response message. When the FileAct Get request is received, the responder checks to see if there is a HeaderInfo file located in the same directory as the download file. The HeaderInfo file must have the same name as the logical filename specified in the request except with an additional filename extension (.hdr). For example, if the logical filename is **payload.txt**, the HeaderInfo filename should be **payload.txt.hdr**. Prior to the Get request, the responder is responsible to provide both the download file and the HeaderInfo file in the correct directory.

When the FileAct Get request is received, this HeaderInfo is validated by the application using the SWIFTNet Service Profile. Depending on the Request Type profile in the SWIFTNet Service Profile, the HeaderInfo file can be mandatory, optional, or forbidden.

**Note:** Please do not include <Sw:HeaderInfo> tag when specifying the content of HeaderInfo file (\*.hdr). This <Sw:HeaderInfo> tag is automatically included during the process.



## SWIFTNet Service Profile

The HeaderInfo block is optional, except for those services that mandate it. If the HeaderInfo block is not used, it must not be present, and if it is used, it must be validated by the schema.

The SWIFTNet Service Profile enables you to easily port Service Profiles from one application instance to another. This function allows you to associate SWIFTNet Request Type with a Schema for Header Validation. You need to create the SWIFTNet Service Profile and associate the request type with the selected schema. This allows the application to validate the HeaderInfo when it is present in the request.

**Note:** The schema must be saved in application.

The Request Type parameter can accept a wildcard (\*) to be used only at the end of the string. To determine which Service Profile to be used for a particular Request Type, the application uses a best-match policy. For example, if there are two Service Profile defined, for pain.\* and pain.001.\*, and the actual request type is pain.002.001, then the first one will be selected.

Two SWIFTNet Service Profiles are preloaded into application. The **pac**s.\* and **pa**ins.\* service profiles are associated with the Transaction Count schema and set to **Required for validation**. The Transaction Count and Payment Summary schemas are also preloaded into the application.

You can also import and export SWIFTNet Service Profiles from one application instance to another.

### Creating a SWIFTNet Service Profile

To create a SWIFTNet service profile:

1. From the application **Deployment** menu, select **Adapter Utilities > SWIFTNet Service Profile**.
2. To the right of **Create new SWIFTNet service profile**, click **Go!**.
3. Complete the following parameters and click **Next**:

Parameter	Description
Request Type	Type the request type. A wildcard (*) is only allowed at the end of the string, for example for example, <b>pac</b> s.* or <b>pac</b> s.001.*. Required.
Schema Name	Select the schema used to validate the header information for this request type. Required.
Validation Type	Select whether validation is mandatory or should only be used if header information is specified. Optional. Valid values are: <ul style="list-style-type: none"><li>• Validates only if Header Information is specified (default)</li><li>• Validation of Header Information is required</li></ul>

4. Click **Finish** to save the service profile.

### Searching for a SWIFTNet Request Type

To edit or delete a SWIFTNet request type, you must first locate the appropriate request type. You can locate a specific request type in two ways:

- Search for the request type by name.
- Select the request type from an alphabetical list.

Searching for the request type by name is more precise and provides fewer results. Searching from an alphabetical list will result in a list of all request type or all types beginning with a specified letter or digit.

Once you search for the request type, you can easily edit or delete it from the SWIFTNet Service Profile interface.

### Searching for a Request Type by Name

To search for a request type by name:

1. From the application **Deployment** menu, select **Adapter Utilities > SWIFTNet Service Profile**.
2. In the Search section, type the name of the request type. Case does not matter and you can type part of a name.  
The application returns a list of matches unless no request type meet the criteria you specified.
3. When the list of matches is returned, click **edit** next to the request type you want to modify, or click **delete** next to the request type you want to remove.

### Searching for a Request Type from a List

To select a request type from a list:

1. From the application **Deployment** menu, select **Adapter Utilities > SWIFTNet Service Profile**.
2. In the List section, select one of the following:
  - Alphabetically – Select **All** and click **Go!**
  - Alphabetically – Select a specific letter or digit (0 - 9) and click **Go!**The application returns a list of matches unless no request type meet your criteria.
3. When the list of matches is returned, click **edit** next to the request type you want to modify, or click **delete** next to the request type you want to remove.

### Exporting and Importing a SWIFTNet Service Profile

The application Import/Export feature enables you to save time and increase the accuracy of duplicating supported resources on different environments that are set up for unique purposes. To import and export resources from one application environment to another application environment, both environments must be the same version.

---

## Chapter 38. SWIFTNet Server Adapter (Build 5104 or higher)

The SWIFTNet Server adapter communicates to the SWIFTNet Network through the SWIFTNet MEFG server. It responds to and accepts InterAct and FileAct messages that are sent by remote SWIFTNet correspondents.

The SWIFTNet Server Adapter includes new parameters for the Command Line Adapter 2 authentication and SSL options. All default or custom preconfigured SWIFTNet Adapter instances have authentication that is enabled and SSL that is disabled. For more information about the changes to the Command Line Adapter 2 server, see *Command Line Adapter 2 (Build 5104 or higher)*.

For existing SWIFTNet Server Adapter instances (including custom instances), the new parameters are blank and a value of None Provided is displayed on the Service Configuration Summary page. However, during execution these values default to Authentication = Yes, System Certificate = cla2auth, and CLA2 SSL = No. You can edit these default values in the corresponding adapter instance and save your changes.

All new instances of the SWIFTNet Server Adapter have Authentication that is enabled by default and SSL that is disabled by default.

**Tip:** You are encouraged to at least enable the Authentication option.

The following table provides an overview of the SWIFTNet Server adapter:

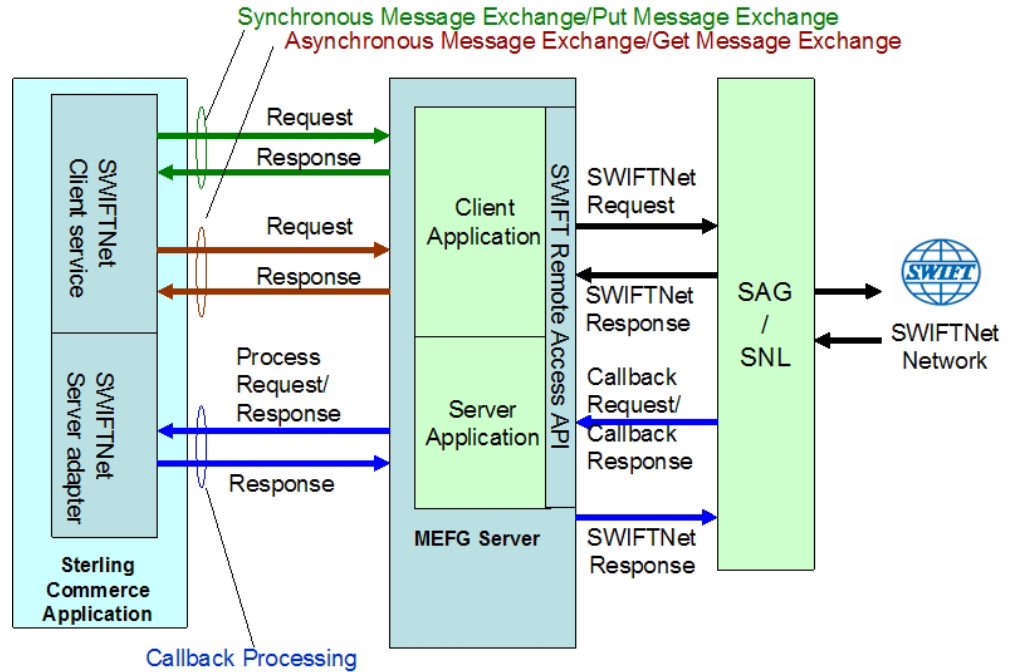
System Name	SWIFTNet Server Adapter
Graphical Process Modeler (GPM) categories)	All services
Description	This adapter is responsible for receiving and responding to SWIFTNet InterAct and FileAct messages using the SWIFTNet MEFG Server.
Business usage	A business would use this adapter in order to exchange SWIFTNet InterAct and FileAct messages with its trading partners over the SWIFTNet system. <b>Note:</b> You will also need to configure this adapter if you are transporting CHIPS messages using the SWIFTNet transport mode. This adapter also sends acknowledgements to CHIPS.
Usage example	When an InterAct or FileAct message is received, a business process is executed to process the message and, when required, to generate the SWIFTNet response.
Preconfigured?	This adapter is preconfigured as part of the application installation.
Requires third party files?	No third party files are required.
Platform availability	All supported application platforms.

Related services	This is designed to work in conjunction with the SWIFTNet MEFG Server and the Command Line Adapter 2. This service also works with the SWIFTNet HTTP Server adapter to provide SSL support.
Application requirements	SSL can be implemented between the application and the MEFG Server if the SWIFTNet HTTP Server adapter is configured for that setup.
Initiates business processes?	Initiates system business processes.
Invocation	By the Multi-Enterprise Financial Gateway for SWIFTNet application.
Business process context considerations	None.
Returned status values	<ul style="list-style-type: none"> <li>• Fatal—non-recoverable error</li> <li>• Transient—recoverable error</li> <li>• Logic—recoverable error</li> <li>• Success—Success</li> <li>• Warning—Success with warning</li> </ul>
Restrictions	Only one SWIFTNet MEFG Server can be configured to talk to one SWIFTNet Server adapter instance in the application.
Persistence level	N/A
Testing considerations	N/A

## How the SWIFTNet Server Adapter Works

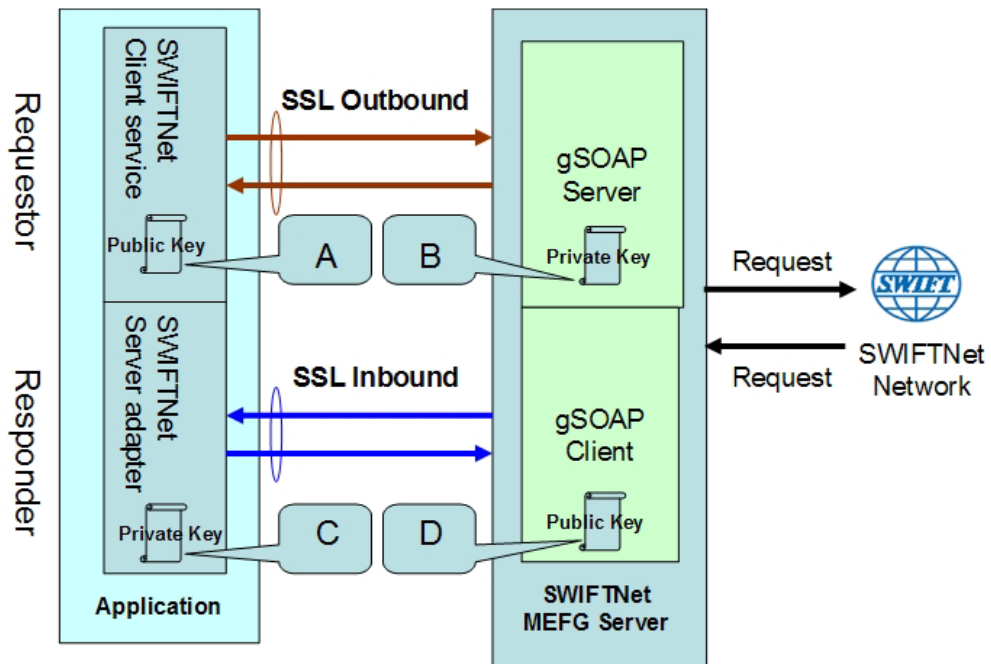
The SWIFTNet Server adapter is comprised of two parts: the service part and the adapter part. The service part is used in a business process that does not require configuration except for enabling it for document tracking. The adapter part is configured through the Admin Console or the GPM, and this adapter is responsible for starting and stopping the SWIFTNet MEFG Server from the application using the Command Line Adapter 2 (CLA2), which is built into the SWIFTNet Server adapter. Starting and stopping the operation of the SWIFTNet MEFG Server will only work correctly if the CLA2Client.jar is deployed in the same machine where the SWIFTNet MEFG Server is installed. The CLA2Client.jar file must also be started by a user who has permission to access the SWIFTNet MEFG Server home directory.

This diagram illustrates the process flow between the application and the SWIFTNet network through the SWIFTNet MEFG Server (not using SSL):



The SWIFTNet Server adapter (in conjunction with the SWIFTNet HTTP Server adapter) enables you to use Secure Sockets Layer (SSL) to provide secure authentication, using the SWIFTNet HTTP Server adapter to accept the forwarded request from the SWIFTNet MEFG Server. When you use SSL with the application, two channels are secured: an Outbound channel (the application acting as the Requestor) and an Inbound channel (the application acting as the Responder).

This diagram illustrates the configuration necessary between the application and the SWIFTNet network through the SWIFTNet MEFG Server to set up the Outbound and Inbound channels (using the SWIFTNet HTTP Server adapter for SSL):



You will need 2 pairs of certificates. The first pair belongs to the SWIFTNet MEFG Server (A and B in the diagram above) and is used to secure the outbound channel. The second pair of certificates belongs to the application (C and D in the diagram above) and is used to secure the inbound channel. In the above diagram, the callouts signify the following:

- A — A public key certificate file belongs to the SWIFTNet MEFG Server that is configured on the SWIFTNet Client service (the certificate is specified for the CA Certificate parameter).
- B — A private key certificate file that is stored on the SWIFTNet MEFG Server as a key file (which you configure through the SSL Configuration utility named `sslUtil.jar` in the SWIFTNet MEFG Server installation bin subdirectory). The `sslUtil.jar` file is located in the bin subdirectory of the SWIFTNet MEFG Server installation directory.
- C — A private key certificate file that is configured on the SWIFTNet HTTP Server adapter (the certificate is specified for the System Cert parameter).
- D — A public key file that belongs to the application and is stored for the SWIFTNet MEFG Server as a CA Cert file or trusted list (that you configure through the SSL Configuration utility named `sslUtil.jar` in the SWIFTNet MEFG Server installation directory).

**Note:** To configure SSL on the SWIFTNet MEFG Server, run the following command in the bin directory of the MEFG SWIFTNet Server installation bin sub-directory:

```
java -jar sslUtil.jar
```

**Note:** The application enables you easily renew certificates. See *Renewing a Certificate* for more information.

## Implementing the SWIFTNet Server Adapter

To implement the SWIFTNet Server adapter, complete the following tasks:

1. Enable the Command Line Adapter 2, see *Enabling the Command Line Adapter 2*.

**Tip:** For a secure configuration, enable both the Authentication and the SSL security options available in the Command Line Adapter 2 configuration.

2. Create a configuration of the Command Line Adapter 2, see *Command Line Adapter 2 (Build 5104 or higher)*. For information about the remote installation of the Command Line Adapter 2, see *Installing the Command Line Adapter 2 server remotely*.

**Restriction:** You must enable the same secure options on the SWIFTNet Server adapter that you enable when configuring the Command Line Adapter 2.

3. Create a configuration of the SWIFTNet Server adapter. See *Managing Adapters and Services*. For information about the fields specific to this adapter, see *Configuring the SWIFTNet Server Adapter*.
4. Specify field settings for the adapter configuration in the application Admin Console and in the GPM as necessary. See *Creating or Setting Up a Adapter Configuration in the Admin Console* or *Setting Up the Adapter in the GPM*.
5. Configure the business process you are using for the SWIFTNet Server adapter. The business processes that work with SWIFTNet Server adapter include the following:
  - handleSWIFTNetServerSnFRequest
  - handleSWIFTNetInboundCorrelation
  - handleSWIFTNetOutboundCorrelation
  - handleSWIFTNetServerFADelNotif
  - handleSWIFTNetServerFAEvent
  - handleSWIFTNetServerFARequest
  - handleSWIFTNetServerFASnFDelNotif
  - handleSWIFTNetServerFASnFRequest
  - handleSWIFTNetServerRequest
  - handleSWIFTNetServerSnFDelNotif
  - handleSWIFTNetServerFASnFEvent
  - handleSWIFTNetSnFinboundCorrelation
  - handleSWIFTNetSnFOutboundCorrelation
6. Define the **HTTP Listen Port** in the SWIFTNet HTTP Server adapter instance, which should have the same value as the **GIS HTTP Server Adapter Port** defined in the SWIFTNet Server adapter configuration.
7. Specify field settings in the business process. See *Business Process Example*.

## Configuring the SWIFTNet Server Adapter

1. Select **Deployment > Services > Configuration**.
2. Search for SWIFTNet Server adapter or select it from the list and click **Go!**
3. Click **Edit**.
4. Specify field settings in the Admin Console (*Creating or Setting Up a Adapter Configuration in the Admin Console*)—alternatively you can specify field settings in the GPM (*Setting Up the Adapter in the GPM*), but you will need to access the adapter instance through the Admin console to enable the instance (as described in step 5).

**Note:** Specify failover processing to ensure that failover is supported if a SAG connection fails by configuring **Active-Active Configuration**.

**Note:** For specific instructions on configuring an input channel, see *SWIFT Input Channel*.

5. After configuring the SWIFTNet Server adapter in the Admin Console, click the **Enable Service for Business Process** check box on the Confirm page to enable the instance.
6. Once the SWIFTNet Server adapter is configured and saved, click the **Enabled** check box on the Services Configuration page. This starts the SWIFTNet MEFG Server.
7. On the Confirm page, verify that the **Enable Service for Business Processes** check box is selected to enable the adapter instance.

You must specify field settings in the application, using the Admin Console and the GPM.

## Creating or Setting Up a Adapter Configuration in the Admin Console

Use the field definitions in the following table to create a new configuration of the SWIFTNet Server adapter, or to set up the configuration provided with the application. Some fields are available in both the Admin Console and in the GPM.

**Note:** The business entities (accessible through the Business Entities wizard as part of the SWIFTNet Server adapter configuration) are shared by both RA1 and RA2. The Business Entities wizard enables you to add multiple entities.

Field	Description
Name	Unique and meaningful name for the service configuration. Required.
Description	Meaningful description for the service configuration, for reference purposes. Required.
Select a Group	<p>Select one of the options:</p> <ul style="list-style-type: none"> <li>• None – Do not include the configuration in a service group at this time.</li> <li>• Create New Group – Enter a unique name for a new group, which will be created with this configuration. (You can then add other services to the group as well.)</li> <li>• Select Group – If service groups already exist for this service type, they are displayed in the list. Select a group from the list.</li> </ul> <p><b>Note:</b> Only select group if this adapter is clustered in a group. See <i>Managing Services and Services</i>.</p>
GIS Server IP	<p>The callback IP of the application for the SWIFTNet MEFG Server. Required.</p> <p><b>Note:</b> The default value is the IP address of the machine where the application is installed.</p>



Field	Description
GIS HTTP Server Adapter Port	<p>This is the listening port for the SWIFTNet HTTP Server Adapter. Required. The default populated value is the instance port number of the application instance plus 53. For example, if the application instance port is 34600, the listening port populated by default is 34653.</p> <p><b>Note:</b> The HTTP Server adapter functions between the SWIFTNet Server adapter and the SWIFTNet MEFG Server. For an SSL connection, this value should be server name because the certificate is made with the server name.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
MEFG SWIFTNet IP	The IP address of the SWIFTNet MEFG Server. Required.
MEFG SWIFTNet Port	The port of the SWIFTNet MEFG Server. Required.
CLA2Client Listening Port	<p>The listening port used by the client command adapter (CLA2Client) running along the SWIFTNet MEFG Server. Required.</p> <p><b>Note:</b> This port listens for requests to start and stop the SWIFTNet MEFG Server.</p>
MEFG SWIFTNet Home	The home directory of the SWIFTNet MEFG Server. Required.
Use SSL for HTTP communication with MEFG SWIFTNet	Whether to enable Secure Sockets Layer (SSL) over HTTP communication between the application and the SWIFTNet MEFG Server. Valid values are False (default) and True.
Cipher Strength	<p>Specifies the strength of the algorithms (cipher suites) used to encrypt data. Valid values are:</p> <ul style="list-style-type: none"> <li>• STRONG - Required if Use SSL is Must</li> <li>• ALL - All cipher strengths are supported</li> <li>• WEAK - Often required for international trade, because government regulations prohibit STRONG encryption from being exported</li> </ul> <p>Default is ALL. Required if Use SSL for HTTP communication with MEFG SWIFTNet is checked.</p>

Field	Description
CA Certificate	Move one or more CA Certificates to the use column. These are the digital security certificates that the SSL server will use to authenticate the client. Optional.
Access Authentication?	<p>Turn on authentication for Command Line Adapter 2?</p> <p><b>Tip:</b> For a secure configuration, enable authentication.</p> <p>Valid values:</p> <ul style="list-style-type: none"> <li>• Yes - Default</li> <li>• No</li> </ul> <p>For more information on the authentication configuration of the Command Line Adapter 2, see <i>Command Line Adapter 2 (Build 5104 or higher)</i>.</p> <p><b>Restriction:</b> The authentication parameters set here must be the same as those in the Command Line Adapter 2 configuration.</p>
System Authentication Certificate	Select the Command Line Adapter 2 authentication certificate that you want to run. Default value: cla2auth
Use SSL for CLA2 communication (Note: User Authentication without SSL will result in a weak security configuration.)	<p>Use SSL to secure the Command Line Adapter 2?</p> <p><b>Tip:</b> For a secure configuration, enable SSL.</p> <p>Valid values:</p> <ul style="list-style-type: none"> <li>• Yes</li> <li>• No - Default</li> </ul> <p>For more information on the SSL configuration of the Command Line Adapter 2, see <i>Command Line Adapter 2 (Build 5104 or higher)</i>.</p> <p><b>Restriction:</b> The SSL parameters set here must be the same as those in the Command Line Adapter 2 configuration.</p>
SSL Public CA Certificate	Select the Command Line Adapter 2 SSL Public CA Certificate for validation. Default value: cla2ssl
Message Partner Client Name	<p>The client message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server client application.</p> <p><b>Note:</b> The Message Partner Client Name must correspond to the Application Interface Message Partner that is defined on the SAG as the client interface for the SWIFTNet MEFG Server.</p>

Field	Description
Message Partner Server Name	<p>The server message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server server application.</p> <p><b>Note:</b> The Message Partner Server Name must correspond to the Application Interface Message Partner that is defined on the SAG as the client interface for the SWIFTNet MEFG Server.</p>
Active-Active Configuration	<p>Enables you to set up active-active configuration using two separate instances of the Remote API (RA), RA1 and RA2. Each RA should be configured to point to a different SAG to support failover processing. Possible values are True and False (default). Required.</p> <p><b>Note:</b> This parameter specifies whether to support failover if one SAG fails. When this parameter is set to True, you are presented with parameters for both an RA1 Profile and an RA2 Profile. When you are operating in an environment with multiple SAGs configured in active-active mode, setting this parameter enables you to define an alternate RA connection to a secondary SAG for failover support.</p>
SNL Endpoint (for Store and Forward only)	<p>The SNL endpoint used to receive data from SnF queues (for example, <code>snl_sft</code>). Optional—complete only if using store and forward processing.</p> <p><b>Note:</b> You must define endpoints on the SAG to route the InterAct messages to the correct application interface. If you are using store-and-forward, an extra endpoint is required to route messages coming from the store-and-forward queue (you can use the default endpoint for store-and-forward, <code>snl_sft</code>).</p>
SnF Monitoring Interval (in seconds)	<p>The store and forward monitoring interval (in seconds). Optional.</p> <p><b>Note:</b> This parameter enables you to indicate the interval that you want the SWIFTNet MEFG Server to check on the queue status. The SWIFTNet MEFG Server sets a timer to send the <code>GetSnFStatusRequest</code> message based on the value you enter.</p>
Return Signature List	<p>Whether you want your own signature returned. Valid values are False (default - normal Crypto is used) and True. Optional for T-Copy and Y-Copy implementation.</p>

Field	Description
Use Input Channel (for InterAct Store and Forward only)	<p>Whether to use the input channel with this adapter. Valid values are False (default) and True. You have to select True if you are using an input channel. Required.</p> <p><b>Note:</b> Used for InterAct store-and-forward only. If you configure this parameter, the SWIFTNet MEFG Server opens the Input Channel automatically during the startup (when the SWIFTNet Server Adapter is enabled). This Input Channel remains open until the SWIFTNet MEFG Server is shut down (or the SWIFTNet Server Adapter is disabled). During this time, you still have an option to send message using the input channel or without the input channel. All you need to do is to indicate this by using this parameter in SWIFTNet Client service.</p>
SWIFTNet RA	<p>The absolute path of the RA1 installation directory for RA1 SWIFTNet. Required. For example, <code>/SWIFTAlliance/RA</code>.</p> <p><b>Note:</b> This parameter specifies where to pick up the remote API and execute to SAG.</p>
Config	<p>The relative path of the RA1 instance configuration directory (relative to the RA installation directory). Required. For example, <code>RA1/cfg</code>.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Bin	<p>This is added to the PATH environment variable to contain the SWIFTNet MEFG Server binaries. Possible value is bin. Required.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>

Field	Description
Lib	<p>This is added to the library path environment variable. Possible value is lib. Required.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Category	<p>This is the category of RA. Possible values are:</p> <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> <p>Required.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Delivery Notification	<p>Determines whether the RA1 server is handling a delivery notification. Possible values are True and False (default). Optional. This is used for a FileAct get.</p>
Delivery Notification DN	<p>Distinguished name of the responder of the delivery notification. Optional.</p>
Request Type of Del. Notifn	<p>Request type of the delivery notification. This is used for a FileAct Get. Required.</p>
Send Del. Notifn before Backend Processing	<p>Indicates if the server will send a delivery notification before the internal process is executed. Required.</p>
Event Status Tracking	<p>Indicates if the server requires all the FileAct Event statuses to be returned. Valid values are:</p> <ul style="list-style-type: none"> <li>• Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>• Full (all statuses are returned)</li> </ul> <p>Required.</p>

Field	Description
SWIFTNet RA	<p>The absolute path of the RA2 installation directory for RA2 SWIFTNet. Required (based on Active-Active configuration). For example, <code>/SWIFTAlliance/RA</code>.</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p>
Config	<p>The relative path of the RA2 instance configuration directory (relative to the RA2 installation directory). Required (based on Active-Active configuration). For example, <code>/RA2/cfg</code>.</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Bin	<p>This is added to the PATH environment variable to contain the SWIFTNet MEFG Server binaries. Required (based on Active-Active configuration).</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Lib	<p>This is added to the library path environment variable. Required (based on Active-Active configuration).</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>

Field	Description
Category	<p>This is the category of RA2. Possible values are:</p> <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> <p>Required (based on Active-Active configuration).</p> <p><b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.</p> <p><b>Note:</b> If you are using the SWIFTNet Server adapter in your current installation, prior to installing a new version of the Standards Library, you need to note the value you have configured for this parameter. This parameter may be overwritten during the upgrade process (replaced with the default value). If this parameter is overwritten, you need to restore it to the original value after the upgrade process is complete.</p>
Delivery Notification	Determines whether the RA2 server is handling a delivery notification. Possible values are True and False (default). Optional. This is used for a FileAct get.
Delivery Notification DN	Distinguished name of the responder of the delivery notification. Optional.
Request Type of Del. Notifn	Request type of the delivery notification. This is used for a FileAct Get. Required.
Send Del. Notifn before Backend Processing	Indicates if the server will send a delivery notification before the internal process is executed. Required.
Event Status Tracking	<p>Indicates if the server requires all the FileAct Event statuses to be returned. Valid values are:</p> <ul style="list-style-type: none"> <li>• Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>• Full (all statuses are returned)</li> </ul> <p>Required.</p>
Input Channel Name	The name of the input channel. Required only if you specified <b>True</b> for <b>Use Input Channel</b> .
Authoriser DN	The authorized distinguished name that will be used to open the input channel. Required only if you specified <b>True</b> for <b>Use Input Channel</b> .
Force Open the Input Channel	Whether to force open the input channel or use normal mode. Valid values are False (use normal mode, which is the default) and True (force the input channel). Required only if you specified <b>True</b> for <b>Use Input Channel</b> .

Field	Description
Max. Resend Attempts	The maximum number of resend attempts allowed before the application automatically sends a Resolve Gap request to SWIFT. The default is 3. Required only if you specified <b>True</b> for <b>Use Input Channel</b> .
Run As User	Identify a user who has permission to run the scheduled activity. You can type the user ID, click the button to select the user ID from the list, and click <b>Save</b> . Optional. <b>Note:</b> You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Use 24 Hour Clock Display	By default, the scheduling wizard displays times using a 12-hour clock (which designates the time in hours as a.m. or p.m.). Use this option to display times using a 24-hour clock. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Do not use schedule	Removes all references to a schedule from the service. If you select this option, you cannot enable the schedule in the future. You must recreate the schedule instead. Use this option only when you do not need a schedule for the service. This is the default option. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Run based on timer	Run the service at a certain time or time interval, such as every 2 hours. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Select Time	Type the time at which you want the Resend Scheduler to run. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.



Field	Description
Run daily	Run the service one or more times every day. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Run based on days of the week	Run the service on certain days of the week, such as every Monday. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Run based on days of the month	Run the service on certain days of the month, such as the 1st or 15th of every month. Optional. <b>Note:</b> We recommend that you set the Resend Scheduler to <b>Run based on timer</b> and set it for one minute under normal usage (that is, every 1 Mins(s)). You must configure the parameters on the Schedule Type page for the Resend Scheduler to work correctly.
Schedule Exclusions	Allows you to add any schedule anomalies (when the Resend Scheduler should not run). <b>Note:</b> We recommend you leave this parameter blank (that is, do not create any schedule exclusions).
Date Exclusions	Allows you to add any date anomalies (any date on which the Resend Scheduler should not run). <b>Note:</b> We recommend you leave this parameter blank (that is, do not create any date exclusions).
New Business Entity	Click <b>add</b> to create a new business entity or click <b>edit</b> to modify an existing entity. <b>Note:</b> You must have at least one business entity created to proceed.

Field	Description
Entity	<p>Identifies the security context to be used. For the client, the business entity is the requester. For the server, the business entity is the responder. Required for each configured entity to access a proprietary SWIFTNet PK1 certificate to set up a valid security context.</p> <p><b>Note:</b> This is the distinguished name created by SWIFT. This parameter is only displayed if you edit an existing Business Entity or add a new Business Entity. The business entities are shared by both the RA1 and RA2 profiles.</p>
UserId	<p>The user identifier for this business entity (to log in to SWIFTNet). Required for each configured entity.</p> <p><b>Note:</b> The UserName is created in SAG (in the Users Module) and must also have a certificate created for it in the SAG. This parameter is only displayed if you edit an existing Business Entity or add a new Business Entity.</p>
Password	<p>The user password for this business entity (to log in to SWIFTNet). Required for each configured entity.</p> <p><b>Note:</b> This password is automatically encrypted. This parameter is only displayed if you edit an existing Business Entity or add a new Business Entity.</p>
Message Queue	<p>The name of the store and forward queue from which to receive messages. Required in Store and Forward mode.</p>
Notification Queue	<p>The Name of the store-and-forward queue to retrieve delivery notifications (optional; if empty, same as Message Queue). Required in Store and Forward mode.</p>
Acquire queue by force	<p>Whether to acquire the queue by force. Valid values are False (default) and True. Required.</p>
Use Default Delivery Notification	<p>Indicates whether to use the default delivery notification configuration on the RA1 page. Required.</p>
Delivery Notification (Del. Notifn)	<p>Indicates whether the sender asked the receiver to send a delivery notification. Optional. Valid values are True (default) or False.</p> <p><b>Note:</b> This parameter is only available when <b>Use Default Delivery Notification</b> is not selected.</p>

Field	Description
Request Type of Del. Notifn	If <b>Delivery Notification (Del. Notifn)</b> is set to True, the value of this parameter is used to request a specific delivery notification message from the remote receiving server application when it returns the delivery notification. Optional. <b>Note:</b> This parameter is only available when <b>Use Default Delivery Notification</b> is not selected.
Reception Directory	The full directory path where the file is received and stored during FileAct Put mode. Required for FileAct. Optional.
Download Directory	The full directory path where the file is picked up and sent to the requestor during FileAct Get mode. Required for FileAct. Optional.
Success Directory	The full directory path that must be specified when using the FileAct #OLDEST_FILE feature. Required for FileAct. Optional.

## Setting Up the Adapter in the GPM

Use the field definitions in the following table to set up the adapter configuration in the GPM:

Field	Description
Active-Active Configuration	Enables you to set up active-active configuration using two separate instances of the Remote API (RA), RA1 and RA2. Each RA should be configured to point to a different SAG to support failover processing. Possible values are True and False (default). Required. <b>Note:</b> This parameter specifies whether to support failover if one SAG fails. When this parameter is set to True, you are presented with parameters for both an RA1 Profile and an RA2 Profile. When you are operating in an environment with multiple SAGs configured in active-active mode, setting this parameter enables you to define an alternate RA connection to a secondary SAG for failover support.
commandLinePort	The listening port used by the client command adapter (CLA2Client) running along the SWIFTNet MEFG Server. Required. <b>Note:</b> This port listens for requests to stop the SWIFTNet MEFG Server.

Field	Description
deliveryNotification	Determines whether the server is handling a delivery notification. Possible values are True and False (default). Optional. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
Description	Error description for the rejected response. Optional. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
downloadDir	The full directory path where the file is picked up and sent to the requestor during FileAct Get mode. Required for FileAct.
Info	Error information for the rejected response. Optional. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
interfaceMode	SWIFTNet message type. Valid values are InterAct or FileAct. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
localServerAddress	The callback IP of the application for the SWIFTNet MEFG Server. Required.
localServerPort	The is the listening port for the SWIFTNet HTTP Server adapter. Required. <b>Note:</b> The HTTP Server adapter functions in between the SWIFTNet Server adapter and the SWIFTNet MEFG Server.
messageID	Message identifier for the incoming message. Required. <b>Note:</b> This is a unique identifier. This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
messagePartnerClient Name	The client message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server client application.
messagePartnerServer Name	The server message partner name that the SNL server application recognizes for the SWIFTNet MEFG Server server application. <b>Note:</b> The Message Partner Server Name must correspond to the Application Interface Message Partner that is defined on the SAG as the client interface for the SWIFTNet MEFG Server.
RA1Bin	This is added to the PATH environment variable. Possible value is bin. Required.

Field	Description
RA1Category	This is the category of RA. Possible values are: <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> Required.
RA1Config	The relative path of the RA1 instance configuration directory (relative to the RA installation directory). Required.
RA1deliveryNotification	Determines whether the RA1 server is handling a delivery notification. Possible values are True and False (default). Optional.
RA1deliveryNotification DN	Distinguished name of the responder of the delivery notification. Optional.
RA1deliverynotification RT	Request type of the delivery notification. This is used for a FileAct Get. Required.
RA1eventstatusTracking	Indicates if the server requires all the FileAct Event statuses to be return. Valid values are: <ul style="list-style-type: none"> <li>• Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>• Full (all statuses are returned)</li> </ul> Required.
RA1Lib	This is added to the library path environment variable. Possible value is lib. Required.
RA1sendDNb4bkend Process	Indicates if the server will send a delivery notification before the internal process is executed. Required.
RA1Swiftnethome	The home directory of the SWIFTNet MEFG Server. Required. <b>Note:</b> This is an absolute path location. This parameter specifies where to pick up the remote API and execute to SAG.
RA2Bin	This is added to the PATH environment variable. Possible value is bin. Required. <b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.
RA2Category	This is the category of RA. Possible values are: <ul style="list-style-type: none"> <li>• RA (SNL facade library to access an SAG)</li> <li>• SNL (a native SNL interface)</li> <li>• DEFAULT (default set for the RA1 instance)</li> </ul> Required. <b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True.
RA2Config	The relative path of the RA2 instance configuration directory (relative to the RA installation directory). Required.

Field	Description
RA2deliveryNotification	Determines whether the RA2 server is handling a delivery notification. Possible values are True and False (default). Optional.
RA2deliveryNotification DN	Distinguished name of the responder of the delivery notification. Optional.
RA2deliverynotification RT	Request type of the delivery notification. This is used for a FileAct Get. Required.
RA2eventstatusTracking	Indicates if the server requires all the FileAct Event statuses to be return. Valid values are: <ul style="list-style-type: none"> <li>Minimal (only Completed, Rejected, Duplicated statuses will be returned)</li> <li>Full (all statuses are returned)</li> </ul> Required.
RA2Lib	This is added to the library path environment variable. Possible value is lib. Required.
RA2sendDNb4bkend Process	Indicates if the server will send a delivery notification before the internal process is executed. Required.
RA2Swiftnethome	The home directory of the RA2. Required. <b>Note:</b> This parameter is only displayed if <b>Active-Active Configuration</b> is set to True. <b>Note:</b> This is an absolute path location. This parameter specifies where to pick up the remote API and execute to SAG.
receptionDir	The full directory path where the file is received and stored during FileAct Put mode. Required for FileAct.
remoteServerAddress	The IP address of the SWIFTNet MEFG Server. Required.
remoteServerPort	The port of the SWIFTNet MEFG Server. Required.
SnF	Indicates if you are using the store-and-forward method. Valid values are True (use Store-and-Forward) and False (default—do not use Store-and-Forward). Required. <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.
snlEndPoint	The SNL endpoint used to receive data from SnF queues (for example, snl_sft). Optional—complete only if using store and forward processing.

Field	Description
Status	<p>The status of the message. Possible values are:</p> <ul style="list-style-type: none"> <li>• Accepted</li> <li>• Rejected</li> <li>• Failed</li> <li>• Duplicated</li> </ul> <p>Required.  <b>Note:</b> This is a BPML parameter used by the SWIFTNet Server adapter to construct the response back to the SWIFTNet MEFG Server.</p>
successDir	<p>The full directory path that must be specified when using the FileAct #OLDEST_FILE feature. Required for FileAct.</p>
swiftnetServerHome Directory	<p>The home directory where the SWIFTNet MEFG Server is installed. Required.</p>

## Business Process Example

The service part of the SWIFTNet Server adapter that is used in the business process is bootstrapped when the SWIFTNet MEFG Server posts the request through the URI defined in the HTTP Server adapter. For more information about the HTTP Server adapter, see *HTTP Server Adapter*.

## Renewing a Certificate

You can create a business process and schedule it to be executed at an interval of three months. You only need to pass in the distinguished name that is specified in the SWIFTNet Server Adapter Business Entities. When the request is passed to the SWIFTNet MEFG Server, it looks up the user identifier and the encrypted password in the configuration file. The SWIFTNet MEFG Server then performs an `initRequest` and `CreateSecurityContext` to open the certificate.

```

<operation>

  <participant
name="SWIFTNetClientService"/>

  <output
message="renewSecurityContext">

    <assign
to="renewDN">o=yourDN,o=swift</assign>

    <assign
to="." from="*"/>

  </output>

  <input
message="testing">

    <assign
to="." from="*"/>

  </input></operation>

```

## Interact Business Process Without Store-and-Forward Processing

The following business process example (in which the service part of the SWIFTNet Server adapter as part of InterAct processing) is used if you are not using store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerRequest business process.

```
<process name="handleSWIFTNetServerRequest">
  <sequence>
    <operation
name="set user token">
      <participant
name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign
to="USER_TOKEN">admin</assign>
        <assign
to="." from="*" />
      </output>
      <input
message="inmsg">
        <assign to="." from="*" />
      </input>
    </operation>
    <operation
name="SoapIn">
      <participant name="SOAPInbound"/>
      <output
message="output">
        <assign to="." from="*" />
        <assign
to="bootstrap">>false</assign>
        <assign
to="SOAP_INTERMEDIATE_NODE">>false</assign>
      </output>
      <input
message="input">
        <assign to="." from="*" />
      </input>
    </operation>
    <operation>
      <participant
name="SWIFTNetServerAdapter"/>
      <output
message="handleServerRequest">
        <assign
to="." from="*" />
      </output>
      <input
message="testing">
        <assign to="." from="*" />
      </input>
    </operation>
    <!--
internal processing by invoking a subprocess -->
    <!--
business-specific processing that will return a response for InterAct
-->
    <operation>
      <participant
name="InvokeSubProcessService"/>
      <output
message="Xout">
```



```

        <assign to="INVOKE_MODE">SYNC</assign>
        <assign
to="." from="*" />
    </output>
    <input
message="Xin">
        <assign to="." from="*" />
    </input>
</operation>
<!--
this is to construct the server response message back to GIS Server
application -->
    <operation>
        <participant
name="SWIFTNetServerAdapter" />
        <output
message="handleServerResponse">
            <assign
to="." from="*" />
            <assign to="interface"
from="SwiftServerRequest/interface/text()" />
            <assign
to="messageID" from="SwiftServerRequest/messageID/text()" />
            <assign
to="Status">Accepted</assign>
            <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()" />
            <assign
to="SnF" from="SwiftServerRequest/SnF/text()" />
        </output>
        <input
message="testing">
            <assign to="." from="*" />
        </input>
    </operation>
    <operation
name="SoapOut">
        <participant name="SOAPOutbound" />
        <output
message="output">
            <assign to="." from="*" />
            <assign
to="SOAP_MODE">respond</assign>
        </output>
        <input
message="input">
            <assign to="." from="*" />
        </input>
    </operation>
    <assign
to="doc-has-headers">>true</assign>
    <operation
name="HttpResponse">
        <participant name="HttpRespond" />
        <output
message="Xout">
            <assign to="." from="*" />
        </output>
        <input
message="Xin">
            <assign to="." from="*" />
        </input>
    </operation>
</onFault>
<!--
On Fault, we will clear PrimDoc, construct Rejected response and
soap-envelope it -->

```

```

    <sequence>
      <operation
name="ReleasePrimDoc">
        <participant
name="ReleaseService"/>
        <output message="outmsg">
          <assign
to="TARGET"/>/ProcessData/PrimaryDocument</assign>
          <assign
to="." from="*"/>
        </output>
        <input
message="inmsg"/>
      </operation>
      <operation>
        <participant
name="SWIFTNetServerAdapter"/>
        <output
message="handleServerResponse">
          <assign
to="." from="*"/>
          <assign to="interface"
from="SwiftServerRequest/interface/text()"/>
          <assign
to="messageID" from="SwiftServerRequest/messageID/text()"/>
          <assign
to="Status">Rejected</assign>
          <assign
to="Description">Unable to get the Server Response</assign>
          <assign
to="Info">Failure in getting the Server Response</assign>
          <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()"/>
          <assign
to="SnF" from="SwiftServerRequest/SnF/text()"/>
        </output>
        <input
message="testing">
          <assign to="."
from="*"/>
        </input>
      </operation>
      <operation
name="SoapOut">
        <participant name="SOAPOutbound"/>
        <output
message="output">
          <assign to="."
from="*"/>
          <assign to="SOAP_MODE">respond</assign>
        </output>
        <input
message="input">
          <assign to="." from="*"/>
        </input>
      </operation>
      <assign
to="doc-has-headers">true</assign>
      <operation
name="HttpResponse">
        <participant
name="HttpRespond"/>
        <output message="Xout">
          <assign
to="." from="*"/>
        </output>
      </input>

```

```

message="Xin">
  <assign to="." from="*" />
  </input>
  </operation>
</sequence>
</onFault>
</sequence>
</process>

```

## InterAct Business Process With Store-and-Forward Processing

The following business process example demonstrates the service part of the SWIFTNet Server adapter being used as part of InterAct processing if you are using store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerSnFRequest business process.

```

<process name="handleSWIFTNetServerSnFRequest">

  <rule name="IsAuthNotification">
    <condition>SwiftServerRequest/AuthResponse
= 'TRUE'</condition>
  </rule>

  <sequence>
    <operation name="set
user token">
      <participant name="SetUserToken"/>

      <output message="SetUserTokenMessage">

        <assign to="USER_TOKEN">admin</assign>

        <assign to="." from="*" />
      </output>

      <input message="inmsg">
        <assign
to="." from="*" />
      </input>

    </operation>
    <operation name="SoapIn">

      <participant name="SOAPInbound"/>

      <output message="output">
        <assign
to="." from="*" />
        <assign to="bootstrap">>false</assign>

        <assign to="SOAP_INTERMEDIATE_NODE">>false</assign>

      </output>
      <input message="input">

        <assign to="." from="*" />
      </input>

    </operation>
  </operation>

  <participant name="SWIFTNetServerAdapter"/>

  <output message="handleServerRequest">

```

```

        <assign to="." from="*" />
    </output>

    <input message="testing">
        <assign
to="." from="*" />
    </input>

</operation>
<choice name="AddToMailbox">

    <select>
        <case ref="IsAuthNotification"
negative="true" activity="Mailbox Add Service" />

    </select>

    <!-- internal processing
for SnF is to put into a Mailbox so that it can bootstrap internal
business process later-->
    <!-- Mailbox
path is based on SwiftServerRequest/responderDN/requestorDN/for
InterAct -->
    <operation name="Mailbox
Add Service">
        <participant name="MailboxAdd" />

        <output message="AddRequest">

            <assign to="." from="*" />

            <assign to="MailboxPath" from="concat('/',
SwiftServerRequest/responderDN/text(), '/',
SwiftServerRequest/requestorDN/text())" />

            <assign to="ContentType">ascii</assign>

        </output>
        <input message="inmsg">

            <assign to="AddResults" from="*" />

        </input>
    </operation>

</choice>
<operation>

    <participant name="SWIFTNetServerAdapter" />

    <output message="handleServerResponse">

        <assign to="." from="*" />

    <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

        <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

        <assign to="Status">Accepted</assign>

        <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()" />

        <assign to="SnF" from="SwiftServerRequest/SnF/text()" />

    </output>
    <input message="testing">

```

```

        <assign to="." from="*" />
    </input>

</operation>
<operation name="SoapOut">

    <participant name="SOAPOutbound" />

    <output message="output">
        <assign
to="." from="*" />
        <assign to="SOAP_MODE">respond</assign>

    </output>
    <input message="input">

        <assign to="." from="*" />
    </input>

</operation>
    <assign to="doc-has-headers">true</assign>

<operation name="HttpResponse">
<participant name="HttpRespond" />

    <output message="Xout">
        <assign
to="." from="*" />
    </output>

    <input message="Xin">
        <assign
to="." from="*" />
    </input>

</operation>
<onFault>

<sequence>
    <operation name="ReleasePrimDoc">

        <participant name="ReleaseService" />

        <output message="outmsg">

            <assign to="TARGET">/ProcessData/PrimaryDocument</assign>

            <assign to="." from="*" />

        </output>
        <input message="inmsg" />

    </operation>
    <operation>

        <participant name="SWIFTNetServerAdapter" />

        <output message="handleServerResponse">

            <assign to="." from="*" />

            <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

            <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

```

```

        <assign to="Status">Rejected</assign>
        <assign to="Description">Unable to get the Server Response</assign>
        <assign to="Info">Failure in getting the Server Response</assign>
        <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()"/>
        <assign to="SnF" from="SwiftServerRequest/SnF/text()"/>
    </output>
    <input message="testing">
        <assign to="." from="*" />
    </input>
</operation>
<operation name="SoapOut">
<participant name="SOAPOutbound"/>
    <output message="output">
        <assign to="." from="*" />
    <assign to="SOAP_MODE">respond</assign>
    </output>
    <input message="input">
        <assign to="." from="*" />
    </input>
</operation>
<assign to="doc-has-headers">>true</assign>
<operation name="HttpResponse">
    <participant name="HttpRespond"/>
    <output message="Xout">
        <assign to="." from="*" />
    </output>
    <input message="Xin">
        <assign to="." from="*" />
    </input>
</operation>
</sequence>
</onFault>
</sequence>
</process>

```

## Fileact Business Process Without Store-and-Forward Processing

The following business process example shows the service part of the SWIFTNet Server adapter as part of FileAct processing without using store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerRequest business process.

```

<process name="handleSWIFTNetServerFARequest">
  <sequence>
    <operation
name="set user token">
      <participant
name="SetUserToken"/>
      <output message="SetUserTokenMessage">
        <assign
to="USER_TOKEN">admin</assign>
        <assign
to="." from="*" />
      </output>
      <input
message="inmsg">
        <assign to="." from="*" />
      </input>
    </operation>
    <operation
name="SoapIn">
      <participant name="SOAPInbound"/>
      <output
message="output">
        <assign to="." from="*" />
        <assign
to="bootstrap">>false</assign>
        <assign
to="SOAP_INTERMEDIATE_NODE">>false</assign>
      </output>
      <input
message="input">
        <assign to="." from="*" />
      </input>
    </operation>
    <operation>
      <participant
name="SWIFTNetServerAdapter"/>
      <output
message="handleServerRequest">
        <assign
to="." from="*" />
      </output>
      <input
message="testing">
        <assign to="." from="*" />
      </input>
    </operation>
    <!--
this is to construct the server response message back to GIS Server
application -->
    <operation>
      <participant
name="SWIFTNetServerAdapter"/>
      <output
message="handleServerResponse">
        <assign
to="." from="*" />
        <assign to="interfaceMode"
from="SwiftServerRequest/interfaceMode/text()" />
        <assign
to="messageID" from="SwiftServerRequest/messageID/text()" />
        <assign
to="Status">Accepted</assign>
        <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()" />

```

```

    <assign
to="SnF" from="SwiftServerRequest/SnF/text()"/>
  </output>
  <input
message="testing">
  <assign to="." from="*" />
  </input>
</operation>
<operation
name="SoapOut">
  <participant name="SOAPOutbound" />
  <output
message="output">
  <assign to="." from="*" />
  <assign
to="SOAP_MODE">respond</assign>
  </output>
  <input
message="input">
  <assign to="." from="*" />
  </input>
</operation>
<assign
to="doc-has-headers">true</assign>
  <operation
name="HttpResponse">
  <participant name="HttpRespond" />
  <output
message="Xout">
  <assign to="." from="*" />
  </output>
  <input
message="Xin">
  <assign to="." from="*" />
  </input>
</operation>
  <onFault>
  <!--
On Fault, we will clear PrimDoc, construct Rejected response and
soap-envelope it -->
  <sequence>
  <operation
name="ReleasePrimDoc">
  <participant
name="ReleaseService" />
  <output message="outmsg">
  <assign
to="TARGET">/ProcessData/PrimaryDocument</assign>
  <assign
to="." from="*" />
  </output>
  <input
message="inmsg" />
  </operation>
  <operation>
  <participant
name="SWIFTNetServerAdapter" />
  <output
message="handleServerResponse">
  <assign
to="." from="*" />
  <assign to="interfaceMode"
from="SwiftServerRequest/interfaceMode/text()"/>
  <assign
to="messageID" from="SwiftServerRequest/messageID/text()"/>
  <assign
to="Status">Rejected</assign>

```



```

    <assign
to="Description">Unable to get the Server Response</assign>
    <assign
to="Info">Failure in getting the Server Response</assign>
    <assign
to="deliveryNotification" from="SwiftServerRequest/deliveryNotification/text()"/>
    <assign
to="SnF" from="SwiftServerRequest/SnF/text()"/>
    </output>
    <input
message="testing">
    <assign to="."
from="*" />
    </input>
    </operation>
    <operation
name="SoapOut">
    <participant name="SOAPOutbound" />
    <output
message="output">
    <assign to="."
from="*" />
    <assign to="SOAP_MODE">respond</assign>
    </output>
    <input
message="input">
    <assign to="." from="*" />
    </input>
    </operation>
    <assign
to="doc-has-headers">true</assign>
    <operation
name="HttpResponse">
    <participant
name="HttpRespond" />
    <output message="Xout">
    <assign
to="." from="*" />
    </output>
    <input
message="Xin">
    <assign to="." from="*" />
    </input>
    </operation>
    </sequence>
    </onFault>
    </sequence>
</process>

```

## FileAct Business Process With Store-and-Forward Processing

The following business process example shows the service part of the SWIFTNet Server adapter used as part of FileAct processing with store-and-forward processing:

**Note:** This business process is from the handleSWIFTNetServerFASnFRequest business process.

```

<process name="handleSWIFTNetServerFASnFRequest">
<rule name="UndefinedCopyOrForceReject">
    <condition>SwiftServerRequest/AuthRequest = 'N' or
SwiftServerRequest/FileInfoForceMode =
'Rejected'</condition>
    </rule>

```

```

<rule name="AuthorizationNeeded">
  <condition>SwiftServerRequest/AuthRequest
  = 'Y' and SwiftServerRequest/FileInfoForceMode != 'Refused'</condition>
</rule>
<rule name="ForceRefusal">
  <condition>SwiftServerRequest/FileInfoForceMode = 'Refused'</condition>
</rule>
<sequence>
  <operation name="set user token">
    <participant name="SetUserToken"/>
    <output message="SetUserTokenMessage">
      <assign to="USER_TOKEN">admin</assign>
      <assign to="." from="*" />
    </output>
    <input message="inmsg">
      <assign
to="." from="*" />
    </input>
  </operation>
  <operation name="SoapIn">
    <participant name="SOAPInbound"/>
    <output message="output">
      <assign
to="." from="*" />
      <assign to="bootstrap">>false</assign>
      <assign to="SOAP_INTERMEDIATE_NODE">>false</assign>
    </output>
    <input message="input">
      <assign to="." from="*" />
    </input>
  </operation>
  <operation>
    <participant name="SWIFTNetServerAdapter"/>
    <output message="handleServerRequest">
      <assign to="." from="*" />
    </output>
    <input message="testing">
      <assign
to="." from="*" />
    </input>
  </operation>
  <choice name="NeedAuthorization">
    <select>
      <case ref="AuthorizationNeeded"

```

```

activity="Mailbox Add Service"/>
</select>

<!-- Put into a Mailbox so that it can bootstrap internal
authorization business process later -->

<!-- Mailbox path is based on SwiftServerRequest/recipientDN/requestorDN/
-->
<operation name="Mailbox Add
Service">
  <participant name="MailboxAdd"/>

  <output message="AddRequest">

    <assign to="." from="*" />

    <assign to="PrimaryDocument" from="HeaderInfo/@SCIObjectID"/>

    <assign to="MessageName" from="concat('ThirdParty_',
SwiftServerRequest/copySnFReference/text())"/>

    <assign to="MailboxPath" from="concat('/',
SwiftServerRequest/recipientDN/text(), '/',
SwiftServerRequest/requestorDN/text())"/>

    <assign to="ExtractableCount">1</assign>

    <assign to="ContentType">ascii</assign>

  </output>
  <input message="inmsg">

    <assign to="AddResults" from="*" />

  </input>
</operation>

</choice>
<choice name="IsUndefinedCopyOrForceReject">

  <select>
    <case ref="UndefinedCopyOrForceReject"
negative="true" activity="AcceptRequest"/>

    <case ref="UndefinedCopyOrForceReject" activity="RejectRequest"/>
  </select>

  <operation name="AcceptRequest">

    <participant name="SWIFTNetServerAdapter"/>

    <output message="handleServerResponse">

      <assign to="." from="*" />

      <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()"/>

      <assign to="messageID" from="SwiftServerRequest/messageID/text()"/>

      <assign to="Status">Accepted</assign>

      <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()"/>

      <assign to="SnF" from="SwiftServerRequest/SnF/text()"/>

    </output>

```

```

        <input message="testing">
        <assign to="." from="*" />
    </input>
    </operation>

    <sequence name="RejectRequest">

        <operation name="ReleasePrimDoc">
            <participant name="ReleaseService" />
            <output message="outmsg">
                <assign to="TARGET">/ProcessData/PrimaryDocument</assign>
                <assign to="." from="*" />
            </output>
            <input message="inmsg" />
        </operation>
        <operation
name="Form Reject Response">
            <participant
name="SWIFTNetServerAdapter" />
            <output
message="handleServerResponse">
                <assign to="." from="*" />
                <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />
                <assign to="messageID" from="SwiftServerRequest/messageID/text()" />
                <assign to="Status">Rejected</assign>
                <assign to="Description">Copy Profile is undefined
or Responder forced to reject</assign>
                <assign to="Info">Unable to determine copy mode or
FileInfo force responder's rejection</assign>
                <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()" />
                <assign to="SnF" from="SwiftServerRequest/SnF/text()" />
            </output>
            <input message="testing">
                <assign to="." from="*" />
        </input>
    </operation>

    </sequence>
</choice>

    <operation name="SoapOut">
        <participant
name="SOAPOutbound" />
        <output message="output">
            <assign to="." from="*" />

```

```

<assign to="SOAP_MODE">respond</assign>
  </output>
  <input message="input">
    <assign to="." from="*" />
  </input>
</operation>
  <assign to="doc-has-headers">>true</assign>
<operation name="HttpResponse">
<participant name="HttpRespond"/>
  <output message="Xout">
    <assign
to="." from="*" />
  </output>
  <input message="Xin">
    <assign
to="." from="*" />
  </input>
</operation>
  <choice name="IsThirdPartyForceRefusal">
    <select>
      <case ref="ForceRefusal"
activity="InvokeForceRefusalProcess"/>
    </select>
    <operation name="InvokeForceRefusalProcess">
      <participant name="InvokeBusinessProcessService"/>
      <output message="Invoke_In">
        <assign to="." from="*" />
      </output>
      <assign to="INVOKE_MODE">ASYNC</assign>
      <assign to="WFD_NAME">SWIFTNet3rdPartyClientForceRefusal</assign>
      </output>
      <input message="Invoke_Out">
        <assign to="." from="*" />
      </input>
    </operation>
  </choice>
  <onFault>
    <sequence>
      <operation name="ReleasePrimDoc">
        <participant name="ReleaseService"/>
        <output message="outmsg">
          <assign to="TARGET">/ProcessData/PrimaryDocument</assign>
          <assign to="." from="*" />
        </output>
      </operation>
    </sequence>
  </onFault>

```

```

</output>
  <input message="inmsg"/>
</operation>
<operation>

  <participant name="SWIFTNetServerAdapter"/>

  <output message="handleServerResponse">

    <assign to="." from="*" />

    <assign to="interfaceMode" from="SwiftServerRequest/interfaceMode/text()" />

    <assign to="messageID" from="SwiftServerRequest/messageID/text()" />

    <assign to="Status">Rejected</assign>

    <assign to="Description">Unable to get the Server Response</assign>

    <assign to="Info">Failure in getting the Server Response</assign>

    <assign to="deliveryNotification" from="SwiftServerRequest/
deliveryNotification/text()" />

    <assign to="SnF" from="SwiftServerRequest/SnF/text()" />

  </output>
  <input message="testing">

    <assign to="." from="*" />

  </input>
</operation>

<operation name="SoapOut">
<participant name="SOAPOutbound"/>

  <output message="output">

    <assign to="." from="*" />

    <assign to="SOAP_MODE">respond</assign>

  </output>
  <input message="input">

    <assign to="." from="*" />

  </input>
</operation>

<assign to="doc-has-headers">true</assign>

<operation name="HttpResponse">

  <participant name="HttpRespond"/>

  <output message="Xout">

    <assign to="." from="*" />

  </output>
  <input message="Xin">

    <assign to="." from="*" />

```

```

        </input>
        </operation>

    </sequence>
</onFault>

</sequence>
</process>

```

## Parameters Passed From Business Process to Adapter

The following table contains the parameters passed from the business process to the SWIFTNet Server adapter:

Parameter	Description
messageID	Message identifier for the incoming message. Required.
interfaceMode	This is the SWIFTNet interface. Possible values are InterAct (default) or FileAct. Required.
deliveryNotification	Determines whether the server is handling a delivery notification. Valid values are True and False. Required.
SnF	Indicates whether you are using the store-and-forward method. Valid values are True (use store-and-forward) and False (do not use store-and-forward—this is the default). Required.
Status	The status of the message. Possible values are: <ul style="list-style-type: none"> <li>• Accepted</li> <li>• Rejected</li> <li>• Failed</li> <li>• Duplicated</li> </ul> Required.
Description	Description of the message. Optional. <b>Note:</b> Only necessary when there is an error message.
Information	Information about the message. Optional. <b>Note:</b> Only necessary when there is an error message.

## Enabling SWIFTNet Document Tracking

You need to enable document tracking in the system business process you are using for the SWIFTNet Server adapter—`handleSWIFTNetServerRequest` (if you are not using store-and-forward processing) or `handleSWIFTNetServerSnFRequest` (if you are using store-and-forward processing)—so the system can track the document during the process. In the business process text editor, you can easily enable SWIFTNet document tracking in the application by selecting the **Document Tracking** check box on the Process Levels page. Set the following options as needed and leave the rest of the business process parameters as the defaults:

- On the **Deadline Settings** page, set the deadline and notification options, if necessary.
- On the **Life Span** page, set the life span, if necessary.

## SWIFTNet Header Info Support

With SWIFTNet version 6.1, SWIFT introduced a new FileAct header field (HeaderInfo) to contain key summary information related to the file. The presence of Sw:HeaderInfo within the request is an indication to invoke the feature to validate the Sw:HeaderInfo. With SWIFTNet version 6.3, SWIFT will perform stronger central validation on the HeaderInfo fields for FileAct.

Once a service is activated for the validation, SWIFT checks the HeaderInfo contents (that is, presence, syntax, and semantic). SWIFT rejects files with HeaderInfo contents that either do not pass this validation, or that do not use the HeaderInfo field according to the rules defined for the service.

### Header Info Support on the Client (the Application as Requestor)

As a requestor, the HeaderInfo is only allowed on the FileAct Put Request message. To specify the Header Info information, you must set the HeaderInfo parameter in the SWIFTNet Client Service accordingly. Please refer to *SWIFTNet Client Service* documentation for more details.

The application then validates the HeaderInfo if there is a matching **Request Type** profile in the SWIFTNet Service Profile before sending out the HeaderInfo. Therefore, when you specify HeaderInfo during sending, you must configure the **Request Type** profile in the SWIFTNet Service Profile. You can configure the HeaderInfo as a mandatory or an optional parameter; however, if there is no matching profile, this request is forbidden.

**Note:** Please do not include <Sw:HeaderInfo> tag when specifying the value of HeaderInfo parameter. This <Sw:HeaderInfo> tag is automatically included during the process.

### Header Info Support on the Server (the Application as Responder)

As a responder, the HeaderInfo is only allowed on the FileAct Get Response message. When the FileAct Get request is received, the responder checks to see if there is a HeaderInfo file located in the same directory as the download file. The HeaderInfo file must have the same name as the logical filename specified in the request except with an additional filename extension (.hdr). For example, if the logical filename is **payload.txt**, the HeaderInfo filename should be **payload.txt.hdr**. Prior to the Get request, the responder is responsible to provide both the download file and the HeaderInfo file in the correct directory.

When the FileAct Get request is received, this HeaderInfo is validated by the application using the SWIFTNet Service Profile. Depending on the Request Type profile in the SWIFTNet Service Profile, the HeaderInfo file can be mandatory, optional, or forbidden.

**Note:** Please do not include <Sw:HeaderInfo> tag when specifying the content of HeaderInfo file (\*.hdr). This <Sw:HeaderInfo> tag is automatically included during the process.



## SWIFTNet Service Profile

The HeaderInfo block is optional, except for those services that mandate it. If the HeaderInfo block is not used, it must not be present, and if it is used, it must be validated by the schema.

The SWIFTNet Service Profile enables you to easily port Service Profiles from one application instance to another. This function allows you to associate SWIFTNet Request Type with a Schema for Header Validation. You need to create the SWIFTNet Service Profile and associate the request type with the selected schema. This allows the application to validate the HeaderInfo when it is present in the request.

**Note:** The schema must be saved in application.

The Request Type parameter can accept a wildcard (\*) to be used only at the end of the string. To determine which Service Profile to be used for a particular Request Type, the application uses a best-match policy. For example, if there are two Service Profile defined, for pain.\* and pain.001.\*, and the actual request type is pain.002.001, then the first one will be selected.

Two SWIFTNet Service Profiles are preloaded into application. The **pacs.\*** and **pains.\*** service profiles are associated with the Transaction Count schema and set to **Required for validation**. The Transaction Count and Payment Summary schemas are also preloaded into the application.

You can also import and export SWIFTNet Service Profiles from one application instance to another.

### Creating a SWIFTNet Service Profile

To create a SWIFTNet service profile:

1. From the application **Deployment** menu, select **Adapter Utilities > SWIFTNet Service Profile**.
2. To the right of **Create new SWIFTNet service profile**, click **Go!**.
3. Complete the following parameters and click **Next**:

Parameter	Description
Request Type	Type the request type. A wildcard (*) is only allowed at the end of the string, for example for example, <b>pacs.*</b> or <b>pacs.001.*</b> . Required.
Schema Name	Select the schema used to validate the header information for this request type. Required.
Validation Type	Select whether validation is mandatory or should only be used if header information is specified. Optional. Valid values are: <ul style="list-style-type: none"><li>• Validates only if Header Information is specified (default)</li><li>• Validation of Header Information is required</li></ul>

4. Click **Finish** to save the service profile.

### Searching for a SWIFTNet Request Type

To edit or delete a SWIFTNet request type, you must first locate the appropriate request type. You can locate a specific request type in two ways:

- Search for the request type by name.
- Select the request type from an alphabetical list.

Searching for the request type by name is more precise and provides fewer results. Searching from an alphabetical list will result in a list of all request type or all types beginning with a specified letter or digit.

Once you search for the request type, you can easily edit or delete it from the SWIFTNet Service Profile interface.

### Searching for a Request Type by Name

To search for a request type by name:

1. From the application **Deployment** menu, select **Adapter Utilities > SWIFTNet Service Profile**.
2. In the Search section, type the name of the request type. Case does not matter and you can type part of a name.  
The application returns a list of matches unless no request type meet the criteria you specified.
3. When the list of matches is returned, click **edit** next to the request type you want to modify, or click **delete** next to the request type you want to remove.

### Searching for a Request Type from a List

To select a request type from a list:

1. From the application **Deployment** menu, select **Adapter Utilities > SWIFTNet Service Profile**.
2. In the List section, select one of the following:
  - Alphabetically – Select **All** and click **Go!**
  - Alphabetically – Select a specific letter or digit (0 - 9) and click **Go!**The application returns a list of matches unless no request type meet your criteria.
3. When the list of matches is returned, click **edit** next to the request type you want to modify, or click **delete** next to the request type you want to remove.

### Exporting and Importing a SWIFTNet Service Profile

The application Import/Export feature enables you to save time and increase the accuracy of duplicating supported resources on different environments that are set up for unique purposes. To import and export resources from one application environment to another application environment, both environments must be the same version.

## Chapter 39. SWIFT Reconciliation Service

The SWIFT Reconciliation service is responsible for accepting and routing error messages from SWIFT Alliance Access (SAA) to the application. Additionally, it is used for tracking whether messages sent out were accepted or rejected by the SWIFT communication. The service operates on the primary document.

The following table provides an overview of the SWIFT Reconciliation service:

System Name	SWIFT Reconciliation Service
Graphical Process Modeler (GPM) categories)	All Services and EDI/SWIFT
Description	This service is responsible for enabling the application to accept and handle error notifications from SWIFTAlliance Access (SAA).
Business usage	Use this service to track when messages are rejected locally by SWIFTAlliance Access using MQSA (WebSphere MQ Interface for SWIFTAlliance Access) or AFT (Automated File Transfer). The Reconciliation Service can also be used to track the UUMID (User Unique Message ID) by setting a correlation on the document when it is used in conjunction with MQSA.
Usage example	SAA rejects a message, so it needs to be routed to an error handling process and passed back to the application (the originating application).
Preconfigured?	Yes.
Requires third party files?	No third party files are required.
Platform availability	All supported application platforms.
Related services	None
Application requirements	The service does not require anything to run. It is receiving input from SAA.
Initiates business processes?	No
Invocation	A user must have permission to execute the business process that invokes this service to retrieve data through MQ, FTP, AFT, and so forth.
Business process context considerations	None
Returned status values	<ul style="list-style-type: none"> <li>• Error—if no primary document was found, or if the primary document is not in the correct format, an error occurs</li> <li>• Success—if the primary document was found and successfully parsed</li> </ul>
Restrictions	None
Persistence level	Not applicable

Testing considerations	To test this service in MQSA mode, you must know the report format. Also, MQSA mode checks the NAN/PAN status flag in the workflow, so the tester must set either /ProcessData/PrimaryDocument/feedback or /ProcessData/MQ/msgFeedback to emulate the status flag (set to 276 to emulate a NAN; all other values are ignored). To test in AFT mode, the file process is the original message that was sent.
------------------------	---

## How the SWIFT Reconciliation Service Works

The SWIFT Reconciliation service accepts and routes error messages from SWIFT Alliance Access (SAA) to the application. Additionally, it is used for tracking whether messages sent out were accepted or rejected by the SWIFT communication. The service operates on the primary document.

## Implementing the SWIFT Reconciliation Service

You do not need to do anything to implement the SWIFT Reconciliation service.

## Configuring the SWIFT Reconciliation Service

You do not need to do anything to configure the SWIFT Reconciliation service.

## MQ Business Process Example

The following assumptions and preconditions apply to using the SWIFT Reconciliation service with MQ:

- You must set up outbound envelopes to “Expect an acknowledgement.” It is highly recommended that you use a single global control number for all outbound envelopes that are using this feature.
- In the business process that puts the outbound message on the queue, you must:
  - Set the datagram to request a NAN (Negative Acknowledgement) report. You may also set it to request a PAN (Positive Acknowledgement) report, if you wish to reconcile the UUMID.
  - Specify the queue and queue manager that the return report should use.
  - Set the feedback field so that the original message, error code, and (optionally) the UUMID are included in the return report. You do this by setting the feedback to MQFB\_APPL\_FIRST (65536) + 128 (include the message) + 256 (include the error code), or to MQFB\_APPL\_FIRST + 128 + 256 + 64 (include the UUMID). These values correspond to 65920 and 65984, respectively.
- You have already configured an instance of the MQ adapter named **ToSAA**.

This is the BPML for the MQ example business process:

```
<process
name="FromSAA">
  <sequence>

    <operation
name="WebSphere MQ Adapter">
      <participant
name="FromSAA"/>
```

```

        <output message="WebsphereMQInputMessage">
          <assign
to="." from="*"></assign>
          <assign
to="rcv_MQGMO_wait">Yes</assign>
          <assign
to="rcv_MQGMO_waitInterval">10000</assign>
        </output>
        <input
message="inmsg">
          <assign to="." from="*"></assign>
        </input>
      </operation>

<assign to="PrimaryDocument" from="MQ/document/@SCIObjectID"/>

      <operation
name="SWIFT Reconcile">
        <participant
name= "ReconcileSWIFT"/>
        <output message="ReconcileSWIFTInputMessage">
          <assign
to="." from="*"></assign>
          <assign
to="Mode">MQSA</assign>
        </output>
        <input
message="inmsg">
          <assign to="." from="*"></assign>
        </input>
      </operation>

    </sequence>
  </process>

```

## AFT Business Process Example

To use the SWIFT Reconciliation service with AFT, you must:

Configure SAA so that rejected messages are output to a directory that is separate from the output directory for valid messages. Some of the steps below will vary depending on your system configuration; for full details, see the SWIFTAlliance documentation.

1. In the Application Interface application of SWIFTAlliance Workstation, configure an exit point.
  - a. Open the Application Interface application within SWIFTAlliance Workstation.
  - b. From the View menu, select **Exit Point**.
  - c. From the Exit Point menu, select **New**.
  - d. Type a name for the exit point.
  - e. Set the Queue threshold as desired.
  - f. Leave **Assigned to message partner** blank.
  - g. Save the exit point.
2. In the Application Interface application of SWIFTAlliance Workstation, configure a Message Partner for the rejected messages.
  - a. Open the Application Interface application within SWIFTAlliance Workstation.
  - b. From the View menu, select **Message Partner**.

- c. From the Message Partner menu, select **New**.
  - d. Type a name for the message partner.
  - e. For Allowed direction, select **To Message Partner**.
  - f. For Connection method, select **File Transfer**.
  - g. For Session initiation, select **Manual**.
  - h. For Data format, select **RJE**.
  - i. For Parameter file, select **Not Required**.
  - j. For Output path name, enter the desired path for storing the rejected messages.
  - k. Set the **Run output session** parameters as desired.
  - l. Select the **Emission** tab, and select the Exit Point you created above. Leave the other settings as the default.
  - m. Save and enable this message partner.
3. In the Routing application of SWIFTAlliance Workstation, configure the routing point or points for SWIFT messages so that rejected messages are routed to the exit point you're created.

**Note:** SAA must be running in Housekeeping mode to modify routing points.

- a. Open the Routing application within SWIFTAlliance Workstation.
- b. From the View Menu, select **routing point**.
- c. Select the routing point for messages being sent to SAA (this will vary depending on how your system has been configured).
- d. Select the routing rule for message failing validation if one exists. If it doesn't exist, create one. For the routing condition, use **Condition on of Function and Function Result of Validation Error** for this rule.
- e. Right-click the rule, select **Open**, then click the **Action** tab.
- f. For Action On, select **On Source**.
- g. For Action, select **Route To** in the upper box, and select the exit point you created above from the second box.
- h. For Append intervention, select **No Intervention**.
- i. Select the Unit as desired, and save the routing rule.

Once you have configured SAA, you will need to configure your system so that the application can read the output (for example, FTP or direct file system access). This will vary depending on your system. Once that is configured, you can invoke the SWIFT Reconciliation service on the resulting documents similar to what is shown for MQSA above (note that the mode must be set to AFT).

This is the BPML for the AFT example business process:

```

<operation
name="SWIFT Reconcile">
  <participant
name= "ReconcileSWIFT"/>
  <output message="ReconcileSWIFTInputMessage">
    <assign
to="." from="*"></assign>
    <assign
to="Mode">AFT</assign>
  </output>
</input

```

```
message="inmsg">
  <assign to="." from="*"></assign>
</input>
</operation>
```





---

## Chapter 40. Translation Service

The following table provides an overview of the Translation service:

System name	Translation Type
Graphical Process Modeler (GPM) categories	All Services, Translation
Description	Performs translation of the primary document using a specified map, and replaces the primary document with the result of the translation.
Business usage	Performs translation of the primary document within business processes.
Usage example	You want to take positional data from your order system and translate it to variable-length-delimited data so that it can be read by your billing system. Use the application Map Editor to create a map that will translate the incoming data from positional data to variable-length-delimited data. Write a business process that will put the data into the primary document, then start the Translation service. Using the map you created, the service translates the data from positional data to variable-length-delimited, and replaces the old data with newly translated data in the primary document.
Preconfigured?	There is a configuration of the service delivered with the application, but you must configure parameters for it in the GPM.
Requires third party files?	No
Platform availability	All supported application platforms
Related services	No
Application requirements	The map specified in the <code>map_name</code> parameter must be registered with the application and activated. If either of these conditions is not met then the translation will not be performed.
Initiates business processes?	No
Invocation	Runs as part of a business process.

Business process context considerations	<p>The Translation service looks for the following parameters in the business process context. If the service finds them, it uses them during translations where either the input or output is EDI:</p> <ul style="list-style-type: none"> <li>• edi_output_tag_delimiter</li> <li>• edi_output_segment_delimiter</li> <li>• edi_output_element_delimiter</li> <li>• edi_output_sub_element_delimiter</li> <li>• edi_output_repeating_element_delimiter</li> <li>• edi_output_release_character</li> <li>• edi_output_decimal_separator</li> <li>• edi_input_tag_delimiter</li> <li>• edi_input_segment_delimiter</li> <li>• edi_input_element_delimiter</li> <li>• edi_input_sub_element_delimiter</li> <li>• edi_input_repeating_element_delimiter</li> <li>• edi_input_release_character</li> <li>• edi_input_decimal_separator</li> </ul>
Returned status values	<ul style="list-style-type: none"> <li>• Success – Translation was successful.△</li> <li>• Error – Errors were encountered during translation or translation could not be performed. See the Translator report contained in the Business Process Context Status report for further detail.△</li> </ul>
Restrictions	No
Persistence level	None
Testing considerations	<p>The best way to test is within a simple business process where the Translation service is the only operation. After the business process runs, verify the output in the application, and review the translator report for detail on what occurred during the translation.</p>

## How the Translation Service Works

The Translation service translates data in the following file formats:

- Electronic data interchange (EDI)
- Positional
- Variable-length-delimited
- Extensible Markup Language (XML)
- Structured Query Language (SQL)
- Japanese Center for Informatization of Industry (CII)

**Note:** If the input document character encoding is specified in the application, it overrides the encoding specified in the map. The output document content type and character encoding are set based on the information contained in the map.

The Translation service creates a translation report.

## Implementing the Translation Service

To implement the Translation service, complete the following tasks:

1. Activate your license for the Translation service. See *An Overview of Implementing Services*.
2. If you are using a map that has a database on the output side, you must set up a connection to the database that contains the tables you want to access. See *Setting Up a Connection to an External Database*.
3. Create a Translation service configuration. See *Creating a Service Configuration*.
4. Configure the Translation service. See *Configuring the Translation Service*.
5. Use the Translation service in a business process.

## Configuring the Translation Service

To configure the Translation service, you must specify settings for the following fields in the GPM:

Field	Description
Config	Name of the service configuration.
edi_input_decimal_separator	Character used to indicate the decimal point on the input side.
edi_input_element_delimiter	Character used to delimit elements (fields) on the input side.
edi_input_release_character	Character used to quote elements (fields) that contain the delimiter on the input side.
edi_input_repeating_element_delimiter	Character used to delimit repeating elements on the input side.
edi_input_segment_delimiter	Character used to delimit segments on the input side.
edi_input_sub_element_delimiter	Character used to delimit sub-elements on the input side.
edi_input_tag_delimiter	Character used to delimit tags on the input side.
edi_output_decimal_separator	Character used to indicate the decimal point on the output side.
edi_output_element_delimiter	Character used to delimit elements (fields) on the output side.
edi_output_release_character	Character used to quote elements (fields) that contain the delimiter on the output side.
edi_output_repeating_element_delimiter	Character used to delimit repeating elements on the output side.
edi_output_segment_delimiter	Character used to delimit segments on the output side.
edi_output_sub_element_delimiter	Character used to delimit sub-elements on the output side.
edi_output_tag_delimiter	Character used to delimit tags on the output side.

Field	Description
exhaust_input	Whether to execute the map until the Translation service has translated all of the input. Valid values are Yes and No. <b>Note:</b> If your map design is faulty (that is, if the data structure does not match the layout of the map), the data in the input file cannot be properly processed. If a segment is present in the input file it must be defined and active in the map and in the proper sequence. When the translator reads a segment, it tries to match it to the records in the map based on their tag values. If exhaust_input is set to "Yes" the translator attempts to match each segment in the input file to a segment in the map, until it reaches the end of the input file. Conversely, if exhaust_input is set to "No," the translator does not re-invoke the map to continue processing the remaining data in the input file.
map_name	Name of the map used for translation. The map must already be checked in to the application and enabled.
output_report_to_process_data	Whether to output the report to process data. Valid values are: <ul style="list-style-type: none"> <li>• Yes: Output the report to process data.</li> <li>• No: Do no output the report to process data.</li> </ul>
output_to_process_data	Whether the output of the translation should be placed in the process data tree. The output must be XML. Valid values are Yes and No.
useStreams	Whether to support large files (streaming mode). Valid values are Yes (default), No, and blank (which uses default). The default was changed with release 4.1.1, patch 1973. In versions previous to that, the service did not use document streaming by default.
validate_input	Validates the input to the input side of the map. Valid values are Yes and No.
validate_input_against_dtd	Validates the input to the DTD specified in the input document. Valid values are No validation, Validate using a DTD, and Validate using an XML schema.
validate_output	Validates the output to the output side of the map. Valid values are Yes and No.
ErrorIfAllDataNotConsumed	Whether to throw an error if all of the input data is not consumed during the translation (please note that this is valid only if <b>exhaust_input</b> is not set to true). The default is false (do not throw an error if all of the input data is not consumed).

Field	Description
ErrorIfNoOutput	Whether to throw an error if the output document generated by the translation is empty. The default is false (do not throw an error).

### Parameters Passed Through BPML Only

The following parameters can be passed through BPML using an Assign statement. Note that these parameters are not available through the GPM.

Parameter	Description
FromSchema	Used to enable manipulation of a database schema prefix within the SQL Table/View or SQL Statement of a map. This parameter is required when overriding schema names within one or more SQL Statement fields. If the FromSchema and ToSchema parameters are not supplied, then no schema name substitution is performed. <b>Note:</b> The schema search/replace is case-sensitive.

Parameter	Description
ToSchema	<p>Used to enable manipulation of a database schema prefix within the SQL Table/View or SQL Statement of a map.</p> <p><b>Note:</b> The schema search/replace is case-sensitive. If the FromSchema and ToSchema parameters are not supplied, then no schema name substitution is performed. If the ToSchema parameter is supplied and contains a non-empty value, then any matching schema names are changed at translation time to use the supplied ToSchema schema value as follows:</p> <ul style="list-style-type: none"> <li>For a SQL Statement, only schema names that match the FromSchema value will be substituted. The FromSchema parameter is required—otherwise, no schema values are substituted. To match and substitute more than one value pair, the FromSchema and ToSchema parameter strings can be delimited with an @ sign. For example:  FromSchema="from1@from2"  ToSchema="to1@to2"  In this example, any schema names matching "from1" are changed to "to1," and any schema names matching "from2" are changed to "to2."</li> </ul> <p>For convenience, you can supply fewer ToSchema fragments than FromSchema fragments, and when there is no corresponding ToSchema fragment, the last fragment in the ToSchema string is used. For example:  FromSchema="from1@from2@from3"  ToSchema="to" In this example, any schema names matching "from1," "from2," or "from3" will be changed to "to."</p> <ul style="list-style-type: none"> <li>For a SQL Table/View, the FromSchema parameter is optional. If it is not supplied, all schema names are changed to the supplied ToSchema value. If it is supplied, the substitution occurs in the same way as it does for a SQL Statement. If the translator property <code>sql.driver.useIdentifierQuoteString</code> is set to True within <code>customer_overrides.properties</code>, then matching and substitution occurs with quoted schema names.</li> <li>If the ToSchema parameter is supplied but is empty (equal to "" (two double quotation marks) or " (two single quotation marks)), then any matching schema names contained in the map are removed at translation time.</li> </ul>

Parameter	Description
sql_statement_use_batching	Whether to enable batching. Valid values are Yes and No (default).
sql_statement_maximum_batchsize	The maximum size of the SQL statement batch. Only valid if sql_statement_use_batching is set to Yes.

## Turning on SQL Statement Batching

In your map, any record for which the **On failure, automatically switch selected operation and retry Inserts as Updates or Updates as Inserts** setting is turned on (enabled) is not be batched, because batching is not supported for records that have retry enabled. For these records, the SQL is executed with no batching, and records that do not have retry enabled are batched.

Additionally, in the map, the data source must have **Use Transaction** enabled. If **Use Transaction** is turned off, then batching is not performed

Finally, the database must support batching. If the database does not support batching, the batch service parameters will be ignored and the SQL statements will not be batched.

This example BPML demonstrates how you might enable SQL statement batching:

```

<operation
name="Translation">
  <participant
name="Translation"/>
    <output message="TranslationTypeInputMessage">

      <assign to="map_name">insert</assign>

      <assign to="sql_statement_use_batching" from="'yes'"/>

      <assign to="sql_statement_maximum_batchsize" from="'500'"/>

      <assign to="." from="*"></assign>

    </output>
    <input message="in">

      <assign to="." from="*"></assign>

    </input>
  </operation>

```





---

## Chapter 41. X12 Deenvelope Service

### CAUTION:

This is an internal service that should not be used externally for steps in creating business processes because it is subject to change without notice, and use may cause unpredictable results and loss of data. This section is intended for information purposes only.

The following table provides an overview of the X12 Deenvelope service:

System name	DeenvelopeX12Type
Graphical Process Modeler (GPM) categories	All Services, EDI > X12
Description	Handles deenveloping of inbound X12 interchanges. It does compliance checking (except for sequence checking). It also generates 997, 999, and TA1 acknowledgements, and reconciles inbound 997 and 999 acknowledgements, if no sequence checking is required.
Business usage	The business value of the service is to improve performance by deferring sequence checking (if required) so that database updates of control numbers can be done by the EDI Post Processing service.
Usage example	An inbound purchase order is received inside an X12 interchange. The EDI envelopes are parsed and the document envelopes that match the envelope data are retrieved. With the document envelopes, this service knows what to do with the purchase order, such as initiating a business process to perform some business logic.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	If sequence checking is required, this service works in conjunction with the EDI Post Processor service.
Application requirements	None

### Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

## Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

## Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service
- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM. It must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

**Note:** All EDI services assign a Unique ID to each log message.

## Adding Translation Map Name to Process Data

The X12 Deenvelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound

translation to process data. The X12 Develope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.



## Chapter 42. X12 Envelope Service

The following table provides an overview of the X12 Envelope service:

System name	EnvelopeX12Type
Graphical Process Modeler (GPM) categories	All Services, EDI > X12
Description	Handles enveloping of outbound X12 interchanges.
Business usage	This service improves performance of EDI X12 by consolidating EnvelopeST, EnvelopeGS, and EnvelopeISA into a single service.
Usage example	An outbound purchase order is to be sent inside an X12 interchange. The document envelopes that match the SenderID, ReceiverID, and AcceptorLookupAlias specified in the EDI Encoder service are retrieved. If required by the ST envelope, translation is performed using the map specified by the envelope. The ST, GS, and ISA envelopes are applied to the output of this step. It checks if the document being enveloped is a 997 or a 999. The service will then initiate a business process if one is specified in the ISA envelope definition.
Preconfigured?	Yes
Requires third party files?	No
Platform availability	All supported application platforms.
Related services	EDI Encoder, EDI Envelope
Application requirements	None
Initiates business processes?	Runs a business process specified in the interchange envelope definition.
Invocation	Runs as part of a predefined EDI X12EnvelopeUnified business process.
Business process context considerations	None
Returned status values	<ul style="list-style-type: none"> <li>• Translation Error–Translation produced errors.</li> <li>• No_Documents_To_Envelope–EDIEncoder has not run prior to X12 Envelope service.</li> <li>• No_Envelope_Defined–The ST envelope defined has a SenderID, ReceiverID, or AcceptorLookupAlias different from that in the EDIEncoder step of the business process.</li> <li>• Error–A database or other exception takes place.</li> <li>• Success–Service returns success if none of the above takes place.</li> </ul>
Restrictions	None

Persistence level	System default
Testing considerations	None

## How the X12 Envelope Service Works

The X12 Envelope service is used in the predefined X12EnvelopeUnified business process to envelope outbound EDI documents. No configuration of the service is required.

The following example illustrates one way that the X12 enveloping process could happen:

1. A purchase order is deposited into a folder on your application system by your inventory management system.
2. The file is collected by a File System adapter, which then initiates a business process that takes care of translating the file into EDI X12 format.
3. That business process initiates the X12EnvelopeUnified business process, the document is translated into EDI X12 format, and then enveloped.
4. The document envelopes that match the SenderID, ReceiverID, and AcceptorLookupAlias specified in the EDI Encoder service are retrieved. If required by the ST envelope, translation is performed using the map specified by the envelope. The ST, GS, and ISA envelopes are applied to the output of this step. It checks if the document being enveloped is a 997 or a 999. The service will then initiate a business process if one is specified in the ISA envelope definition.

**Note:** All EDI services assign a Unique ID to each log message.

## Document Tracking Levels and Performance

You can boost EDI performance in the application by using the TRACKING\_LEVEL parameter to adjust the tracking level for business processes.

You set the default global settings for the TRACKING\_LEVEL parameter in the enveloping.properties file. However, these global settings can be overridden for certain EDI-related services by using the BPML-only TRACKING\_LEVEL parameter. This enables you to obtain maximum EDI performance in some business processes and maximum search and tracking functionality in others. This parameter can be set for the following services:

### Inbound

- CII Deenvelope service
- EDIFACT Deenvelope service
- EDI Post Processor service
- X12 Deenvelope service
- Generic Deenvelope service

### Outbound

- EDI Encoder service
- CII Envelope service
- EDIFACT Envelope service
- Envelope Generic service

- X12 Envelope service

This performance boost is done at the expense of Tracking and Search functionality. The tracking level setting affects the following EDI functionality:

- EDI Correlation Search
- EDI Document Tracking
- EDI Reporting

The TRACKING\_LEVEL parameter is not available in the application service configuration or in the GPM. It must be added manually to the BPML. Use the TRACKING\_LEVEL parameter with one of the following settings:

Setting	Description
none	Provides the largest EDI performance boost with the least tracking and search functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are nonfunctional.
basic	Provides an EDI performance boost while also providing search functionality. EDI Correlation Search is functional. EDI Document Tracking and EDI Reporting are nonfunctional.
full	Default setting. Provides the lowest EDI performance with the highest search and tracking functionality. EDI Correlation Search, EDI Document Tracking and EDI Reporting are fully functional.

**Note:** Document tracking is turned off by default in the system-defined EDI business processes. If you define an EDI business process and turn Document Tracking on, that will override the TRACKING\_LEVEL settings in both the enveloping.properties file and the EDI service parameter.

**Note:** All EDI services assign a Unique ID to each log message.

## Using Wildcards in Enveloping

To help reduce the number of envelopes that need to be created and maintained in the system, X12 and EDIFACT enveloping allows users to create wildcard envelope definitions. There are two aspects to this feature in outbound processing. The first is the use of an asterisk (\*) in any mandatory field in an outbound envelope. The second is the ability to override values set in an envelope definition through the use of correlations. By using an asterisk in the Sender ID, Receiver ID, and Acceptor Lookup Alias fields, it allows the EDI Encoder Service to match and use that envelope for every document it prepares for enveloping. You may use wildcards for one, two, or all three fields when you define an envelope, and the EDI Encoder will find and use the most specific match when it processes a document.

If an envelope field is set to an asterisk, the X12 Envelope service must obtain the actual value to use from a different source—the correlations. You must provide a correlation for an envelope value that is set to asterisk, but you can also set others. Correlations set on the document for other fields in an envelope override what is in the envelope itself. This enables you to create an envelope with default values

that you can override only when desired. The exception to this rule is when the field is Sender ID, Receiver ID, or a qualifier for one of these fields. In these fields, you must define the value as an asterisk in the envelope definition if you want to override it with a correlation, otherwise the value from the envelope is always used.

The following list contains the correlation values that can be set inside of process data prior to calling the Correlation service to override outbound envelope values:

- X12EnvelopeParms/Out\_ExpectAcknowledgement
- X12EnvelopeParms/Out\_AcknowledgementFormat
- X12EnvelopeParms/Out\_GroupAcknowledgementOverdueTime
- X12EnvelopeParms/Out\_InterchangeAcknowledgementOverdueTime
- X12EnvelopeParms/Out\_GroupAcknowledgementOverdueTimeMinutes
- X12EnvelopeParms/Out\_InterchangeAcknowledgementOverdueTimeMinutes
- X12EnvelopeParms/OutDocEncoding
- X12EnvelopeParms/Out\_TransactionSetIDCode
- X12EnvelopeParms/Out\_GroupFunctionalIDCode
- X12EnvelopeParms/Out\_GroupResponsibleAgencyCode
- X12EnvelopeParms/Out\_GroupVersionReleaseIDCode
- X12EnvelopeParms/Out\_GroupSenderID
- X12EnvelopeParms/Out\_GroupReceiverID
- X12EnvelopeParms/Out\_InterchangeAuthorizationInformationQualifier
- X12EnvelopeParms/Out\_InterchangeAuthorizationInformation
- X12EnvelopeParms/Out\_InterchangeSecurityInformationQualifier
- X12EnvelopeParms/Out\_InterchangeSecurityInformation
- X12EnvelopeParms/Out\_InterchangeSenderIDQualifier
- X12EnvelopeParms/Out\_InterchangeSenderID
- X12EnvelopeParms/Out\_InterchangeReceiverIDQualifier
- X12EnvelopeParms/Out\_InterchangeReceiverID
- X12EnvelopeParms/Out\_InterchangeControlStandardsIdentifier
- X12EnvelopeParms/Out\_InterchangeControlVersionNumber
- X12EnvelopeParms/Out\_InterchangeAcknowledgmentRequested
- X12EnvelopeParms/Out\_InterchangeTestIndicator

The following example shows how you might set correlation values in a business process:

```
<operation name="SetTheCorrelations">
    <participant name="CorrelationService"/>
    <output message="Xout">
        <assign
to="TYPE">DOCUMENT</assign>
        <assign
to="CORRELATION_PATH">/ProcessData/X12EnvelopeParms/*</assign>
        <assign to="SCOPE" from="'EDI'"/>
        <assign to="." from="*"></assign>
    </output>
    <input message="xin">
```



```

        <assign to="." from="*"></assign>
    </input>
</operation>

<operation name="EDI Encoder">
<participant name="EDIEncoder"/>
<output message="EDIEncoderTypeInputMessage">
    <assign to="AcceptorLookupAlias">837</assign>
    <assign to="EDIStandard">X12</assign>
    <assign to="ReceiverID">TestA-GS-R</assign>
    <assign to="SenderID">TestA-GS-S</assign>
    <assign to="." from="*"></assign>
</output>
    <input message="inmsg">
        <assign to="." from="*"></assign>
    </input>
</operation>

<operation name="EDI Envelope">
<participant name="EDIEnvelope"/>
<output message="EDIEnvelopeTypeInputMessage">
    <assign to="MODE">DEFERRED</assign>
    <assign to="RECEIVER_ID">TestA-GS-R</assign>
    <assign to="SENDER_ID">TestA-GS-S</assign>
    <assign to="." from="*"></assign>
</output>
    <input message="inmsg">
        <assign to="." from="*"></assign>
    </input>
</operation>

```

After the steps shown in the previous example, you would include the Correlation service to set the values as correlations against your documents, then follow that with the EDI Encoder service.

## Adding Translation Map Name to Process Data

The X12 Envelope service automatically adds the name of the map used by the translator (as specified when building the envelope) in an inbound or outbound translation to process data. The X12 Envelope service writes the map name into the process data regardless of the reason the translator was invoked; that is, for a compliance check only, or for both compliance check and translation. The map

name in process data enables enhanced configuration possibilities for your business process models. For example, you can configure business processes to use the map name for tracking or cross reference purposes, configure decisions in your process models to choose a subprocess according to the map that was run, or to create a report when there are translation errors.

---

## 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 grant you any license to these patents. You can send license inquiries, in writing, to:

*IBM Director of Licensing*

*IBM Corporation*

*North Castle Drive*

*Armonk, NY 10504-1785*

*U.S.A.*

For license inquiries regarding double-byte character set (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 Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites 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 document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

#### COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© IBM 2014. Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. 2014.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

#### **Trademarks**

IBM, the IBM logo, and [ibm.com](http://www.ibm.com)<sup>®</sup> 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 <http://www.ibm.com/legal/copytrade.shtml>.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Linear Tape-Open, LTO, the LTO Logo, Ultrium and the Ultrium Logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and other countries.

Connect Control Center®, Connect:Direct®, Connect:Enterprise®, Gentran®, Gentran®:Basic®, Gentran:Control®, Gentran:Director®, Gentran:Plus®, Gentran:Realtime®, Gentran:Server®, Gentran:Viewpoint®, Sterling Commerce™, Sterling Information Broker®, and Sterling Integrator® are trademarks or registered trademarks of Sterling Commerce®, Inc., an IBM Company.

Other company, product, and service names may be trademarks or service marks of others.





Product Number:

Printed in USA