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	*Electronic Data Interchange-Internet Integration (EDIINT)* 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 (up to 2GB or more). **Note:** Application also includes the EDIINT Message service, which builds and parses both EDIINT AS1 and 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 **Build MDNs** parameter is set to **No**), 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.
Business usage	You can use this service in two ways by embedding it in a business process: 1 To parse EDIINT AS2 messages. 2 To build EDIINT AS2 messages, which can then be sent to a trading partner.
Usage examples	See *Business Process Examples*.
Preconfigured?	Configurations of the EDIINT Pipeline service for building and parsing messages are installed with Application.
Requires third party files?	◆ TrustpointAll.jar and TrustpointProviders.jar files from SecurityBuilder PKI-J 3.1 ◆ EccpressoAll.jar file from SecurityBuilder Crypto-J 2.3
Platform availability	All supported Application platforms
Related services	◆ EDIINT Acknowledge Check service. Use this service when MDNs are expected for messages built with the EDIINT Pipeline service. ◆ EDIINT Message service ◆ EDIINT MDN Building service ◆ EDIINT Header Scanning service

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	◆ You must have profiles and contracts configured for the trading partner. ◆ You must have the HTTP Server adapter configurations that pass inbound messages to this service configured to send raw messages.
Persistence level	None
Testing considerations	To test and verify that the EDIINT Pipeline service can parse an AS2 message: 1 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. 2 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 2GB) AS2 messages
  - 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.

Several packaged AS2 business processes were updated to use the new EDIINT Pipeline Service, as opposed to the older EDIINT Message Service. The EDIINT Pipeline Service has the same functional behavior as the older service, however 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 specify settings for the following fields in Application:

Field	Description
Config	Name of the service configuration.
Action	Valid values are Parse or Build. Required.
Parameters for Parse action:	
RequireAddresses	Required. Valid values are: ◆ True – Service fails if a contract for an inbound message cannot be looked up. (default) ◆ False – Service attempts to parse a message, and if successful, tries to run the default business process.
Default Data Business Process	Name of a business process to run if False is selected for RequireAddresses and no contract has been found for an inbound message. Must be the name of a business process. Required if RequireAddresses is set to False.
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 Building Process	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 Building 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 **Yes** (the default), 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 (default) and No. Required. **Note:** 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.
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. **Note:** 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.

Field	Description
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 True and False. Default is False. Required.
Process Duplicate Messages	Send business data from messages with duplicate message IDs into Application. Valid values are True and False. Default is False. Required. **Note:** If you select True, you may receive duplicate data. **Note:** 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 TRUE. After you change the rule and save the customer_overrides.properties file, you must restart Application for the changes to take effect.
Update expired messages	Whether to update the status of EDIINT transactions that currently have a status of Expired when an acknowledgement is received. Valid values are True and False. Default is False. Required. **Note:** If you select False, and an MDN arrives for an expired transaction, status information is placed in the additional data field of the transaction record.
Pipeline timeout (Seconds)	Maximum processing time for a request before it is timed out. Default is 1800 (seconds). Required.
Parameter for Build action:	
Contract ID	The ID of a contract containing information for building messages. Must be the ID of an existing contract. Optional.

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>
```