webMethods HIPAA Module Installation and User's Guide

Version 6.0.2 Feature Pack 2

July 2004

webMethods

This document applies to webMethods HIPAA Module Version 6.0.2 Feature Pack 2 and to all subsequent releases.

Specifications contained herein are subject to change and these changes will be reported in subsequent release notes or new editions.

Document ID: ESTD-HIPAA-IUG-602-FP2-20121011

Contents

About This Guide

This guide provides a description of the webMethods HIPAA Module and instructions to use the webMethods HIPAA Module.

Document Conventions

Convention	Description
Bold	Identifies elements on a screen.
Italic	Identifies variable information that you must supply or change based on your specific situation or environment. Also identifies service input and output variables.
Narrow font	Identifies storage locations for services on the webMethods Integration Server using the convention *folder.subfolder:service*.
`Typewriter font`	Identifies characters and values that you must type exactly or messages that the system displays on the console.
UPPERCASE	Identifies keyboard keys. Keys that you must press simultaneously are joined with the "+" symbol.
\	Directory paths use the "\" directory delimiter unless the subject is UNIX-specific.
[ ]	Optional keywords or values are enclosed in [ ]. Do not type the [ ] symbols in your own code.

Additional Information

The webMethods Advantage Web site at http://advantage.webmethods.com provides you with important sources of information about your webMethods components:

- **Troubleshooting Information.** webMethods provides troubleshooting information for various webMethods components in the webMethods Knowledge Base.
- **Documentation Feedback.** To provide documentation feedback to webMethods, go to the Documentation Feedback Form.
- **Additional Documentation.** All of the webMethods documentation is available on the webMethods Bookshelf.

webMethods

CHAPTER 1

Concepts

What is the webMethods HIPAA Module?

The webMethods HIPAA Module is a comprehensive and highly scalable solution that allows your organization to implement the HIPAA standard. This webMethods component provides out-of-the-box ability to receive, parse, and validate all of the mandated HIPAA transactions as well as respond with the appropriate acknowledgments. The HIPAA Module streamlines health care industry transactions by providing a solution for rapid and seamless integration of Providers, Payers, Routes, and Sponsors.

Important! The HIPAA Module runs on top of the webMethods EDI Module. When you install the HIPAA Module, it changes the behavior of some of the functions of the webMethods EDI Module to meet HIPAA standards. If you plan to process both HIPAA-related and non-HIPAA related EDI documents, webMethods recommends that you set up the processing on different machines. If you prefer to use the same machine, be careful when setting up and testing to ensure that the processing for HIPAA and non-HIPAA related documents functions as anticipated.

What HIPAA Transactions Does the HIPAA Module Support?

The following table lists the HIPAA transactions and addenda that the HIPAA Module supports.

Action	Enables users to...
270 Health Care Eligibility Benefit Inquiry	Send a Health Care Eligibility Inquiry, commonly known as 270, to submit an inquiry to the trading partner (generally an Insurance company), to determine whether a patient is eligible for certain claim benefits The HIPAA Module supports both version 004010X092 (standard implementation) and version 004010X092A1 (addendum).
271 Health Care Eligibility Benefit Response	Send a Health Care Eligibility Response, commonly known as 271, to submit a response to the trading partner stating whether the patient is eligible. The response document contains the details like eligibility status, maximum benefits, in-plan/out of plan benefits, co-payments, etc. The HIPAA Module supports both version 004010X092 (standard implementation) and version 004010X092A1 (addendum).

Action	Enables users to...
276 Health Care Claim Status Request	Send a Health Care Claim Status request, commonly known as 276, to request the current status of a specified claim. The HIPAA Module supports both version 004010X093 (standard implementation) and version 004010X093A1 (addendum).
277 Health Care Claim Status Response	Send a Health Care Claim Status Response, commonly known as 277, to transmit the current status within the adjudication process to the requester. When the status request does not uniquely identify the claim within the payer's system, the response may include multiple claims that meet the identification parameters supplied by the requester. The HIPAA Module supports both version 004010X093 (standard implementation) and version 004010X093A1 (addendum).
278 Health Care Services Review (Request and Response)	Send a Health Care Services Review - Inquiry and Response, commonly known as 278, to handle informational inquiries and responses.The 278 can be exchanged between interested participants in a bi-directional inquiry/response mode of operation. This mode would allow a participant to inquire about existing certifications. The HIPAA Module supports only version 004010X094A1 (addendum).
820 Payment Order/ Remittance Advice	Send a Payment Order/Remittance Advice, commonly known as 820, to validate a premium payment advice for a sender to move only money, to move money and detailed or summary remittance information, or to move only detailed or summary remittance information. The HIPAA Module supports both version 4010X061 (standard implementation) and version 4010X061A1 (addendum).
834 Benefit Enrollment and Maintenance	Send a Benefit Enrollment and Maintenance, commonly known as 834, to transfer enrollment information from the sponsor of the insurance coverage, benefits, or policy to a payer. The HIPAA Module supports both version 4010X095 (standard implementation) and version 4010X095A1 (addendum).

Action	Enables users to...
835 Claim Payment/Advice	Send a Claim Payment/Advice, commonly known as 835, to make a payment, send an Explanation of Benefits (EOB) remittance advice, or make a payment and send an EOB remittance advice from a Health Care payer to a Health Care provider, either directly or through a Depository Financial Institution (DFI). The HIPAA Module supports both version 4010X091 (standard implementation) and version 4010X091A1 (addendum).
837 Health Care Claims fall into three categories: Institutional, Dental, and Professional. The HIPAA Module supports all three categories.	
837 Health Care Claim (Institutional)	Send a Health Care Claim (Institutional), commonly known as 837-I, to handle coordination of benefits (COB) in a totally EDI environment. The HIPAA Module supports both version 4010X096 (standard implementation) and version 4010X096A1 (addendum).
837 Health Care Claim (Dental)	Send a Health Care Claim (Dental), commonly known as 837-D, to handle coordination of benefits (COB) in a totally EDI environment. The HIPAA Module supports both version 4010X097 (standard implementation) and version 4010X097A1 (addendum).
837 Health Care Claim (Professional)	Send a Health Care Claim (Professional), commonly known as 837-P, to handle coordination of benefits (COB) in a totally EDI environment. The HIPAA Module supports both version 4010X098 (standard implementation) and version 4010X098A1 (addendum).

webMethods HIPAA Module Features

The HIPAA Module runs on top of the webMethods Integration Server, webMethods Trading Networks, and the webMethods EDI Module. The HIPAA Module can help clients achieve an easy, secure, and reliable solution for seamless integration with their back-end systems as well as their trading partners.

The HIPAA Module provides support for the following.

- **Out-of-the-box HIPAA transactions.** This enables you to quickly implement production solutions for automating the many interactions between you and your trading partners.
- **Out-of-the-box validation.** The HIPAA Module provides out-of-the-box validation through Level 5 as defined by WEDI-SNIP certification guidelines.

  By adding other code sets and customizing validation, the HIPAA Module also is capable of Level 6 and 7 validation as defined by WEDI-SNIP certification guidelines.
- **Code sets.** You can import all code sets that HIPAA supports using the HIPAA Module. For more information, see Chapter 8, "Managing HIPAA Code Sources" in this guide.
- **Large document processing.** Instead of processing large HIPAA EDI documents all at once, the HIPAA Module processes documents segment by segment to improve performance.
- **TA1 and 997 acknowledgments**. The HIPAA Module fully supports HIPAA-defined success/failure notification of envelope errors using TA1 technical acknowledgments. The HIPAA Module also supports 997 functional acknowledgements to communicate syntax errors in HIPAA messages.
- **Error reports**. The HIPAA Module generates detailed, human-readable error reports. You can send these reports in an e-mail message to a person in your enterprise or to a trading partner.
- **Leveraging existing investments in enterprise solutions.** You do so by accepting information from EDI-based systems to populate documents in HIPAA format.
- **The widest range of eStandards**. The HIPAA Module supports EDI, EDIINT, OBI, Acord, cXML, OAG, and BizTalk. It gives you the ability to use a preferred approach in-house while supporting the different standards requirements of your customers.
- **Transaction logging and audit trails.** This ensures the integrity of all trading partner transactions. Automatic archival of transaction messages ensures non-repudiation of content.

webMethods HIPAA Module Packages

The HIPAA Module contains the following packages (sets of services and related files) that you install on the Integration Server.

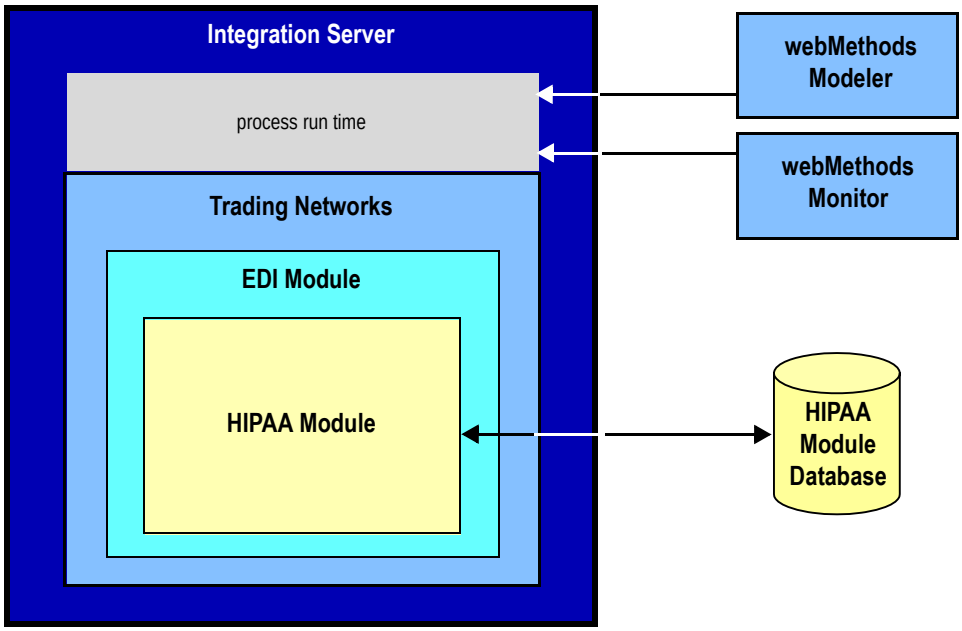
Package	Description
WmHipaa	Contains general application functionality and serves as the main holding area for application user interfaces, typically DSPs, that are not provided by the Integration Server Administrator. This package also contains shared components including implementations of common utilities, common validation services, and transport services.
WmHipaaCodeSource	Support for the HIPAA Code Source Utility that you can use to add and manage HIPAA Code Sources.
WmHipaaTransactions	Holds the flat file schemas and validation services corresponding to the HIPAA transactions.
WmHipaaSample	Illustrates how to use the HIPAA Module to process large HIPAA messages to meet the HIPAA standard. You can use the sample services to process your own messages. For more information, see Appendix B, "webMethods HIPAA Module Sample" in this guide and the services themselves in the webMethods Developer.

For detailed information about the HIPAA Module services, see Chapter 9, "WmHipaa, WmHipaaCodeSource, and WmHipaaSample Package Services" and Chapter 10, "WmHipaaTransactions Package Services" in this guide.

webMethods HIPAA Module Architecture

The following diagram illustrates how the HIPAA Module fits into the webMethods architecture. For more information about the various elements in the diagram, see the table following the diagram.

Architecture and Components



Component	Description
Integration Server	The Integration Server is the underlying foundation of the webMethods architecture. It processes requests from and relays responses to a back-end system. Among the services provided with the Integration Server, the HIPAA Module specifically uses services in the WmFlatFile package, which use flat file schemas to validate HIPAA transactions.
Trading Networks	Trading Networks is a webMethods component that enables your enterprise to link with other companies and exchanges to form a business-to-business trading network. For more information, see the *webMethods Trading Networks—Getting Started with Trading Networks* guide.

Component	Description
webMethods EDI Module	The webMethods EDI Module is a webMethods component that enables your enterprise to receive and process EDI documents. To use the HIPAA Module, you use two packages of the webMethods EDI Module: ■ **WmEDI package**, which contains the basic functionality that provides support for the EDI standard to the webMethods architecture. ■ **WmEDIforTN package**, which allows for the interaction between the WmEDI package and Trading Networks. This interaction allows you to use Trading Networks as a gateway for EDI document exchange.
Modeler	Modeler is a design-time tool that you can use to create process models that define the steps in a *business process* (also known as a *conversation*). After you design the process models, you generate them to create the run-time elements (for example, flow services, triggers, etc.) that reside in the Integration Server. The process run time facility of the Integration Server executes the business processes (conversations) at run time. You also can use process models to process HIPAA documents. For more information, see the *webMethods EDI Module Trading Networks User's Guide* and the *webMethods Modeler User's Guide*.
Monitor	Monitor is a webMethods component that allows you to monitor the progress and status of the business processes (conversations) involving EDI documents. Monitor interacts with the process run time to obtain the status information. For more information about Monitor, see *webMethods Monitor User's Guide*.
HIPAA Module	The HIPAA Module is a webMethods component that adds support for the HIPAA standard.
HIPAA Module Database	The HIPAA Module database is a relational database that the HIPAA Module uses to store code sources.

Process Overview

To process HIPAA messages, you use the facilities provided by:

- The HIPAA Module, which provides the HIPAA-related validation services
- The webMethods EDI Module, which provides processing for EDI documents
- Trading Networks, which handles the routing of messages to trading partners
- Integration Server, which handles the basic sending and receiving of messages to and from the Integration Server

Additionally, you must add your own processing to do the following:

- **Send HIPAA messages to your trading partners.** If you are acting as a sender, you can use a Trading Networks delivery service to send a valid standard HIPAA message to your trading partner (acting as the receiver).
- **Send the appropriate acknowledgements to the HIPAA messages.** The HIPAA Module provides built-in services that a receiver can invoke to generate acknowledgements. To send acknowledgements, you can use Trading Networks delivery features. For more information, see the Trading Networks documentation.

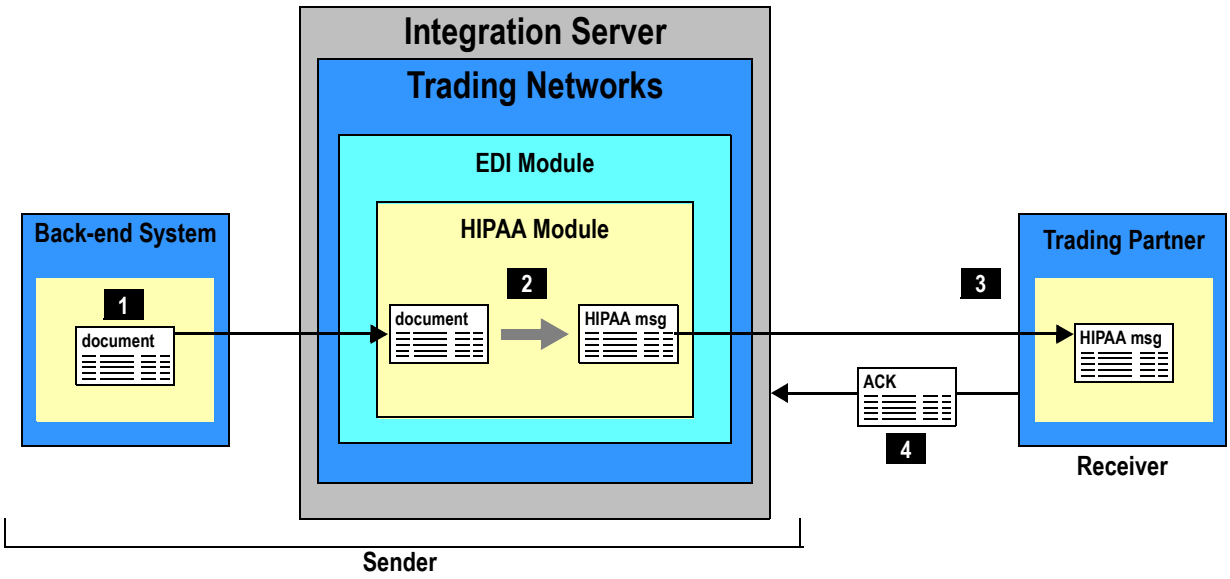
  The HIPAA Module sample illustrates how to use its built-in services. For more information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.
- **Process the HIPAA transactions to meet your specific needs.** For example, you might want to map the data from a 834 Benefit Enrollment and Maintenance to a back-end system document and send that document to your back-end system.

To add your own processing, you can use either Trading Networks processing rules or you can use Modeler to define a process model for a business process.

Sender-Side Processing

The sender forms a HIPAA message and sends it to a trading partner (that is the *receiver* of the HIPAA message). The following diagram illustrates sender-side processing. For more information, see the table after the diagram.

Sender-side processing of HIPAA messages

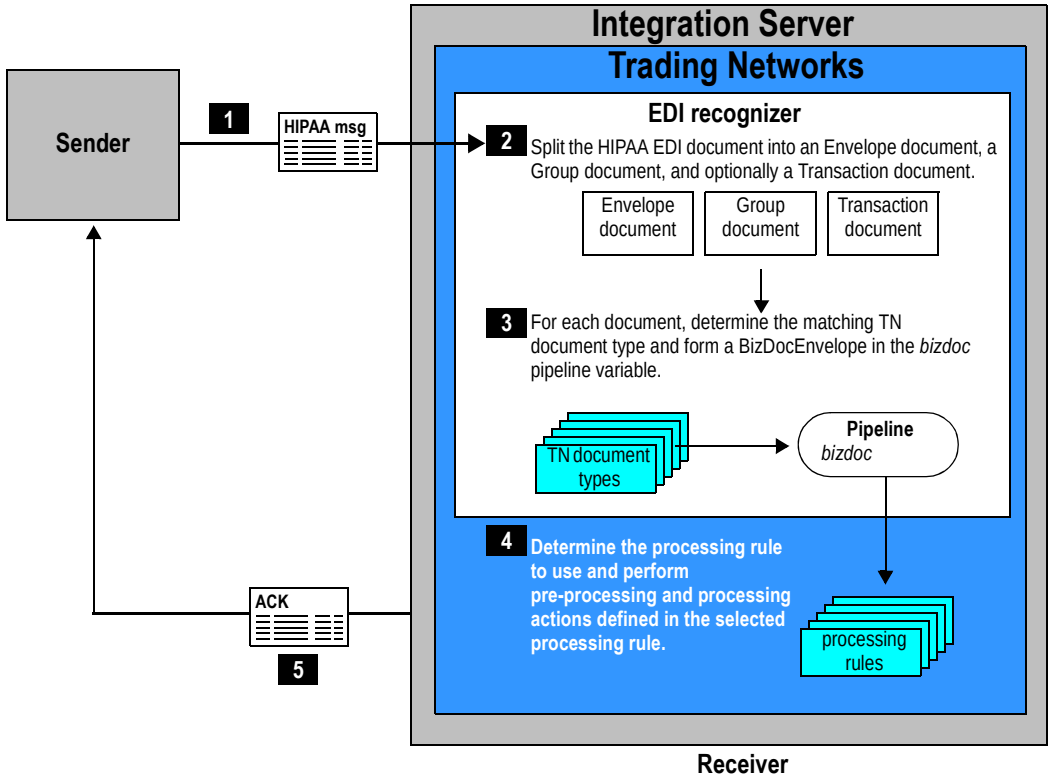


Step	Description
1	The back-end system sends a document to the Integration Server. The document can be: ■ In an internal format used by the back-end system. ■ A valid HIPAA message.
2	The actions performed on the Integration Server depend on the type of document sent: ■ If the back-end system sends documents in an internal format, you set up logic on the Integration Server to map the data from the internal format document to a standard HIPAA message. For more information about how to map data from an internal format document to a HIPAA message, see information about mapping data in documentation for the webMethods EDI Module. ■ If the back-end system sends valid HIPAA messages, the Integration Server needs only to send the HIPAA message to the receiver. You can use Trading Networks delivery features to send the HIPAA message to the receiver. For more information, see the Trading Networks documentation.
3	After a valid standard HIPAA message is available, use a Trading Networks delivery service to send the document to your trading partner (acting as the receiver). For more information, see the Trading Networks documentation.
4	The sender processes the inbound acknowledgements that are sent by the trading partner (acting as the receiver). For information about how to process acknowledgements, see Chapter 6, "Processing HIPAA Acknowledgements (TA1 & 997)" in this guide.

Receiver-Side Processing

The following diagram illustrates receiver-side processing when using Trading Networks processing rules. For more information, see the table after the diagram.

Receiver-side processing



Step	Description
1	Your trading partner creates a client that forms a HIPAA standard message and sends the HIPAA message to your Integration Server.
2	The Integration Server receives the HIPAA message and sends it to Trading Networks for processing. Because the HIPAA message is an EDI document, Trading Networks passes the document to its EDI recognizer, which is a recognizer that is added to Trading Networks when you install the webMethods EDI Module. The EDI recognizer splits the HIPAA message into separate documents. You define the level of documents (Envelope, Group, or Transaction) that you want the HIPAA message split into by setting a variable in an EDI trading partner agreement (EDITPA). For HIPAA processing, you *must* split at least at the Group level to form both Envelope and Group documents. For more information about EDITPAs and the EDITPA *splitOption* variable, see Chapter 3, "Configuring the webMethods HIPAA Module" in this guide and the documentation for the EDI Module.
3	For each document (Envelope, Group, or Transaction) split from the original HIPAA message, the EDI recognizer uses the TN document types to determine the type of document (for example, X12 Envelope, X12 Group, or X12 4010 835). The EDI Module provides the TN document types for EDI documents. You install the TN document types that you need. For more information, see "Step 1: Install TN Document Types for HIPAA Transactions" on page 32. The only exception is the TA1 technical acknowledgement, for which the HIPAA Module automatically installs a flat file TN document type. After recognizing the type of document using TN document types, the EDI recognizer forms a *BizDocEnvelope* for the EDI document. The BizDocEnvelope is in the *bizdoc* pipeline variable. A BizDocEnvelope contains the original document (Envelope, Group, or Transaction) and includes additional information that Trading Networks requires for routing and processing the document. In other words, the BizDocEnvelope represents a routable Trading Networks transaction.

Step	Description
4	After forming the BizDocEnvelope, the document is passed to regular Trading Networks processing. Trading Networks determines the processing rule to use to process the document and executes the processing rule. You create processing rules to define the processing you want performed on each type of document. For example, ■ You set up processing for the Envelope document to validate the envelope and generate a TA1 technical acknowledgement if appropriate. ■ You set up processing for a Group document to validate the group and transactions and generate a 997 functional acknowledgment. ■ You set up processing rules for Transaction documents to perform any processing on the transaction. For example, you might map the data to a back-end system document and send the newly formed document to your back-end system. For more information about defining processing rules, see "Using Processing Rules to Process Inbound HIPAA Messages" on page 47.
5	Send acknowledgements (997 and TA1) back to the sender using Trading Networks delivery features. For more information, see the Trading Networks documentation.

webMethods

CHAPTER 2

Installing the webMethods HIPAA Module

Important! The information in this chapter might have been updated since the guide was published. Go to the webMethods Advantage Web site at http://advantage.webmethods.com for the latest version of the guide.

System Requirements

This section describes the system requirements that must be met before you can install the webMethods HIPAA Module.

Platform and Operating System Requirements

The webMethods HIPAA Module has no system requirements beyond those of the webMethods Integration Server. For Integration Server system requirements, see the *webMethods Integration Platform Installation Guide*.

webMethods Software Requirements

The following table lists the webMethods components you must install before you can install the webMethods HIPAA Module:

Component	Version
webMethods Integration Server	6.0.1, Service Pack 2 for 6.0.1, and Service Pack 2 for 6.0.1 Fixes 68 and 92
webMethods Trading Networks Server	6.0.1
webMethods Developer	6.0.1
webMethods Trading Networks Console	6.0.1
webMethods EDI Module	6.0.1 and Fix 9 and Fix 12 for WmEDI
webMethods EDI Module Trading Networks	6.0.1 and the HIPAA 6.0.2 Fix for WmEDIforTN

Third-Party Software Requirements

None.

Database Requirements

The HIPAA Module requires the use of a database. The following are the supported databases for the HIPAA Module.

Database	Platforms Database Supported On
Oracle 8.1.7 and 9.0.1	All platforms supported by webMethods Trading Networks.
SQL Server 2000	All platforms supported by webMethods Trading Networks.

Hardware Requirements

The webMethods HIPAA Module has no hardware requirements beyond those for the Integration Server. For Integration Server hardware requirements, see the *webMethods Integration Platform Installation Guide*.

webMethods Component Compatibility

webMethods Integration Server 6.0.1, webMethods Trading Networks 6.0.1, webMethods Developer 6.0.1, webMethods EDI Module 6.0.1, webMethods EDI Module Trading Networks 6.0.1, webMethods Modeler 6.0.1, and webMethods Monitor 6.0.1 are compatible with the webMethods HIPAA Module Version 6.0.2 Feature Pack 2. For information about the Service Packs and Fixes needed for these components, see "webMethods Software Requirements" on page 24.

Installing webMethods HIPAA Module

To install the webMethods HIPAA Module, perform the following steps:

Step	Description
1	Install the webMethods HIPAA Module.
2	Create the database tables that the webMethods HIPAA Module requires.
3	Populate the HIPAA Module database with initial values.

Step 1: Install the webMethods HIPAA Module

When you install the webMethods HIPAA Module, you install the following packages:

- WmHipaa
- WmHipaaCodeSource
- WmHipaaTransactions
- WmHipaaSample

To install the webMethods HIPAA Module

1. Download the newest version of the webMethods Installer from the webMethods Advantage Web site at http://advantage.webmethods.com.
2. Run the installer using the username and password you received from webMethods. Specify the installation directory as the webMethods 6.0.1 installation directory (by default, webMethods6).
3. In the component selection list, navigate to **webMethods Platform** ▸ **eStandards** ▸ **webMethods HIPAA Module** and select the desired components:
   - **Documentation 6.0** (Optional). Contains the documentation for this package.
   - **Program Files** (Required). Contains the program files for this package.
   - **Samples 6.0** (Optional, but recommended). Contains the services that demonstrate how to use the HIPAA Module services.
   - Any required webMethods components you have not installed.
4. Click **Next** until the installer displays the **Installation Complete** panel.
5. Click **Close**.

The webMethods HIPAA Module starts automatically when you start the webMethods Integration Server.

Step 2: Create the Database Tables that the webMethods HIPAA Module Requires

You must create the database tables that the webMethods HIPAA Module requires on the server on which Trading Networks is installed. webMethods provides script files to help you create the database tables for SQL Server and Oracle. If you are using a database other than SQL Server and Oracle, you must modify the script files so that they apply to your database.

To create the database tables that the webMethods HIPAA Module requires

1 Using the tool provided with your database, import and run the appropriate script file in *webMethods6**IntegrationServer*\packages\WmHipaa\config\dbScripts to create the database tables on the server on which Trading Networks is installed.

- **If you are using SQL Server,** use the create_hipaatables_SQLServer.sql script.
- **If you are using Oracle,** use the create_hipaatables_Oracle.sql script.

Step 3: Populate the HIPAA Module Database with Initial Values

You must initialize the database tables that the HIPAA Module requires. webMethods provides script files to help you initialize the database tables for SQL Server and Oracle. If you are using a database other than SQL Server or Oracle, you must modify the script files so that they apply to your database.

To populate the webMethods HIPAA Module database with initial values

1 Using the tool provided with your database, import and run the appropriate script file in *webMethods6**IntegrationServer*\packages\WmHipaa\config\dbScripts to initialize the database tables.

- **If you are using SQL Server,** run the insert_hipaatables_SQLServer.sql script.
- **If you are using Oracle,** run the insert_hipaatables_Oracle.sql script.

Upgrading webMethods HIPAA Module

Upgrade from webMethods HIPAA Module 6.0.2 to 6.0.2 FP2

To upgrade from webMethods HIPAA Module Version 6.0.2 to webMethods HIPAA Module Version 6.0.2 FP2, you must install webMethods HIPAA Module Version 6.0.2 FP2 on top of your existing webMethods HIPAA Module Version 6.0.2.

For steps to install the webMethods HIPAA Module Version 6.0.2 FP2, see "Installing webMethods HIPAA Module" on page 25.

Uninstalling webMethods HIPAA Module

Important! This section provides only instructions that are specific to uninstalling the webMethods HIPAA Module. For complete instructions on using the webMethods Uninstaller, see the *webMethods Integration Platform Installation Guide*.

Uninstalling webMethods HIPAA Module 6.0.2/6.0.2 FP2

To uninstall webMethods HIPAA Module Version 6.0.2/6.0.2 FP2, you use the webMethods Uninstaller to uninstall the packages that you installed in "Step 1: Install the webMethods HIPAA Module" on page 26.

To uninstall webMethods HIPAA Module 6.0.2/6.0.2 FP2

1. Shut down all webMethods components and applications that are running on your machine.
2. Start the webMethods Uninstaller.
3. Select **webMethods HIPAA Module** as the program to uninstall.
4. The Uninstaller removes all webMethods HIPAA Module-related files that were installed into the *Integration Server_directory*\packages directory. The Uninstaller does not delete files created after you installed the webMethods HIPAA Module (for example, user-created or configuration files), nor does it delete the directory structure that contains the files.
5. If you do not want to save the files that the Uninstaller did not delete, navigate to the *Integration Server_directory*\packages directory and delete the HIPAA Module-related folders.

Uninstalling webMethods HIPAA Module 6.0.1

To uninstall webMethods HIPAA Module Version 6.0.1, you manually uninstall the packages that you installed in "Step 1: Install the webMethods HIPAA Module" on page 26.

To uninstall webMethods HIPAA Module 6.0.1

1. Start the Integration Server and open the Server Administrator.
2. On the Server Administrator Navigation Panel, under the **Package** menu, click **Management**.
3. In the **Package** list, locate the WmHIPAA package.

4 In the **Enabled** column for the package, click **Yes** to disable the package. The text in the **Enabled** column changes to **No** and the package is disabled and ready to be deleted.

5 Complete one of the following options:

- Click **Delete** to delete the package without keeping a backup copy. The package is deleted from the list package list in the Server Administrator *and* the file system. Packages deleted in this manner are *not* recoverable. To regain access to packages deleted in this manner, you must reinstall the deleted packages using the procedure described in "Step 1: Install the webMethods HIPAA Module" on page 26.
- Click **Safe Delete** to remove the package and keep a backup copy. (Backup copies are stored in the *webMethods6**IntegrationServer*\replicate\salvage directory on the server.)

6 Repeat steps 3-5 for the WmHipaaTransactions.zip, WmHipaaSample.zip, and WmHippaCodeSource.zip packages.

The webMethods HIPAA Module is now safely uninstalled from the Integration Server.

webMethods

CHAPTER 3

Configuring the webMethods HIPAA Module

Overview

This chapter describes how to set up the webMethods Integration Platform so that you can send and receive HIPAA messages using the services in the webMethods HIPAA Module.

Important! The following procedure assumes that you already have installed the webMethods Integration Server, webMethods Trading Networks, the webMethods EDI Module, and the webMethods HIPAA Module.

Step 1: Install TN Document Types for HIPAA Transactions

When Trading Networks receives a document, it uses its TN document types to determine the type of file it received. This is referred to as *document recognition.* In addition, the TN document type indicates the attributes that Trading Networks is to extract from the document. For more information about TN document types, see the *webMethods Trading Networks--Building Your Trading Network*.

Note: When you install a TN document type, the EDI Module also installs the corresponding flat file schema for the EDI transaction set. For more information about flat file schemas used for EDI documents, see documentation for the EDI Module. For more information about flat file schemas in general, see the *webMethods Flat File Schema Developer's Guide*.

You use the EDI Module to install TN document types for EDI documents. The TN document types are installed into Trading Networks. You need to install the TN document types for:

- The types of EDI transactions that you plan to use.
- The EDI 997 functional acknowledgement transaction (X12 4010 997).

Note: You do *not* need to install a TN document type for the TA1 technical acknowledgement transaction. When you install the HIPAA Module, the X12 TA1 TN document type is automatically installed into Trading Networks for the TA1 technical acknowledgement document. The HIPAA Module automatically installs the X12 TA1 TN document type for you because you cannot use the EDI Module to install it.

To install TN document types for HIPAA transactions

1. Start the Integration Server and open the Server Administrator.
2. On the Server Administrator navigation panel, under **Solutions**, click **EDI**. A new browser window is displayed. This is the home page for the webMethods EDI Module.
3. On the navigation panel, click **Install EDI Doc Types**.
4. Complete the following fields:

For this field...	Specify...
Standard	`X12`
Version	`4010`
Transaction Set	Type of EDI document for which you want to install the associated TN document types, for example, `835`.

Note: You do not need to install a TN document type for the TA1 technical acknowledgement document.

When you install a TN document type for a transaction set, the EDI Module automatically installs the TN document type for the Envelope and Group, if they are not already installed. As a result, the first time you install a TN document type for an X12 transaction, the X12 Envelope and X12 Group TN document types are installed.

5. Click **Add Document Type Definition to Trading Networks**.
6. Repeat steps 4 and 5 for all of the types of EDI documents that you want to use.

Step 2: Define Profiles for Trading Partners

You need to define profiles for trading partners that will be exchanging HIPAA messages. Define profiles for the trading partners that will be identified as senders and receivers on the ISA (envelope) headers.

You create profiles use the Trading Networks Console. For steps to create profiles, see the chapter about creating partner profiles in the *webMethods Trading Networks--Building Your Trading Network*.

External ID Types in Profiles and EDI ID Qualifiers

In a Trading Networks profile, you specify external IDs to indicate how trading partners are identified within the HIPAA messages that they send. For example, if a trading partner uses a D-U-N-S number in a document, you define a **DUNS** external ID type in the Trading Networks profile and specify the trading partner's D-U-N-S number as the corresponding external ID.

For EDI documents, the external IDs correspond to the sender IDs and receiver IDs in the ISA headers of the EDI documents and the external ID types correspond to the EDI ID qualifiers (for example, 01 for a D-U-N-S number).

The table below lists the external ID types that the HIPAA Module installs into Trading Networks the first time you start the Server Administrator after the HIPAA Module is installed and enabled.

Trading Networks External ID Type	Corresponds to this EDI ID Qualifier
Carrier ID	27
DUNS	01
DUNS+4	14
Federal Tax ID	30
Fiscal Intermediary ID	28
Health Industry Number	20
Medicare ID	29
Mutually Defined	ZZ
NAIC Company Code	33

Step 3: Set Up Large Document Handling

To process large HIPAA documents using the HIPAA Module, you set up the webMethods EDI Module and webMethods Trading Networks to handle large documents, which temporarily persists documents to local disk for memory and performance optimization.

Configure the webMethods EDI Module.

Important! Before you edit the properties.cnf file, shut down the Integration Server. After you make the changes, restart the server.

To configure the EDI Module to use large document handling, you must update the following properties as necessary in the *webMethods6\IntegrationServer*\packages\WmEDI\config\properties.cnf file:

- *EDIBigDocThreshold*
- *EDItSpaceTimeOut*

For more information about EDI large document handling and these properties, see the documentation for the EDI Module.

Configure the webMethods Trading Networks

Important! Before you edit the properties.cnf file, shut down the Integration Server. After you make the changes, restart the server.

To configure Trading Networks to use large document handling, you must update the following properties as necessary in the Trading Networks properties.cnf file, which is located in *webMethods6\IntegrationServer*\packages\WmTN\config:

- *tn.BigDocThreshold*
- *tn.tspace.max*
- *tn.tspace.location*

For more information about Trading Networks large document handling and these properties, see the "webMethods Trading Networks Large Document Handling" document.

Step 4: Create EDI Trading Partner Agreements

A trading partner agreement (TPA) is a Trading Networks object that defines how messages are exchanged between two trading partners. An EDI trading partner agreement (EDITPA) is a trading partner agreement that contains EDI Module-specific settings. The EDITPA contains a set of variables that you provide to tailor how the EDI Module exchanges messages between two trading partners.

The EDI Module supports both partner-specific and default EDITPAs. A partner-specific EDITPA contains variables specific to a pair of trading partners where one is defined as the sender and the other the receiver. If a partner-specific EDITPA is not defined or if a value in a partner-specific EDITPA is not set, the EDI Module uses its default EDITPA. For more information about EDITPAs, see documentation for the EDI Module.

- If you are using the EDI Module for use with only the HIPAA Module, you will need to create only a default EDITPA and set values common to all trading partners exchanging EDI documents.
- If you are using the EDI Module for more than just exchanging HIPAA messages, you might need to set up partner-specific EDITPAs for your trading partners exchanging HIPAA messages if the settings required for the HIPAA Module differ from the settings you would use normally for EDI documents.

You must define EDITPAs for envelope sender/receiver pairs identified in the ISA headers of the HIPAA messages that you will exchange. You need to define either:

- The default EDITPA for all envelope-level sender/receiver pairs.

  -OR-

- Create partner-specific EDITPAs for the envelope-level sender/receiver pairs.

For complete steps to create EDITPAs, see the documentation for the EDI Module. The following table shows the EDITPA settings you should use with the HIPAA Module. The HIPAA Module does not use EDITPA variables that are not listed in the table. You can set the values of the EDITPA variables that are not listed to any value you choose.

EDITPA variable...	Value to use for the HIPAA Module
splitOption	**Group** or **Transaction** The EDITPA *splitOption* variable indicates how you want the EDI Module to split the inbound HIPAA message. ■ Use **Group** if you want the EDI Module to split the inbound HIPAA message into an Envelope and Group documents. ■ Use **Transaction** if you want the EDI Module to split the inbound HIPAA message into an Envelope, Group, and Transaction documents.
GSRouting/routingMode	**OFF** The EDITPA *GSRouting/routingMode* variable indicates the value that you want the EDI Module to use for the sender and receiver, which the EDI Module puts in the Envelope, Group, and Transaction documents split from the HIPAA message. You must select **OFF**, which indicates that the EDI Module uses the sender and receiver from the ISA header for all types of documents.

Step 5: Install Necessary HIPAA Code Sources

The HIPAA Implementation guides include the External Code Sources that you use when encoding certain Data Elements in the Transaction Sets. The webMethods HIPAA Module uses these Code Sources to check for HIPAA compliance when exchanging documents.

Important! Some of the code sources provided by webMethods need to be licensed. Please make sure you have a license to use these code sources.

Install Code Sources Provided by the HIPAA Module

The HIPAA Module provides you with 28 of the 42 Codes Sources you may need to exchange documents. For a list of these Code Sources, see Appendix A, "Code Source List" in this guide. To use these Codes Sources, you must unzip the codesets.zip file in the *webMethods6**IntegrationServer*\packages\WmHipaaCodeSource\data directory, and then enable the Code Sources that you want to use. For steps to enable the Code Sources

provided by the HIPAA Module, see Chapter 8, "Managing HIPAA Code Sources" in this guide.

Important! Code sets are constantly changing. The *webMethods6\IntegrationServer* packages\WmHipaaCodeSource\data\codesets.zip file contains the version of each code set that was available when the HIPAA Module was released. Please check the webMethods Advantage Download site for the latest version of these code sets.

Install Additional Code Sources

If you need a Code Source not supplied by the HIPAA Module, you must complete the following steps:

- Acquire the Code Source.
- Ensure that the Code Source is in the format described in "Formatting the Code Source Value File" on page 67.
- Copy the Code Source onto the system where the Integration Server is running.
- Follow the steps in "Adding or Upgrading a Code Source" on page 70.

webMethods

CHAPTER 4

Creating Clients that Send HIPAA Messages

Overview

You create an Integration Server client to send HIPAA messages to the Integration Server. Examples of applications that might use clients to send HIPAA messages are:

- A back-end system (for example, SAP or Oracle Apps) that sends a HIPAA message
- A Integration Server that sends a HIPAA message to another server
- A trading partner that is not using webMethods software that sends a HIPAA message

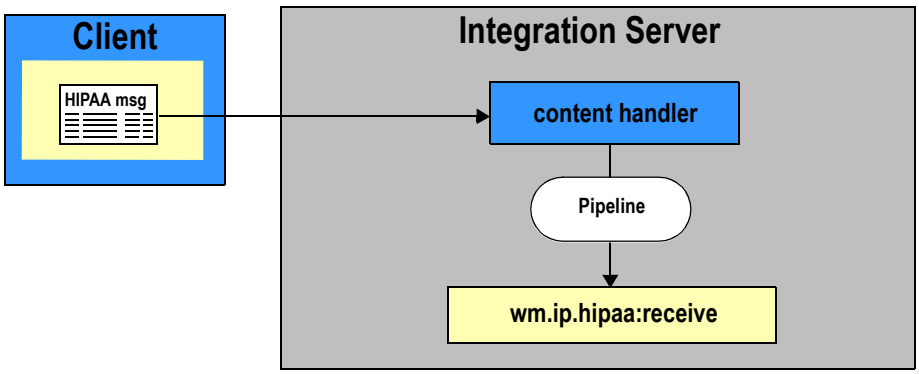
The client can use one of the following transports to send the HIPAA messages:

- HTTP or HTTPS
- FTP
- File Polling
- EDIINT AS1 or EDIINT AS2

For more information about using EDIINT, see documentation for the EDI Module. The rest of this section describes clients that use HTTP, HTTPS, FTP, or File Polling.

When a client sends a message to an Integration Server, the client must specify the content type of the data and identify the service to invoke to start the processing of the message. When the Integration Server receives the message, it passes the message to the appropriate content handler based on the specified content type, and the content handler begins the processing, which includes creating the pipeline. For more information about creating clients, see the *webMethods Developer User's Guide*.

Client sends HIPAA messages to Integration Server



Content Type to Use

The content type your client should use to send the HIPAA messages to the Integration Server depends on the type of HIPAA message that you are sending.

When your client sends....	It should use this content type
TA1 Technical Acknowledgement	application/x-wmflatfile
All other types of HIPAA messages	application/EDIStream **Note:** For backward compatibility, the EDI Module also has content handlers to accept documents with the content types application/EDI and application/X12. With these content types, the EDI Module content handler must convert the document to a String and place it in the pipeline. This can potentially consume a lot of pipeline space and use a significant amount of memory. As a result, it is recommended that you use the content type application/EDIStream because it conserves system memory.

Service the Client Invokes

After the content type handler forms the pipeline, it invokes the service that the client specifies. Your client should invoke the wm.ip.hipaa:receive service. How this service behaves depends on whether you are sending a TA1 technical acknowledgement transaction or another type of HIPAA transaction.

How wm.ip.hipaa:receive Handles TA1 Technical Acknowledgement Transactions

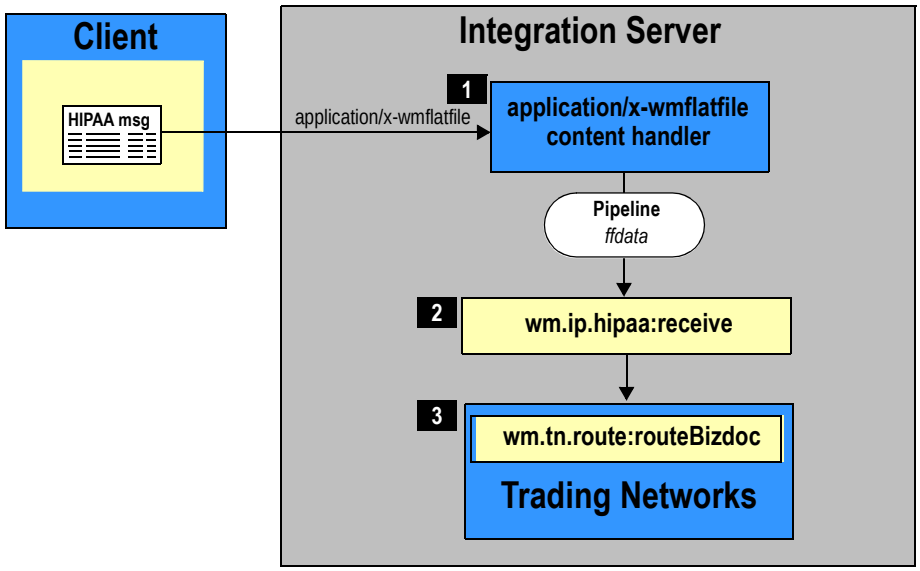
Because the EDI Module does not support TA1 technical acknowledgement transactions, the HIPAA Module adds this support. The HIPAA Module adds support by treating the TA1 technical acknowledgement as a flat file document in Trading Networks rather than an EDI document.

When the client sends a TA1 technical acknowledgement message to the Integration Server, the wm.ip.hipaa:receive service acts as a Trading Networks document gateway service. The gateway service places additional information about the TA1 technical acknowledgement in the pipeline that Trading Networks uses during its recognition processing. During recognition processing, Trading Networks matches the document to the X12 TA1 TN document type. It then proceeds with its normal processing. For more

information about Trading Networks flat file processing, see the *webMethods Trading Networks--Building Your Trading Network*.

The following diagram illustrates the processing that occurs when a client sends a TA1 technical acknowledgement to the Integration Server. For more information, see the table below the diagram.

Client sends a TA1 technical acknowledgement to the Integration Server



Step	Description
1	The client sends the HIPAA message with the content type application/x-wmflatfile to the Integration Server. The Integration Server passes the HIPAA message to the application/x-wmflatfile content handler. The content handler performs initial processing including forming the pipeline and placing the *ffdata* variable in the pipeline. The *ffdata* variable contains the HIPAA message data.

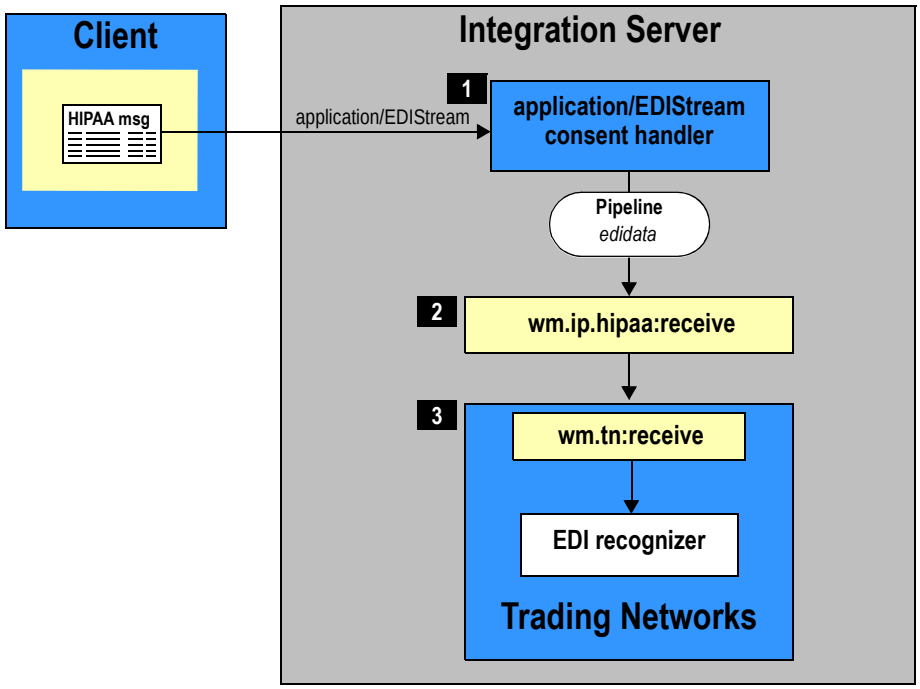
Step	Description
2	The content handler invokes the service that the client specified. For a HIPAA message, the client should specify the wm.ip.hipaa:receive service. The wm.ip.hipaa:receive service determines that the HIPAA message is a TA1 technical acknowledgement, and because it is, the wm.ip.hipaa:receive service acts as a gateway service used for Trading Networks flat file processing. The wm.ip.hipaa:receive service adds information to the pipeline. It also creates the BizDocEnvelope and sets the TN document type for the HIPAA message to X12 TA1. Then the wm.ip.hipaa:receive service invokes the wm.tn.route:routeBizdoc service.
3	The wm.tn.route:routeBizdoc service sends the HIPAA message directly to Trading Networks processing rules, bypassing document recognition. Trading Networks document recognition is bypassed because the wm.ip.hipaa:receive service already performed this function by creating the BizDocEnvelope and determining the TN document type to use for the HIPAA message.

How wm.ip.hipaa.receive Handles Other HIPAA Transactions

All other HIPAA transactions other than a TA1 technical acknowledgement are treated as regular EDI documents. When the client sends any HIPAA message other than a TA1 technical acknowledgement to the Integration Server, the wm.ip.hipaa:receive service invokes the wm.tn:receive service, which is the start of normal Trading Networks processing. Because the HIPAA message is an EDI document, Trading Networks passes the document to the EDI recognizer for processing. For more information about how EDI documents are processed in Trading Networks, see documentation for the EDI Module.

The following diagram illustrates the processing when a client sends a HIPAA message other than a TA1 technical acknowledgement to the Integration Server. For more information, see the table below the diagram.

Client sends a HIPAA message other than a TA1 technical acknowledgement to the Integration Server



Step	Description
1	The client sends the HIPAA message with the content type application/EDIStream to the Integration Server. The Integration Server passes the HIPAA message to the application/EDIStream content handler. The content handler performs initial processing including forming the pipeline and placing the *edidata* variable in the pipeline. The *edidata* variable contains the HIPAA message data.
2	The content handler invokes the service that the client specified. For a HIPAA message, the client should specify the wm.ip.hipaa:receive service. The wm.ip.hipaa:receive service determines that the HIPAA message is *not* a TA1 technical acknowledgement, and because it is *not*, the wm.ip.hipaa:receive service invokes only the wm.tn:receive service.
3	The wm.tn:receive service is the start of normal Trading Networks processing. Because the variable, *edidata*, is in the pipeline, Trading Networks passes the document to the EDI recognizer for EDI specific handling.

webMethods

CHAPTER 5

Processing HIPAA Messages Sent to the Integration Server

Overview

This chapter describes how to set up the Integration Server to process inbound HIPAA messages according to the HIPAA standard. That is how to:

- Set up the Integration Server so that your services receive an inbound HIPAA message
- Validate the HIPAA message
- Respond with the appropriate TA1 and 997 acknowledgements

Important! This chapter describes how to implement processing to meet the HIPAA standard by validating and sending appropriate acknowledgements. It does *not* include how to process the actual transactions. To process the transaction (for example, map the data from a 834 Benefit Enrollment and Maintenance transaction to a back-end system document and send it to your back-end system), you perform functions as you would for any other EDI document. To learn more about processing inbound EDI documents including how to map data from an EDI document to a back-end system document, see the documentation for the EDI Module.

Before You Can Process Inbound HIPAA Messages

Before setting up processing for inbound HIPAA message, do the following:

- **Install the TN document types and flat file schemas for the HIPAA transactions that you want to process**. For instructions, see "Step 1: Install TN Document Types for HIPAA Transactions" on page 32.
- **Define profiles for the senders and receivers identified in the ISA headers of the HIPAA messages.** For instructions, see "Step 2: Define Profiles for Trading Partners" on page 33.
- **Define EDITPA settings for the sender/receiver pairs identified in the ISA headers of the HIPAA messages.** For instructions, see "Step 4: Create EDI Trading Partner Agreements" on page 36.

Using Processing Rules to Process Inbound HIPAA Messages

As described in "Process Overview" of Chapter 1, "Concepts" in this guide, when the Integration Server receives a HIPAA message, it passes the HIPAA message to Trading Networks. Because the HIPAA message is an EDI document, Trading Networks passes the document to the EDI recognizer for EDI Module-specific recognition processing.

The EDI recognizer splits the document based on the EDITPA *splitOption* variable. To perform processing to meet HIPAA standards, you must set the *splitOption* variable to Group or Transaction, so that the EDI recognizer forms at least the Envelope and Group documents from the HIPAA message.

This section describes how to set up processing rules for the Envelope and Group documents. You should set up a one processing rule for an Envelope document and a second for a Group document.

Note: If you set the *splitOption* variable to Transaction, the EDI recognizer also will create Transaction documents that each contain a single transaction set from the HIPAA message. You define the specific processing that you want to perform against the transaction (for example, map the data to another document to send to your back-end system). This chapter does not describe how to do this processing. For more information, see the documentation for the EDI Module.

Defining a Processing Rule for an Envelope Document

To specify the processing you want to perform against the Envelope document, you define a processing rule. To meet the HIPAA standard, processing must include:

- Validating the ISA segment
- Responding with a TA1 technical acknowledgement, if appropriate

The above processing is accomplished by using the **Execute a service** processing action to invoke a service that you create. The HIPAA Module provides a sample service (wm.ip.hipaa.sample:processHipaaEnvelope) that you can use as a guideline. For more information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.

To create a processing rule for the Envelope

You create processing rules using the Trading Networks Console. For steps to create processing rules, see the chapter about processing rules in the *webMethods Trading Networks--Building Your Trading Network*.

1 **Set Processing Rule Criteria.** Set the following criteria on the **Criteria** tab of the processing rule.

Criteria tab field	Set to...
Sender	**Any Senders** You can change this to select selected senders if you want.
Receiver	**My Enterprise** It is important to select **My Enterprise** for **Receiver** so that the processing rule is invoked only when you are receiving an Envelope document (and not when sending one).
Document Type	**Selected Document Types** and select the document type **X12 Envelope** The Envelope document will match the X12 Envelope TN document type. To ensure this processing rule is invoked *only* for the Envelope document, set the criteria to specify the X12 Envelope TN document type.

2 **Set Pre-Processing Actions.** Set the following field on the **Pre-Processing** tab of the processing rule.

Pre-Processing tab field	Set to...
Validate Structure	**Validate Structure** By setting this field to validate the structure, you allow EDI Module-specific validation to proceed to perform Integrity validation of the Envelope document.

Important! If the envelope validation fails, the EDI recognizer does *not* split the Group and Transaction documents from the HIPAA message. As a result, Group and Transaction documents will *not* be processed. Only the Envelope document is passed to Trading Networks processing rules for processing, so your logic can handle the error and send a TA1 technical acknowledgement, if appropriate. For more information about how the HIPAA message is split into Envelope, Group, and/or Transaction documents, see "Process Overview" on page 17.

3 **Set Processing Actions.** On the **Action** tab of the processing rule:

a Select **Perform the following actions**.

b Select **Execute a service** and specify a service that you created to process the Envelope document. The HIPAA Module provides a sample service (wm.ip.hipaa.sample:processHipaaEnvelope) that you can use as a guideline.

For more information about the logic your service must perform to meet HIPAA standards and the built-in services provided by the HIPAA Module that you can use as guidelines, see "Processing the Envelope Document" on page 50.

Defining a Processing Rule for a Group Document

To specify the processing you want to perform against the Group document, you define a processing rule. To meet the HIPAA standard, processing must include:

- Validating the group
- Responding with a 997 functional acknowledgement, if appropriate

The above processing is accomplished by using the **Execute a service** processing action to invoke a service that you create. The HIPAA Module provides a sample service (wm.ip.hipaa.sample.processHipaaGroup) that you can use as a guideline. For more information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.

To create a processing rule for the Group

You create processing rules using the Trading Networks Console. For steps to create processing rules, see the chapter about processing rules in the *webMethods Trading Networks--Building Your Trading Network*.

1 **Set Processing Rule Criteria.** Set the following criteria on the **Criteria** tab of the processing rule.

Criteria tab field	Set to...
Sender	**Any Senders** You can change this to select selected senders if you want.

Criteria tab field	Set to...
Receiver	**My Enterprise** It is important to select **My Enterprise** for **Receiver** so that the processing rule is invoked *only* when you are receiving a Group document (and not when sending one).
Document Type	**Selected Document Types** and select the document type **X12 Group** The Group document will match the X12 Group TN document type. To ensure this processing rule is invoked *only* for the Group document, set the criteria to specify the X12 Group TN document type.

2 **Set Pre-Processing Actions.** No specific settings are required on the **Pre-Processing** tab of the processing rule for a Group document.

3 **Set Processing Actions.** On the **Action** tab of the processing rule:

   a Select **Perform the following actions**.

   b Select **Execute a service** and specify a service that you created to process the Group document. The HIPAA Module provides a sample service (wm.ip.hipaa.sample.processHipaaGroup) that you can use as a guideline.

For more information about the logic your service must perform to meet HIPAA standards and the built-in services provided by the HIPAA Module that you can use as guidelines, see "Processing the Group Document" on page 52.

Using Services to Process Inbound HIPAA Messages

This section describes the logic that you should include in the services you invoke from processing rules to process Envelope and Group documents.

Processing the Envelope Document

The following section enumerates the actions that the service processing the Envelope document must take to meet the HIPAA standard.

Note: The HIPAA Module provides a sample service (wm.ip.hipaa.sample:processHIPAAEnvelope) that you can use as a guideline. For more information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.

1 Validate the envelope.

2 Process the message based on whether envelope validation errors occur:

- **If envelope validation errors do occur**, generate a negative TA1 technical acknowledgement and save it to the Trading Networks database. To send the acknowledgement to the trading partner who sent the HIPAA message being processed, use Trading Networks delivery features. For more information, see the Trading Networks documentation.

Important! If the Validate pre-processing action determines that the envelope is *not* valid, the EDI recognizer does not split Group and Transaction documents from the HIPAA message. Only the Envelope document is passed to Trading Networks processing rules for further processing.

- **If envelope validation errors do *not* occur**, determine whether the sender requested a TA1 technical acknowledgement. If so, generate a TA1 technical acknowledgement and save it to the Trading Networks database. To send the acknowledgement to the trading partner who sent the HIPAA message being processed, use Trading Networks delivery features. For more information, see the Trading Networks documentation.

The following table lists the built-in services that are provided with the HIPAA Module to help you perform the above actions. For more information about the built-in services, see Chapter 9, "WmHipaa, WmHipaaCodeSource, and WmHipaaSample Package Services" in this guide.

Action	Built-in Service to Use
Validate the envelope	wm.ip.hipaa.util:validateEnvelope
Generate a TA1 technical acknowledgement	wm.ip.hipaa.util:generateTA1
Send the TA1 technical acknowledgement to the trading partner who sent the HIPAA message you are processing	You can send the message using Trading Networks functionality. For more information, see the Trading Networks documentation.

Processing the Group Document

The following section enumerates the actions that the service processing the Group document must take to meet the HIPAA standard.

Note: The HIPAA Module provides a sample service (wm.ip.hipaa.sample.processHipaaGroup) that you can use as a guideline. For more information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.

1. Perform HIPAA validation levels 1–2 (Integrity and Requirement validation) on the syntax of the group.

   For more information about validation, see Chapter 7, "Validating HIPAA Messages Using the HIPAA Module" in this guide.

2. Generate a 997 functional acknowledgement to report on the outcome of the validation and save it to Trading Networks.
3. Using Trading Networks, send the 997 functional acknowledgement to the trading partner who sent the HIPAA message that you are processing.
4. Perform HIPAA validation level 3–5 (Balancing, Situation, and Code Set validation) on the group.
5. Optionally, if errors were encountered, you can map the error information to an X12 824.
6. Optionally, if errors were encountered, you can generate a human-readable error report that you can send in an e-mail message to the appropriate person.
7. Optionally, you can perform HIPAA Level 6 (Product Types/Types of Service) and Level 7 (Trading Partner-Specific) validation.
8. Optionally, you might want to update your back-end system based on information in the HIPAA message. To do so, you would map data from the HIPAA message to a document that is the format that your back-end system requires, and then send the back-end system document to your back-end system. For more information about mapping data from HIPAA messages (EDI documents) to another format, see information about mapping in the documentation for the EDI Module.

The following table lists the built-in services that are provided with the HIPAA Module to help you perform the above actions. For more information about the built-in services, see Chapter 9, "WmHipaa, WmHipaaCodeSource, and WmHipaaSample Package Services" in this guide.

Action	Built-in Service to Use
Perform HIPAA validation levels 1-5 (Integrity, Requirement, Balancing, Situation, and Code Set validation) on the group.	Use the validation services for the particular transaction set. For more information about validation, see Chapter 7, "Validating HIPAA Messages Using the HIPAA Module" in this guide.
Generate a 997 functional acknowledgement	wm.ip.hipaa.util:generate997
Send the 997 functional acknowledgement to the trading partner who sent the HIPAA message you are processing.	You can send the message using Trading Networks functionality. For more information, see the Trading Networks documentation.
Generate a human-readable error report that you can send in an e-mail message to the appropriate person.	For 997 functional acknowledgement error reports, use wm.ip.hipaa.sample.util:generate997Report. For balancing, semantic, and Code Source error reports, use wm.ip.hipaa.sample.util:generateErrorReport.

webMethods

CHAPTER 6

Processing HIPAA Acknowledgements (TA1 & 997)

Overview

This chapter describes how to set up the Integration Server to process inbound HIPAA acknowledgements (TA1 and 997 transactions).

The HIPAA standard does not mandate how you are to process an acknowledgement. Typically, you would map the data from the acknowledgement to another document format, which you then can send to a back-end system.

Before You Can Process Inbound HIPAA Messages

Before setting up processing for inbound HIPAA messages, do the following:

- **Install the TN document types and flat file schemas for the X12 997 HIPAA transaction**. For instructions, see "Step 1: Install TN Document Types for HIPAA Transactions" on page 32. The X12 TA1 TN document type is automatically installed for you when you install the HIPAA Module.
- **Define profiles for the senders and receivers identified in the ISA headers of the HIPAA acknowledgements.** For instructions, see "Step 2: Define Profiles for Trading Partners" on page 33.
- **Define EDITPA settings for the sender/receiver pairs identified in the ISA headers of the HIPAA acknowledgements.** For instructions, see "Step 4: Create EDI Trading Partner Agreements" on page 36.

Defining Processing Rules for Inbound HIPAA Acknowledgements

This section describes how to set up processing rules for the TA1 and 997 HIPAA acknowledgements. You should set up a one processing rule for a TA1 technical acknowledgement and a second for a 997 functional acknowledgement.

Defining a Processing Rule for a TA1 Technical Acknowledgement

To specify the processing you want to perform against the TA1 technical acknowledgement, you define a processing rule. Typically, you will use the **Execute a service** processing action to invoke a service that you create to process the TA1 technical acknowledgement. The HIPAA Module provides the sample wm.ip.hipaa.sample:processHipaaTA1 service for you to use as a guideline.

To create a processing rule for a TA1 technical acknowledgement

You create processing rules using the Trading Networks Console. For steps to create processing rules, see the chapter about processing rules in the *webMethods Trading Networks--Building Your Trading Network*.

Note: No specific settings are required on the **Pre-Processing** tab of the processing rule for a Group document.

1 **Set Processing Rule Criteria.** Set the following criteria on the **Criteria** tab of the processing rule.

Criteria tab field	Set to...
Sender	**Any Senders** You can change this to select selected senders if you want.
Receiver	**My Enterprise** It is important to select **My Enterprise** for **Receiver** so that the processing rule is invoked only when you are receiving a TA1 technical acknowledgement (and not when sending one).
Document Type	**Selected Document Type** and select the document type **X12 TA1** The TA1 technical acknowledgement will match the X12 TA1 TN document type. To ensure this processing rule is invoked *only* for the TA1 technical acknowledgement, set the criteria to specify the X12 TA1 TN document type.

2 **Set Processing Actions.** On the **Action** tab of the processing rule, do the following:

a Select **Perform the following actions**.

b Select **Execute a service** and specify a service that you created to process the TA1 technical acknowledgement. The HIPAA Module provides the sample wm.ip.hipaa.sample:processHipaaTA1 service for you to use as a guideline.

For more information about the logic you might want to include in your service, see "Creating Services to Process HIPAA Acknowledgements" on page 59.

Defining a Processing Rule for a 997 Functional Acknowledgement

To specify the processing you want to perform against the 997 functional acknowledgement, you define a processing rule. Typically, you will use the **Execute a service** processing action to invoke a service that you created to process the 997 functional acknowledgement. The HIPAA Module provides the sample wm.ip.hipaa.sample:processHipaaFA service for you to use as a guideline. For more information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.

To create a processing rule for a 997 functional acknowledgement

You create processing rules using the Trading Networks Console. For steps to create processing rules, see the chapter about processing rules in the *webMethods Trading Networks--Building Your Trading Network*.

Note: No specific settings are required on the **Pre-Processing** tab of the processing rule for a Group document.

1 **Set Processing Rule Criteria.** Set the following criteria on the **Criteria** tab of the processing rule.

Criteria tab field	Set to...
Sender	**Any Senders** You can change this to select selected senders if you want.
Receiver	**My Enterprise** It is important to select **My Enterprise** for **Receiver** so that the processing rule is invoked only when you are receiving a 997 functional acknowledgement (and not when sending one).
Document Type	**Selected Document Types** and select the document type **X12 4010 997** The 997 functional acknowledgement will match the X12 4010 997 TN document type. To ensure this processing rule is invoked *only* for the 997 functional acknowledgement, set the criteria to specify the X12 4010 997 TN document type.

2 **Set Processing Actions.** On the **Action** tab of the processing rule, do the following:

a Select **Perform the following actions**.

b Select **Execute a service** and specify a service that you created to process the X12 4010 997 document. The HIPAA Module provides the sample wm.ip.hipaa.sample:processHipaaFA service for you to use as a guideline.

For more information about the logic you might want to include in your service, see "Creating Services to Process HIPAA Acknowledgements" on page 59.

Creating Services to Process HIPAA Acknowledgements

The HIPAA standard does not mandate how to process a TA1 technical acknowledgement or 997 functional acknowledgement. You can create your service to perform any processing you require.

The HIPAA Module provides the following sample services to illustrate how to process HIPAA acknowledgements:

- **For a TA1 technical acknowledgement**, the wm.ip.hipaa.sample:processHipaaTA1 service.
- **For a 997 functional acknowledgement**, the wm.ip.hipaa.sample:processHipaaFA service.

Processing you might want to perform is to update information in your back-end system based on the information in the acknowledgement. To do so, you would map data from the HIPAA acknowledgement to a document in the format that your back-end system requires, and then send that document to your back-end system. For more information about mapping data from HIPAA acknowledgements (EDI documents) to another format, see information about mapping in the documentation provided with the EDI Module.

webMethods

CHAPTER 7

Validating HIPAA Messages Using the HIPAA Module

Overview

The webMethods HIPAA Module validates HIPAA messages for Levels 1-5 based on the message transaction set. The HIPAA Module automatically validates documents at Level 1, Integrity. You set up validation for Levels 2-5 by configuring the processing rules that you created for each TN document type. The processing rule invokes validation services for a particular transaction set, such as X12 4010 835.

Important! Level 6 and Level 7 validation are not provided out-of-the-box. You can implement this logic using the facilities of the webMethods Integration Platform.

For more information about processing messages, see "Using Processing Rules to Process Inbound HIPAA Messages" on page 47 and "Using Services to Process Inbound HIPAA Messages" on page 50.

HIPAA Validation Levels

The HIPAA Implementation Guidelines specify seven levels of message validation. The webMethods HIPAA Module supports up to Level 5 out of the box.

Level	Description
1	**Integrity**. Validates the syntactical integrity of the X12 EDI document. Validation at this level includes testing for valid segments, segment order, element attributes, etc.
2	**Requirement**. Validates using the syntax rules defined by the HIPAA Implementation Guidelines. Validation at this level includes testing for required repeat counts, used and not used codes, required or inter-segmental data elements, etc.
3	**Balancing**. Validates to ensure for balanced field totals, record or segment counts, financial balancing of claims or remittance advice, and balancing of summary fields.
4	**Situation**. Validates for specific inter-segment situations that are described in the HIPAA Implementation Guidelines, for example, if A occurs, B must be populated. For example, if a transaction represents an accident claim, the date of the accident must be present.
5	**Code Set**. Tests that the transaction uses valid code set values as described in the HIPAA Implementation Guidelines, for example, CPT, NDC, etc.

Level	Description
6	**Product Types/Types of Service**. Validates for specific requirements needed for a specialized health care service (for example, chiropractic, durable medical equipment (DME), etc.). The special requirements are described in the HIPAA Implementation Guidelines. This validation level is not provided out-of-the-box. You can implement this logic using the facilities of the webMethods Integration Platform.
7	**Trading Partner-Specific.** Validates for requirements that are unique to a specific trading partner. These requirements are not necessarily part of the HIPAA Implementation Guidelines. This validation level is not provided out-of-the-box. You can implement this logic using the facilities of the webMethods Integration Platform.

Validating Transaction Sets

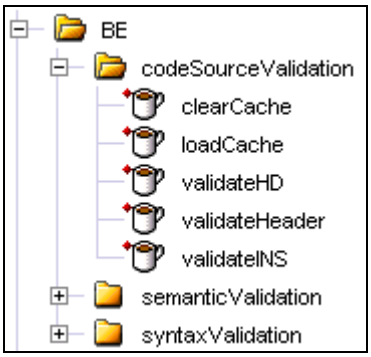
When Trading Networks receives an inbound HIPAA message, it uses a TN document type, such as X12 4010 834, to recognize the transaction type and identify the processing rule that should be used to process the transaction. The processing rule associated with this TN document type invokes the wm.hipaa.largefile:processHipaaGroup service. In turn, the processHipaaGroup service detects the transaction type and version (standard or addendum) of the message and invokes the parsing service that you have created. The parsing service then invokes the appropriate services for each level of validation that you require.

For each transaction set that the HIPAA Module supports, the HIPAA Module provides services that you can invoke to perform each level of message validation. These services are provided in the *webMethods6**IntegrationServer*\packages\WmHippaTransaction\X12\v4010*FunctionalGroupCode**validationType* folder, where *FunctionalGroupCode* is the HIPAA code for the transaction, such as BE, and *validationType* is codeSource, semantic, or syntax validation.

To view a sample of a parsing service that validates all levels of a transaction, see the parseLargeST service in the *webMethods6**IntegrationServer*\packages\WmHippaSamples\largeFile*FunctionalGroupCode* folder.

Note: All of the sample parsing services illustrate invoking all of the available types of validation. To reduce the amount of validation, you can choose not to invoke a particular validation service in the parsing service that you create.

For example, message X12 4010 834, which has a transaction set code of BE, has the sets of services illustrated in the following screen from webMethods Developer.



webMethods

CHAPTER 8

Managing HIPAA Code Sources

Overview

Important! Code sets are constantly changing. The *webMethods6\IntegrationServer* packages\WmHipaaCodeSource\data\codesets.zip file contains the version of each code set that was available when the HIPAA Module was released. Please check the webMethods Advantage Download site for the latest version of these code sets.

The HIPAA Implementation Guidelines include the External Code Sources that you use when encoding certain data elements in the transaction sets. The webMethods HIPAA Module uses these code sets to check for HIPAA compliance when exchanging documents. These *External Code Sources* include Medical and Procedural Codes as well as Non-Medical Codes. The HIPAA standard mandates that the transaction content be validated against the values in these code sources. Such validation falls under type 5 testing recommended by WEDI-SNIP.

Note: HIPAA documentation uses the terms Code Source and code set interchangeably, but code set usually refers to a set of values within a Code Source.

Each Code Source contains a group of code sets. These code sets can change from time to time, but only one version of a particular Code Source can be present in the HIPAA Module database. The HIPAA Module enables you to:

- Incorporate the code sets for each Code Source that is adopted as a HIPAA standard
- Enable or disable any Code Source
- Upgrade to new versions of existing Code Sources when they are released

The HIPAA Module provides you with 28 of the 42 Codes Sources that you may need to exchange documents. Theses files are located in the *webMethods6\IntegrationServer* packages\WmHipaaCodeSource directory. For a list of all of the HIPAA Code Sources and the names of the Codes Sources and corresponding files provided by the HIPAA Module, see Appendix A, "Code Source List" in this guide.

For steps to enable the Code Sources provided by the HIPAA Module, see "Enabling Individual Code Sources" on page 69.

If you need a Code Source not supplied by the HIPAA Module, you must acquire this Code Source, copy it into the *webMethods6\IntegrationServer*packages\ WmHipaaCodeSource directory, and then follow the steps in "Adding or Upgrading a Code Source" on page 70.

Formatting the Code Source Value File

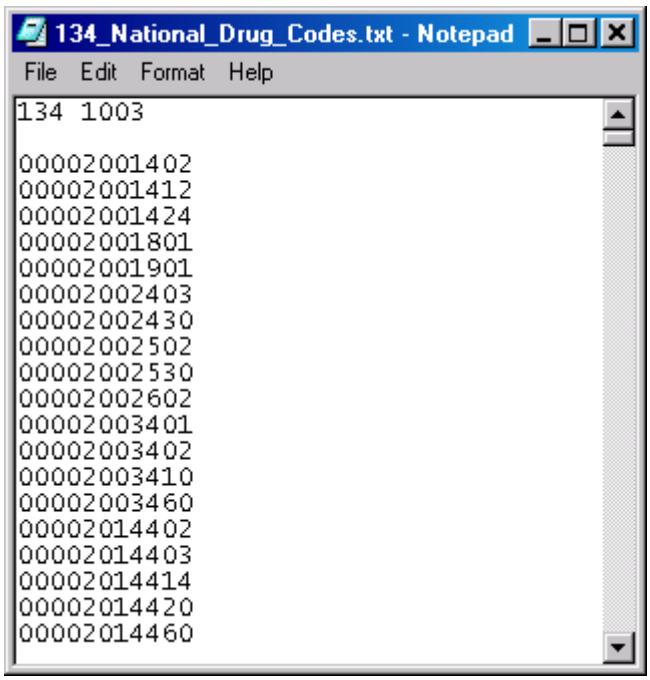
Code Source Value files come in different formats, such as HTML, Adobe Acrobat PDF, and Microsoft Word. Because it is impossible to read all of these file types and extract the values of the codes, the webMethods HIPAA Code Source Utility handles only those Code Source Value files in a .txt format. You must convert each Code Source Value file into a .txt file so that the code values can be added to the HIPAA Module database.

The first line of a Code Source Value file should have a Code Source number, followed by a space and a version number, for example, `134 1003`. If the version number is not provided with the Code Source, the version number will appear as a date in "MMYY" format.

The code values appear in the subsequent lines of the file. You should copy only the Code Source values from the original Code Source Value file into the .txt file. Do not copy the descriptions and so forth that are next to the Code Source value.

The figure below shows an example of the required format of a Code Source Value file. In addition to being in the format shown below, the Code Source Values file must be located on the system where the Integration Server is running.

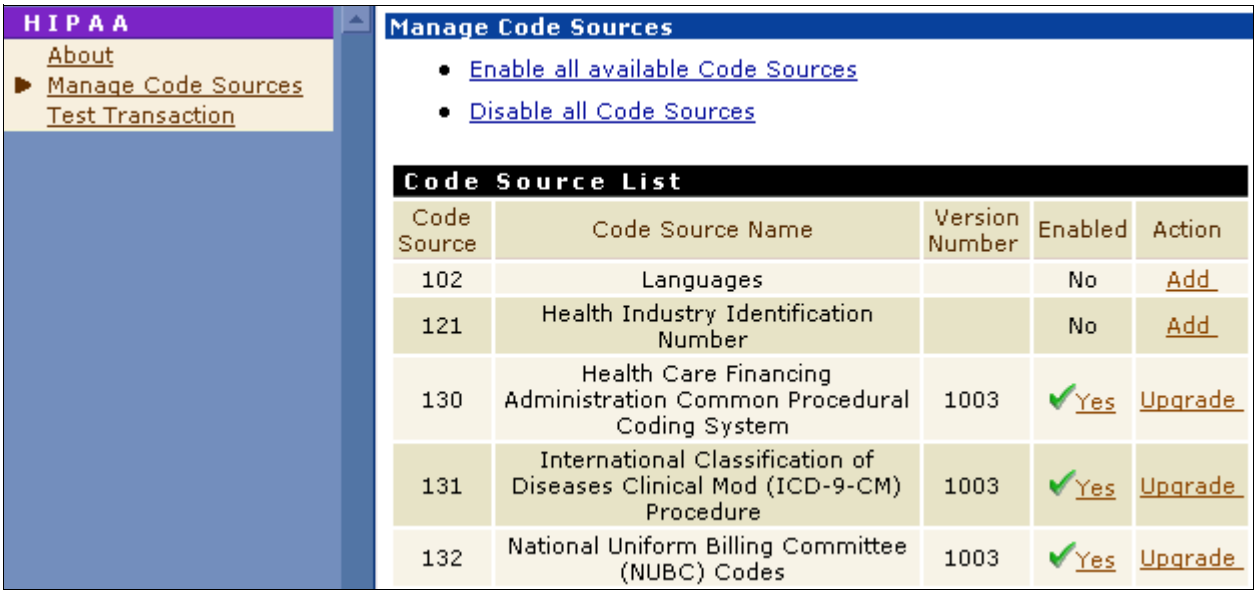
Format of the Code Source Value file



Working with Code Sources

You access the **Manage Code Sources** screen, shown in the following figure, by clicking **Manage Code Sources** on the webMethods HIPAA Module home page in the Server Administrator.

Manage Code Sources Screen



The **Manage Code Source** screen lists all available Code Sources and provides the following information for each code source:

- **Code Source**. Displays the Code Source identifier of the Code Source.
- **Code Source Name**. Displays the name of the Code Source.
- **Version Number**. Displays the version number of the Code Source. The HIPAA Module supports only one version of the Code Source in the HIPAA Module database. If the Code Source is not in the database, no value displays in this column.
- **Enabled**. Indicates whether the code source is enabled (**Yes**) or disabled (**No**).
- **Action**. If the Code Source already exists in the HIPAA Module database, this column displays **Upgrade**. If the Code Source does not exist in the database (indicated by no value in the Version column), this column displays **Add**.

Enabling Individual Code Sources

Important! To maintain performance, enable only those Code Sources that you need.

To enable an individual Code Source

1. In the Server Administrator, under **Adapters,** click **HIPAA**. On the home page of the HIPAA Module, click **Manage Code Sources**.
2. On the **Manage Code Sources** screen, in the Code Source table click **No** in the **Enabled** column for the Code Source that you want to enable. The value in the **Enabled** column for the Code Source changes to **Yes** and the Code Source is enabled.

Enabling All Code Sources

Important! To maintain performance, enable only those Code Sources that you need.

To enable all available Code Sources

1. In the Server Administrator, under **Adapters,** click **HIPAA**. On the home page of the HIPAA Module, click **Manage Code Sources**.
2. On the **Manage Code Sources** screen, click **Enable all available Code Sources**. For each available Code Source, the value in the **Enabled** column of the Code Source table changes to **Yes** and the Code Source is enabled.

Disabling Individual Code Sources

To disable an individual Code Source

1. In the Server Administrator, under **Adapters,** click **HIPAA**. On the home page of the HIPAA Module, click **Manage Code Sources**.
2. On the **Manage Code Sources** screen, in the Code Source table click **Yes** in the **Enabled** column for the Code Source that you want to disable. The value in the **Enabled** column for the Code Source changes to **No** and the Code Source is disabled.

Disabling All Code Sources

To disable all available Code Sources

1 In the Server Administrator, under **Adapters,** click **HIPAA**. On the home page of the HIPAA Module, click **Manage Code Sources**.

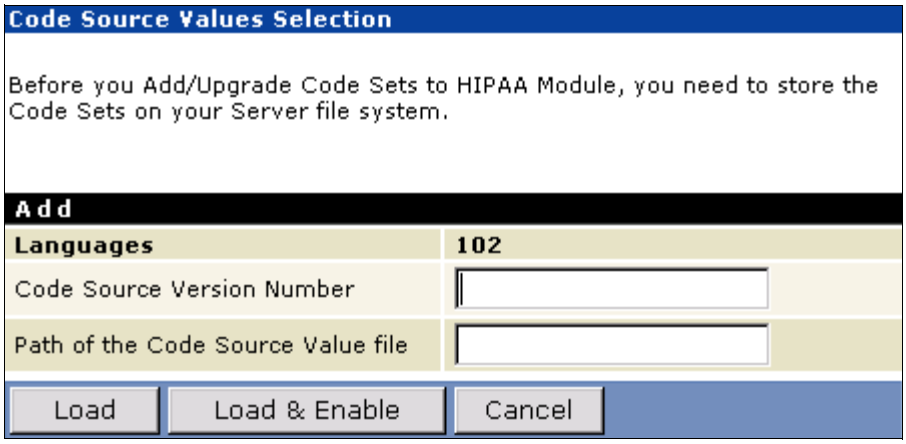
2 On the **Manage Code Sources** screen, click **Disable all code sources**. For each Code Source, the value in the **Enabled** column of the Code Source table changes to **No** and the Code Source is disabled.

Adding or Upgrading a Code Source

To add or upgrade a Code Sources

1 In the Server Administrator, under **Adapters,** click **HIPAA**. On the home page of the HIPAA Module, click **Manage Code Sources**.

2 On the **Manage Code Sources** screen, in the row listing the Code Source that you want to add or upgrade, click **Add** or **Upgrade**. The **Code Source Values Selection** screen appears.



3 In the **Code Source Version Number** field, type the version number of the Code Source.

Note: In Code Sources provided by webMethods, the version will be listed at the top of the Code Source text file.

4 In the **Path of the Code Source Value file** field, type the absolute path location of the Code Source Value file, which must be located on the system where the Integration Server is running and be in the format described in "Formatting the Code Source Value File" on page 67.

5 Click one of the following buttons:

- Click **Load** to read the Code Source from the input file, match the selected Code Source and version with the input file, and then load the Code Source (values). Although this Code Source appears on the **Manage Code Sources** screen, the **Enabled** column value for this Code Source is set to **No**. You then must click **No**, or click **Enable all available code sources** to enable the Code Source. The **Enabled** column value for this Code Source changes to **Yes**.
- Click **Load & Enable** to read the Code Source from the input file, match the selected Code Source and version with the input file, load the Code Source (values), and enable the Code Source. You are returned to the **Manage Code Sources** screen, where the **Enabled** column value for this Code Source changes to **Yes**.

webMethods

CHAPTER 9

WmHipaa, WmHipaaCodeSource, and WmHipaaSample Package Services

WmHipaa Package

wm.ip.hipaa Folder

wm.ip.hipaa:receive

This service receives, recognizes, and saves a HIPAA transaction, a 997 functional acknowledgement, or a TA1 technical acknowledgement to the webMethods Trading Networks database.

Input Variables

edidata	**String** (optional) The HIPAA real-time or batch transaction or 997 functional acknowledgement.
ffdata	**String** (optional) The HIPAA transaction or 997 functional acknowledgement.

Usage Notes

This service is used to receive a HIPAA real-time or batch transaction, a 997 functional acknowledgement or a TA1 technical acknowledgement from a trading partner. When sending the HIPAA message, the trading partner must set the content-type for the post to:

- application/x-wmflatfile when sending a TA1 technical acknowledgement
- application/EDIStream, application/EDI, or application/X12 when sending a HIPAA transaction or 997 functional acknowledgement

wm.ip.hipaa.startup

wm.ip.hipaa.startup:addHIPAAIDTypes

This service maps the sender and receiver EDI ID Qualifiers to the Trading Networks External ID Types. This is necessary for the webMethods EDI Module and Trading Networks to correctly identify the sender and receiver of the HIPAA message.

Input Variables

HIPAAIDTypes	**String List** Descriptions of the HIPAA ID Types.

Usage Notes

The Integration Server invokes this service when it loads the WmHipaa package.

wm.ip.hipaa.startup:addTA1DocType

This service creates and registers the X12 TA1 TN document type in Trading Networks. Trading Networks uses this flat file TN document type to identify the TA1 technical acknowledgement for a HIPAA X12 EDI Envelope.

Usage Notes

The Integration Server invokes this service when it loads the WmHipaa package.

wm.ip.hipaa.ui

wm.ip.hipaa.ui:addMenu

This service adds a link to the HIPAA Module home page under the **Adapters** menu on the Server Administrator.

Usage Notes

The Integration Server invokes this service when it loads the WmHipaa package.

wm.ip.hipaa.ui:getVersion

This service returns the version number, build number, and description of the given package.

Input Variables

packageName	**String** Name of the package for which to retrieve information.

Output Variables

version	**String** (optional) Version number of the package.
build	**String** (optional) Build number of the package.
description	**String** (optional) Description of the package.

Usage Notes

The Integration Server invokes this service when it loads the home page of the WmHipaa package.

wm.ip.hipaa.ui:removeMenu

This service removes the link to the HIPAA Module home page from the **Adapters** menu on the Server Administrator.

Usage Notes

The Integration Server invokes this service when it loads the WmHipaa package.

wm.ip.hipaa.util

wm.ip.hipaa.util:generate997

This service generates a 997 functional acknowledgement for a functional group, based on the syntax validation results of transactions within that functional group.

Input Variables

groupErrors	**Document** The syntax errors in the GS segment and all transaction sets within the GS.
X12env	**Document** The ISA and GS control structure for the EDI document being acknowledged.
delimiters	**Document** The segment, element, and composite delimiters for the EDI document being acknowledged.

Output Variables

997edidata	**String** The 997 functional acknowledgement data string.

Usage Notes

You can use this service to generate a 997 functional acknowledgement for a functional group after syntactically validating transactions within that functional group.

wm.ip.hipaa.util:generateTA1

This service generates a technical acknowledgement based on the results of validating an interchange envelope.

Input Variables

envelopeErrors	**Document** The results of envelope validation.
X12env	**Document** The ISA control structure for the EDI Envelope being acknowledged.

Output Variables

ta1data	**String** The TA1 technical acknowledgement data string.

wm.ip.hipaa.util:sortErrors

This service takes wm.ip.hipaa.rec:errorArray as input and sorts the array based on the *errorArray* and *errorPosition* variables. The *errorArray* can have *childErrors* (which is again of type wm.ip.hipaa.rec:errorArray). You can use this service to sort *childErrors* for every error in the *errorArray*.

Input Variables

errors	**Document List** Syntax, Semantic, Balancing, or Code Source errors to be sorted.

Output Variables

sortedErrors	**Document List** The sorted errors.

Usage Notes

You can use this service to sort the error array returned as a result of a validation service.

wm.ip.hipaa.util:validateEnvelope

This service determines whether the interchange envelope is valid or invalid. If the envelope is invalid, it outputs the errors in the envelope.

Input Variables

edidata	**String** The *edidata* string for which the envelope has to be validated.

Output Variables

X12env	**Document** The parsed interchange envelope for the functional group.
envelopeErrors	**Document** Errors in the envelope.

Usage Notes

You can use this service to validate an X12 Envelope document submitted to Trading Networks.

wm.ip.hipaa.util:validateGroup

This service validates the syntax for transactions within a functional group.

Input Variables

X12env	**Document** The parsed interchange envelope for the functional group.

Output Variables

groupErrors	**Document** Errors in the group as well as transaction sets within the group.
addendum	**String** Whether the transaction sets in the group belong to the addendum version or standard implementation. It will be one of the following:

Value	Meaning
`true`	The transaction sets in the group belong to the addendum version.
`false`	The transaction sets in the group belong to the standard implementation version.

groupType	**String** The functional group code.
transactionType	**String** The transaction identifier for the functional group code.

Usage Notes

This service can be used to validate an X12 Group document submitted to Trading Networks.

WmHipaaCodeSource Package

wm.ip.hipaa.codesource:activateCodeSource

This service enables a particular Code Source

Input Variables

status	**String** Whether the Code Source has to be enabled or disabled, or whether all Code Sources should be enabled or disabled.
codeSource	**Document** If you are enabling only one Code Source, this is the name of the Code Source.

Output Variables

error	**Document** Errors encountered when enabling or disabling the Code Source.

Usage Notes

This service is invoked when you use the **Manage Code Sources** page on the HIPAA Module home page in the Server Administrator to enable a Code Source. When you enable a Code Source, the HIPAA transaction set will be validated against the values in this Code Source.

wm.ip.hipaa.codesource:codesource

This service displays a list of each Code Source and its status (**Enabled** or **Disabled**) that is available to **Add** or **Upgrade**. If the database already contains the values for a Code Source, that Code Source is available to **Upgrade**, which loads a new version of the Code Source into the database. If the database does not have values for this Code Source, the Code Source is available to **Add**.

Output Variables

codeSourceList	**Document List** List of all available Code Sources and their statuses, whether these are available to **Add** or **Upgrade**, and whether they are **Enabled** or **Disabled**.

Usage Notes

This service is invoked when you use the **Manage Code Sources** page on the HIPAA Module home page in the Server Administrator to display a list of each Code Source and its status (**Enabled** or **Disabled**) that is available for **Add** or **Upgrade**. If a Code Source is disabled, the HIPAA transaction set will not be validated against the values in this Code Source.

wm.ip.hipaa.codesource:loadCodeSourceValues

This service loads Code Source values for a given Code Source into the database tables.

Input Variables

filename	**String** Code Source file name.
codeSource	**String** Code Source number.
codeSourceName	**String** Code Source name.
version	**String** Code Source version.
type	**String** Whether you are installing the Code Source for the first time or the Code Source is being upgraded. It will be one of the following:

Input Variables

	Value	Meaning
	`add`	You are installing the Code Source for the first time.
	`upgrade`	You are upgrading the Code Source.

enable **String** Whether the Code Source should be enabled now. It will be one of the following:

Value	Meaning
`now`	Enable the Code Source now.
`later`	Do not enable the Code Source now.

Output Variables

errorLog **Document** Errors encountered when loading the Code Source.

Usage Notes

This service is invoked when you use the **Manage Code Sources** page on the HIPAA Module home page in the Server Administrator to load or load and enable a Code Source.

WmHipaaSample Package

For information about the services themselves, see the webMethods Developer. For information about the sample, see Appendix B, "webMethods HIPAA Module Sample" in this guide.

webMethods

CHAPTER 10

WmHipaaTransactions Package Services

wm.ip.hipaa.transaction.X12.V4010.BE

This folder includes services related to the HIPAA 834 Benefit Enrollment and Maintenance transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.BE.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.BE.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.BE.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.BE.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.BE.codeSourceValidation:validateHeader

Use this service to find code source errors in the Header of the 834 transaction set. The Header contains segments ST, BGN, REF, DTP list, and N1 list.

Input Variables

Header834	**Document** Header segment data to validate.

Output Variables

codeSourceErrors	**Document List** Code source errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.codeSourceValidation:validateHD

Use this service to find code source errors in the HD segment list of the 834 transaction set.

Input Variables

HDDT	**Document** List of segments to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the HD segment list in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.codeSourceValidation:validateINS

Use this service to find code source errors in the INS segment of the 834 transaction set.

Input Variables

MemberDetail	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the INS segment in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.BE.semanticValidation:validateHD

Use this service to find semantic errors in the HD segment list of the 834 transaction set.

Input Variables

HDDT	**Document** List of segments to validate.
INS01	**String** Value of the INS01 element.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the HD segment list in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.semanticValidation:validateINS

Use this service to find semantic errors in the INS segment of the 834 transaction set.

Input Variables

MemberDetail	**Document** Segment to validate.
BGN08	**String** Value of the BGN08 element.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the INS segment in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.BE.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 834 transaction set. The Header contains segments ST, BGN, REF, DTP list, and N1 list.

Input Variables

Header834	**Document** Header segment data to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

BGN08	**String** Value of the BGN08 element.
syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.syntaxValidation:validateHD

Use this service to find syntax errors in the HD segment list of the 834 transaction set.

Input Variables

HDDT	**Document** List of segments to validate.
INS01	**String** Value of the INS01 element.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the the transaction set errors.

Usage Notes

You can use this service to validate the HD segment list in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.syntaxValidation:validateINS

Use this service to find syntax errors in the INS segment of the 834 transaction set.

Input Variables

MemberDetail	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Input Variables

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the INS segment in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.BE.syntaxValidation:validateBGN

Use this service to find syntax errors in the BGN segment of the 834 transaction set.

Input Variables

BGNDT **Document** List of segments to validate.

Output Variables

BGNErrors **Document List** BGN syntax errors for the segment being validated.

BGN08 **String** Value of the BGN08 element.

Usage Notes

You can use this service to validate the BGN segment in an 834 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 834 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB

This folder includes services related to the HIPAA 271 Health Care Eligibility Response transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HB.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HB.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HB.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HB.codeSourceValidation:validateReceiver

Use this service to find code source errors in the receiver of the 271 transaction set.

Input Variables

Receiver_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the receiver HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.codeSourceValidation:validateSubscriber

Use this service to find code source errors in the subscriber of the 271 transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the subscriber HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.codeSourceValidation:validateDependent

Use this service to find code source errors in the Dependent loop of the 271 transaction set.

Input Variables

Dependent_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the dependent HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.codeSourceValidation:validateEB

Use this service to find code source errors in the EB loop of the 271 transaction set.

Input Variables

EB	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the EB loop segments in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HB.semanticValidation:validateReceiver

Use this service to find semantic errors in the receiver HL segment of the 271 transaction set.

Input Variables

Receiver_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the receiver HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.semanticValidation:validateSubscriber

Use this service to find semantic errors in the subscriber HL segment of the 271 transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the subscriber HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.semanticValidation:validateDependent

Use this service to find semantic errors in the dependent HL segment of the 271 transaction set.

Input Variables

Dependent_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the dependent HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.semanticValidation:validateEB

Use this service to find semantic errors in the EB loop segment of the 271 transaction set.

Input Variables

EB	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the EB loop segments in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 271 transaction set. The Header contains segments ST and BHT.

Input Variables

Header271	**Document** Header segment data to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation:validateSource

Use this service to find syntax errors in the source HL segment of the 271 transaction set.

Input Variables

sourceData	**String** HL Segment data of the source.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Input Variables

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Source_HL	**Document** Segment as output for next service.
syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segment being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the unrecognized and unexpected segment errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the source HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation:validateReceiver

Use this service to find syntax errors in the Receiver segments of the 271 transaction set.

Input Variables

receiverData	**String** HL Segment data of the Receiver.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segment being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the unrecognized and unexpected segment errors are appended to the transaction set errors.
Receiver_HL	**Document** Segment as output for the next service

Usage Notes

You can use this service to validate the receiver segments in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber HL segment of the 271 transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber HL segment in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation:validateDependent

Use this service to find syntax errors in the Dependent loop segment of the 271 transaction set.

Input Variables

Dependent_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent loop segments in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HB.syntaxValidation:validateEB

Use this service to find syntax errors in the EB loop segment of the 271 transaction set.

Input Variables

EB	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Input Variables

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

dependentExists **String** Indicates whether the dependent 2000D loop exists.

Value	Meaning
`true`	The dependent exists.
`false`	The dependent does *not* exist.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the EB loop segments in a 271 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 271 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC

This folder includes services related to the HIPAA 837-I Health Care Claim (Institutional) transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation:validatePayToProvider

Use this service to find code source errors in the PayToProvider HL of the 837-I transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the PayToProvider HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation:validatePatient

Use this service to find code source errors in the Patient HL segment of the 837-I transaction set.

Input Variables

Patient_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Patient HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation:validateSubscriber

Use this service to find code source errors in the Subscriber HL segment of the 837-I transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation:validateClaim

Use this service to find code source errors in the CLM segment of the 837-I transaction set.

Input Variables

Claim	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the CLM segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation:validateLX

Use this service to find code source errors in the LX segment of the 837-I transaction set.

Input Variables

Claim_LX	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the LX segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.codeSourceValidation:validateOtherSBR

Use this service to find code source errors in the SBR segment of the 837-I transaction set.

Input Variables

Claim_SBR	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SBR segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation:validatePayToProvider

Use this service to find semantic errors in the PayToProvider HL segment of the 837-I transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Output Variables

PRV_2000A_exists **String** Indicates whether the PRV segment in the 2000 loop exists.

Value	Meaning
`true`	The PRV segment exists.
`false`	The PRV segment does *not* exist.

Usage Notes

You can use this service to validate the PayToProvider HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation:validateSubscriber

Use this service to find semantic errors in the Subscriber HL segment of the 837-I transaction set

Input Variables

Subscriber_HL **Document** Segment to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation:validateClaim

Use this service to find semantic errors in the CLM segment of the 837-I transaction set.

Input Variables

Claim **Document** Segment to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

Input Variables

PRV_2000A_exists	**String** Indicates whether the PRV segment in the 2000 loop exists.

Value	Meaning
`true`	The PRV segment exists.
`false`	The PRV segment does *not* exist.

semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the CLM segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation:validateLX

Use this service to find semantic errors in the LX segment of the 837-I transaction set.

Input Variables

Claim_LX	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
NM109_PR_2330B	**String** Value of the NM109_PR_2330B element.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the LX segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.semanticValidation:validateOtherSBR

Use this service to find semantic errors in the SBR segment of the 837-I transaction set.

Input Variables

Claim_SBR	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

NM109_PR_2330B **String** Value of the NM109_PR_2330B element.

MIA_Present **String** Whether the MIA segment is present.

Value	Meaning
`true`	The MIA segment is present.
`false`	The MIA segment is *not* present.

MOA_Present **String** Whether the MOA segment is present.

Value	Meaning
`true`	The MOA segment is present.
`false`	The MOA segment is *not* present.

Usage Notes

You can use this service to validate the SBR segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 837-I transaction set. The Header contains segments ST, BHT, REF, and N1 list.

Input Variables

Header837I	**Document** Header segment data to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validatePayToProvider

Use this service to find syntax errors in the PayToProvider HL segment of the 837-I transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the PayToProvider HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber HL segment of the 837-I transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validatePatient

Use this service to find syntax errors in the Patient HL segment of the 837-I transaction set.

Input Variables

Patient_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Input Variables

addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Patient HL segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validateClaim

Use this service to find syntax errors in the CLM segment of the 837-I transaction set.

Input Variables

Claim	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the CLM segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validateOtherSBR

Use this service to find syntax errors in the Claim SBR segment of the 837-I transaction set.

Input Variables

Claim_SBR **Document** Segment to validate.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Claim SBR segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validateX096LX

Use this service to find syntax errors in the LX segment of the 837-I transaction set.

Input Variables

Claim_LX **Document** Segment to validate.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

Input Variables

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the LX segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HC.syntaxValidation:validateX096A1LX

Use this service to find syntax errors in the addenda LX segment of the 837-I transaction set.

Input Variables

Claim_LX	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the addenda LX segment in an 837-I transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 837-I transaction set.

wm.ip.hipaa.transaction.X12.V4010.HCX097

This folder includes services related to the HIPAA 837-D Health Care Claim (Dental) transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation:validatePayToProvider

Use this service to find code source errors in the PayToProvider HL of the 837-D transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X097:PayToProvider_HLDT IS document type.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL. This service is used to validate the Code Sources mentioned in the 837-D IG and handles both the Direct and Indirect Code Sources. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction set.

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation:validatePatient

Use this service to find code source errors in the Patient HL segment of the 837-D transaction set.

Input Variables

Patient_HL	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Patient HL segment in 837-D transaction set. The WmHipaaSample package demonstrates how this service is used when validating 837-D transaction set.

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation:validateSubscriber

Use this service to find code source errors in the Subscriber HL segment of the 837-D transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate. This IData object conforms to HIPAAFFSchema.X12.V4010.HC.X097:Subscriber_HLDT IS document type.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL. This service is used to validate the Code Sources mentioned in the 837-D IG and handles both the Direct and Indirect Code sources. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction set.

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation:validateClaim

Use this service to find code source errors in the CLM segment of the 837-D transaction set

Input Variables

Claim	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation:validateOtherSBR

Use this service to find code source errors in the SBR segment of the 837-D transaction set.

Input Variables

Claim_SBR	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.codeSourceValidation:validateLX

Use this service to find the code source errors in the LX segment of the 837-D transaction set

Input Variables

Claim_LX	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation:validateHeader

Use this service to find semantic errors in the Header of the 837-D transaction set.

Input Variables

Header837D	**Document** Segment to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Header segment in 837-D transaction set. The WmHipaaSample package demonstrates how this service is used when validating 837-D transaction set.

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation:validatePayToProvider

Use this service to find semantic errors in the PayToProvider HL segment of the 837-D transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X097:PayToProvider_HLDT IS document type.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.
PRV_2000A_exists	**String** Indicates whether the PRV segment in the 2000 loop exists.

Value	Meaning
`true`	The PRV segment exists.
`false`	The PRV segment does *not* exist.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL to semantically validate the PayToProvider Information for 837-D transaction set. For example, if NM108 has a value of "XX - NPI", then either the Employer's Identification Number, Social Security Number, or Federal Tax Identification Number of the Provider must be carried in the REF in this loop. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation:validateSubscriber

Use this service to find semantic errors of Subscriber HL segment of the 837-D transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X097:Subscriber_HLDT IS document type.

Input Variables

semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL to semantically validate the Subscriber information for 837-D transaction set. For example, SBR02 should not be used when HL03 is 22 and HL04 is 1; NM104 should not be null when NM102 is 1 and NM101 is IL; and DMG should not be null when SBR02 is 18 and NM101 is IL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation:validateClaim

Use this service to find semantic errors in the CLM segment of the 837-D transaction set.

Input Variables

Claim	**Document** Segment to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
PRV_2000A_exists	**String** Indicates whether the PRV segment in the 2000 loop exists.

Value	Meaning
`true`	The PRV segment exists.
`false`	The PRV segment does *not* exist.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Input Variables

NM1_2010AB_nonperson	**String** Indicates whether the NM1 segment is a non-person.

Value	Meaning
`true`	The NM1 segment is a non-person.
`false`	The NM1 segment is not a non-person.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation:validateOtherSBR

Use this service to find semantic errors in the SBR segment of the 837-D transaction set.

Input Variables

Claim_SBR	**Document** Segment to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.
NM108_PR_2330B	**String** Contains the value of NM108 element if the NM101 element in 2330B of NM1 segment is PR.
NM109_PR_2330B	**String** Contains the value of NM109 element if the NM101 element in 2330B of NM1 segment is PR.

Output Variables

DTP_2330B_exist **String** Indicates whether the DTP segment in the 2330B loop exists.

Value	Meaning
`true`	The DTP segment exists.
`false`	The DTP segment does *not* exist.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.semanticValidation:validateLX

Use this service to find semantic errors in the LX segment of the 837-D transaction set

Input Variables

Claim_LX **Document** Segment to validate.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

NM108_PR_2330B **String** Contains the value of NM108 element if the NM101 element in 2330B of NM1 segment is PR.

NM109_PR_2330B **String** Contains the value of NM109 element if the NM101 element in 2330B of NM1 segment is PR.

DTP_2330B_exist **String** Indicates whether the DTP segment in the 2330B loop exists.

Value	Meaning
`true`	The DTP segment exists.
`false`	The DTP segment does *not* exist.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 837-D transaction set.

Input Variables

Header837D **Document** Header segment data to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HCX097:Header837DDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseLargeST. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validatePayToProvider

Use this service to find syntax errors in the PayToProvider HL segment of the 837-D transaction set.

Input Variables

PayToProvider_HL **Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X097:PayToProvider_HLDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

Input Variables

NM1_2010AB_nonperson **String** Indicates whether the NM1 segment is a non-person.

Value	Meaning
`true`	The NM1 segment is a non-person.
`false`	The NM1 segment is not a non-person.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

NM1_2010AB_nonperson **String** Indicates whether the NM1 segment is a non-person.

Value	Meaning
`true`	The NM1 segment is a non-person.
`false`	The NM1 segment is not a non-person.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber HL segment of the 837-D transaction set.

Input Variables

Subscriber_HL **Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X097:Subscriber_HLDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

Input Variables

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validatePatient

Use this service to find syntax errors in the Patient HL segment of the 837-D transaction set.

Input Variables

Patient_HL **Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X097:Patient_HLDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateClaim

Use this service to find syntax errors in the CLM segment of the 837-D transaction set

Input Variables

Claim	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX097:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateOtherSBR

Use this service to find syntax errors in the SBR segment of the 837-D transaction set.

Input Variables

Claim_SBR	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Input Variables

hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateClaim. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateX097LX

Use this service to find syntax errors in the LX segment of the 837-D transaction set.

Input Variables

Claim_LX	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateClaim. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateX097A1LX

Use this service to find syntax errors in the LX segment of the 837-D transaction set when addendum value is true

Input Variables

Claim_LX	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.transaction.X12.V4010.HCX097.syntaxValidation:validateClaim. The WmHipaaSample package demonstrates how this service is used when validating an 837-D transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098

This folder includes services related to the HIPAA 837-P Health Care Claim (Professional) transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validatePayToProvider

Use this service to find code source errors in the PayToProvider HL segment of the 837-P transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:PayToProvider_HLDT IS document type.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. This service is used to validate the Code Sources mentioned in the 837P IG and handles both the Direct and Indirect Code Sources. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validateSubscriber

Use this service to find code source errors in the Subscriber HL segment of the 837-P transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:Subscriber_HLDT IS document type.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. This service is used to validate the Code Sources mentioned in the 837P IG and handles both the Direct and Indirect Code Sources. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validatePatient

Use this service to find code source errors in the Patient loop of the 837-P transaction set.

Input Variables

Patient_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:Patient_HLDT IS document type.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validateClaim

Use this service to find code source errors in the CLM segments of the 837-P transaction set.

Input Variables

Claim	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validateOtherSBR

Use this service to find code source errors in the SBR segments of the 837-P transaction set.

Input Variables

Claim_SBR	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validateLX

Use this service to find code source errors in the LX segments of the 837-P transaction set.

Input Variables

Claim_LX	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.codeSourceValidation:validateX098A1LX

Use this service to find code source errors in the CLM segments of the 837-P transaction set addendum version.

Input Variables

Claim_LX	**Document** Segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validateHeader

Use this service to find semantic errors in the Header of the 837-P transaction set.

Input Variables

Header837P	**Document** Header segment data to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:Header837PDT IS document type.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseLargeST. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validatePayToProvider

Use this service to find semantic errors in the PayToProvider HL segment of the 837-P transaction set.

Input Variables

PayToProvider_HL	**Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HCX098:PayToProvider_HLDT IS document type.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Output Variables

PRV_2000A_exists **String** Indicates whether the PRV segment in the 2000 loop exists.

Value	Meaning
`true`	The PRV segment exists.
`false`	The PRV segment does *not* exist.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validateSubscriber

Use this service to find semantic errors in the Subscriber HL segment of the 837-P transaction set.

Input Variables

Subscriber_HL **Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:Subscriber_HLDT IS document type

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction. For example, SBR02 should not be used when HL03 is 22 and HL04 is 1; NM104 should not be null when NM102 is 1 and NM101 is IL; and DMG should not be null when SBR02 is 18 and NM101 is IL.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validatePatient

Use this service to find semantic errors in the Patient segment of the 837-P transaction set.

Input Variables

Patient_HL	**Document** Segment to validate. This IData object conforms to HIPAAFFSchema.X12.V4010.HC.X098:Patient_HLDT IS document type.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validateClaim

Use this service to find semantic errors in the CLM segment of the 837-P transaction set.

Input Variables

Claim **Document** Segment to validate.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

PRV_2000A_exists **String** Indicates whether the PRV segment in the 2000 loop exists.

Value	Meaning
`true`	The PRV segment exists.
`false`	The PRV segment does *not* exist.

Input Variables

NM1_2010AB_nonperson **String** Indicates whether the NM1 segment is a non-person.

Value	Meaning
`true`	The NM1 segment is a non-person.
`false`	The NM1 segment is not a non-person.

Pat09 **String** Value of the incoming document for pat09.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

HI_2300_exists **String** Indicates whether the HI segment in the 2300 loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

CR212_2300_exists **String** Indicates whether the CR212 segment in the 2300 loop exists.

Value	Meaning
`true`	The CR212 segment exists.
`false`	The CR212 segment does *not* exist.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validateOtherSBR

Use this service to find semantic errors in the SBR segment of the 837-P transaction set.

Input Variables

Claim_SBR **Document** Segment to validate.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.
NM108_PR_2330B	**String** Value of the NM108 element in the 2330B loop.
NM109_PR_2330B	**String** Value of the NM109 element in the 2330B loop.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098: parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validateX098LX

Use this service to find semantic errors in the LX segment of the 837-P transaction set.

Input Variables

Claim_LX	**Document** Segment to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
HI_2300_exists	**String** Indicates whether the HI segment in the 2300 loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

NM108_PR_2330B	**String** Value of the NM108 element in the 2330B loop.
NM109_PR_2330B	**String** Value of the NM109 element in the 2330B loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098: parseHL. This service validates the semantic validation of the entire Line Counter information loop. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.semanticValidation:validateX098A1LX

Use this service to find semantic errors in the LX segment of the 837-P transaction set addendum version.

Input Variables

Claim_LX **Document** Segment to validate.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

HI_2300_exists **String** Indicates whether the HI segment in the 2300 loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

NM108_PR_2330B **String** Value of the NM108 element in the 2330B loop.

NM109_PR_2330B **String** Value of the NM109 element in the 2330B loop.

CR212_2300_exists **String** Indicates whether the CR212 segment in the 2300 loop exists.

Value	Meaning
`true`	The CR212 segment exists.
`false`	The CR212 segment does *not* exist.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098: parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header segment of the 837-P transaction set.

Input Variables

Header837P **Document** Header segment data to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:Header837PDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the Header being validated.

addendum **String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the Header being validated.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseLargeST. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validatePayToProvider

Use this service to find syntax errors in the PayToProvider HL segment of the 837-P transaction set.

Input Variables

PayToProvider_HL **Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:PayToProvider_HLDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the Header being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

Input Variables

addendum **String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

NM1_2010AB_nonperson **String** Indicates whether the NM1 segment is a non-person.

Value	Meaning
`true`	The NM1 segment is a non-person.
`false`	The NM1 segment is not a non-person.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

NM1_2010AB_nonperson **String** Indicates whether the NM1 segment is a non-person.

Value	Meaning
`true`	The NM1 segment is a non-person.
`false`	The NM1 segment is not a non-person.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber HL segment of the 837-P transaction set.

Input Variables

Subscriber_HL **Document** Segment to validate. This IData object conforms to the HIPAAFFSchema.X12.V4010.HC.X098:Subscriber_HLDT IS document type.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

Input Variables

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
Pat09	**String** Value of the incoming document for pat09.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validatePatient

Use this service to find syntax errors in the Patient HL segment of the 837-P transaction set.

Input Variables

Patient_HL	**Document** Segment to validate. This IData object conforms to HIPAAFFSchema.X12.V4010.HC.X098:Patient_HLDT IS document type.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
Pat09	**String** Value of the incoming document for pat09.

Usage Notes

This service is called by wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098Claim

Use this service to find syntax errors in the CLM segment of the 837-P transaction set.

Input Variables

Claim	**Document List** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098A1Claim

Use this service to find syntax errors in the CLM segment of the 837-P transaction set addendum version.

Input Variables

Claim	**Document List** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Input Variables

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.sample.largeFile.HCX098:parseHL. This service validates the syntax validation of all the Claim information loop segments. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateOtherSBR

Use this service to find syntax errors in the SBR segment of the 837-P transaction set.

Input Variables

Claim_SBR	**Document List** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098Claim. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098LX

Use this service to find syntax errors in the LX segment of the 837-P transaction set.

Input Variables

Claim_LX	**Document List** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098Claim. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098A1LX

Use this service to find syntax errors in the LX segment of the 837-P transaction set when addendum value is true.

Input Variables

Claim_LX	**Document List** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

This service is called from wm.ip.hipaa.transaction.X12.V4010.HCX098.syntaxValidation:validateX098A1Claim. The WmHipaaSample package demonstrates how this service is used when validating an 837-P transaction.

wm.ip.hipaa.transaction.X12.V4010.HIReq

This folder includes services related to the HIPAA 278 Health Care Services Review Request transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HIReq.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HIReq.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HIReq.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HIReq.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HIReq.codeSourceValidation:validateX094A1Requester

Use this service to find code source errors in the Requester loop of the 278 Request transaction set.

Input Variables

Requester_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Requester loop in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.codeSourceValidation:validateX094A1Subscriber

Use this service to find code source errors in the Subscriber loop of the 278 Request transaction set.

Input Variables

Subscriber_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber loop in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.codeSourceValidation:validateX094A1Dependant

Use this service to find code source errors in the Dependent loop of the 278 Request transaction set.

Input Variables

Dependent_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent loop in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.codeSourceValidation:validateX094A1Provider

Use this service to find code source errors in the Provider loop of the 278 Request transaction set.

Input Variables

Provider_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider loop in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.codeSourceValidation:validateX094A1Service

Use this service to find code source errors in the Service loop of the 278 Request transaction set.

Input Variables

Service_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Service loop in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HIReq.semanticValidation:validateX094A1Requester

Use this service to find semantic errors in the Requester part of the 278 Request transaction set.

Input Variables

Requester_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Requester information in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.semanticValidation:validateX094A1Subscriber

Use this service to find semantic errors in the Subscriber part of the 278 Request transaction set.

Input Variables

subscriber_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

DTP_439_2000C_Exists **String** Indicates whether the DTP (DTP01=439) segment in the 2000C loop exists.

Value	Meaning
`true`	The DTP (DTP01=439) segment exists.
`false`	The DTP (DTP01=439) segment does *not* exist.

HI_2000C_Exists **String** Indicates whether the HI segment in the 2000C loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

HI_2000C_BK _Exists **String** Indicates whether the HI (HI01 = BK) segment in the 2000C loop exists.

Value	Meaning
`true`	The HI (HI01 = BK) segment exists.
`false`	The HI (HI01 = BK) segment does *not* exist.

HI01_4_2000C_Exists **String** Indicates whether the HI01-4 value in the 2000C loop exists.

Value	Meaning
`true`	The HI01-4 value exists.
`false`	The HI01-4 value does *not* exist.

PWK_2000C_Exists **String** Indicates whether the PWK segment in the 2000C loop exists.

Value	Meaning
`true`	The PWK segment exists.
`false`	The PWK segment does *not* exist.

REF_EJ_Exists **String** Indicates whether the REF (REF01=EJ) segment in the 2000C loop exists.

Value	Meaning
`true`	The REF (REF01=EJ) segment exists.
`false`	The REF (REF01=EJ) segment does *not* exist.

Usage Notes

You can use this service to validate the Subscriber information in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.semanticValidation:validateX094A1Dependant

Use this service to find semantic errors in the Dependent part of the 278 Request transaction set.

Input Variables

Dependent_HL **Document** List of segments to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

PWK_2000C_Exists **String** Indicates whether the PWK segment in the 2000C loop exists.

Value	Meaning
`true`	The PWK segment exists.
`false`	The PWK segment does *not* exist.

HI_2000C_Exists **String** Indicates whether the HI segment in the 2000C loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

REF_EJ_Exists **String** Indicates whether the REF (REF01=EJ) segment in the 2000C loop exists.

Value	Meaning
`true`	The REF (REF01=EJ) segment exists.
`false`	The REF (REF01=EJ) segment does *not* exist.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Output Variables

DTP_439_2000D_Exists **String** Indicates whether the DTP (DTP01=439) segment in the 2000D loop exists.

Value	Meaning
`true`	The DTP (DTP01=439) segment exists.
`false`	The DTP (DTP01=439) segment does *not* exist.

HI_2000D_Exists **String** Indicates whether the HI segment in the 2000D loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

HI_2000D_BK_Exists **String** Indicates whether the HI (HI01 = BK) segment in the 2000D loop exists.

Value	Meaning
`true`	The HI (HI01 = BK) segment exists.
`false`	The HI (HI01 = BK) segment does *not* exist.

HI01_4_2000D_Exists **String** Indicates whether the HI01-4 value in the 2000D loop exists.

Value	Meaning
`true`	The HI01-4 value exists.
`false`	The HI01-4 value does *not* exist.

Usage Notes

You can use this service to validate the Dependent information in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.semanticValidation:validateX094A1Provider

Use this service to find semantic errors in the Provider part of the 278 Request transaction set.

Input Variables

Provider_HL **Document** List of segments to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider information in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.semanticValidation:validateX094A1Service

Use this service to find semantic errors in the Service part of the 278 Request transaction set.

Input Variables

Service_HL **Document** List of segments to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

HL_2000F_Count **String** Value of 2000F HL segments in the transaction set.

HI01_4_2000C_Exists **String** Indicates whether the HI01-4 value in the 2000C loop exists.

Value	Meaning
`true`	The HI01-4 value exists.
`false`	The HI01-4 value does *not* exist.

HI01_4_2000D_Exists **String** Indicates whether the HI01-4 value in the 2000D loop exists.

Value	Meaning
`true`	The HI01-4 value exists.
`false`	The HI01-4 value does *not* exist.

TRN_2000F_Exists **String** Indicates whether the TRN segment in the 2000F loop exists.

Value	Meaning
`true`	The TRN segment exists.
`false`	The TRN segment does *not* exist.

Input Variables

DTP_439_2000C_Exists **String** Indicates whether the DTP (DTP01=439) segment in the 2000C loop exists.

Value	Meaning
`true`	The DTP (DTP01=439) segment exists.
`false`	The DTP (DTP01=439) segment does *not* exist.

DTP_439_2000D_Exists **String** Indicates whether the DTP (DTP01=439) segment in the 2000D loop exists.

Value	Meaning
`true`	The DTP (DTP01=439) segment exists.
`false`	The DTP (DTP01=439) segment does *not* exist.

HI_2000C_Exists **String** Indicates whether the HI segment in the 2000C loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

HI_2000D_Exists **String** Indicates whether the HI segment in the 2000D loop exists.

Value	Meaning
`true`	The HI segment exists.
`false`	The HI segment does *not* exist.

HI_2000C_BK_Exists **String** Indicates whether the HI (HI01 = BK) segment in the 2000C loop exists.

Value	Meaning
`true`	The HI (HI01 = BK) exists.
`false`	The HI (HI01 = BK) does *not* exist.

HI_2000D_BK_Exists **String** Indicates whether the HI (HI01 = BK) segment in the 2000D loop exists.

Value	Meaning
`true`	The HI (HI01 = BK) exists.
`false`	The HI (HI01 = BK) does *not* exist.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.
TRN_2000F_Exists	**String** Indicates whether the TRN segment in the 2000F loop exists.

Value	Meaning
`true`	The TRN segment exists.
`false`	The TRN segment does *not* exist.

Usage Notes

You can use this service to validate the Service information in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HIReq.syntaxValidation:validateX094A1Header

Use this service to find syntax errors in the Header of the 278 Request transaction set. The Header contains segments ST and BHT.

Input Variables

header278Req	**Document** Header segment data to validate.
delimiters	**Document** It contains the segment, field and subfield delimiters from the incoming document.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.syntaxValidation:validateX094A1UMO

Use this service to find syntax errors in the UMO (Source) part of the 278 Request transaction set.

Input Variables

SourceData	**String** Source (UMO) HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

UMO_HL	**Document** Values in the segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the UMO (Source) part in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.syntaxValidation:validateX094A1Requester

Use this service to find syntax errors in the Requester part of the 278 Request transaction set.

Input Variables

receiverData	**String** Requester HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Requester_HL	**Document** Values in the segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Requester part in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.syntaxValidation:validateX094A1Subscriber

Use this service to find syntax errors in the Subscriber part of the 278 Request transaction set.

Input Variables

subscriberData	**String** Subscriber HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

subscriber_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber part in 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.syntaxValidation:validateX094A1Dependant

Use this service to find syntax errors in the Dependent part of the 278 Request transaction set.

Input Variables

dependentData	**String** Dependent HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Dependent_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent part in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.syntaxValidation:validateX094A1Provider

Use this service to find syntax errors in the Provider part of the 278 Request transaction set.

Input Variables

providerData	**String** Provider HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variables

delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Provider_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider part in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlReq.syntaxValidation:validateX094A1Service

Use this service to find syntax errors in the Service part of the 278 Request transaction set.

Input Variables

serviceData	**String** Service HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Service_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Serivce part in a 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIReq.syntaxValidation:HIdatecheck

Use this service to find syntax errors in the HI segment in the 278 Request transaction set.

Input Variables

HI	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the HI segment in the 278 Request transaction set. The WmHipaaSample package demonstrates how this service is used when validating 278 Request transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes

This folder includes services related to the HIPAA 278 Health Care Services Review Response transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HIRes.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HIRes.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HIRes.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HIRes.codeSourceValidation:validateX094A1Subscriber

Use this service to find code source errors in the Subscriber loop of the 278 Response transaction set.

Input Variables

Subscriber_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber loop in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.codeSourceValidation:validateX094A1Dependent

Use this service to find code source errors in the Dependent loop of the 278 Response transaction set.

Input Variables

Dependent_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent loop in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlRes.codeSourceValidation:validateX094A1Provider

Use this service to find code source errors in the Provider loop of the 278 Response transaction set.

Input Variables

Provider_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider loop in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlRes.codeSourceValidation:validateX094A1Service

Use this service to find code source errors in the Service loop of the 278 Response transaction set.

Input Variables

Service_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Service loop in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HIRes.semanticValidation:validateX094A1UMO

Use this service to find semantic errors in the UMO (Source) part of the 278 Response transaction set.

Input Variables

UMO	**Document** List of segments to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the UMO (Source) information in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.semanticValidation:validateX094A1Requester

Use this service to find semantic errors in the Requester part of the 278 Response transaction set.

Input Variables

Requester_HL	**Document** List of segments to validate.

Input Variables

hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Requester information in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlRes.semanticValidation:validateX094A1Subscriber

Use this service to find semantic errors in the Subscriber part of the 278 Response transaction set.

Input Variables

Subscriber_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.
NM1_2010CB_Exists	**String** Indicates whether the NM1 segment in the 2010CB loop exists.

Value	Meaning
`true`	The NM1 segment exists.
`false`	The NM1 segment does *not* exist.

Output Variables

PWK_2000C_Exists **String** Indicates whether the PWK segment in the 2000C loop exists.

Value	Meaning
`true`	The PWK segment exists.
`false`	The PWK segment does *not* exist.

Usage Notes

You can use this service to validate the Subscriber information in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.semanticValidation:validateX094A1Dependent

Use this service to find semantic errors in the Dependent part of the 278 Response transaction set.

Input Variables

Dependent_HL **Document** List of segments to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

NM1_2010CB_Exists **String** Indicates whether the NM1 segment in the 2010CB loop exists.

Value	Meaning
`true`	The NM1 segment exists.
`false`	The NM1 segment does *not* exist.

PWK_2000C_Exists **String** Indicates whether the PWK segment in the 2000C loop exists.

Value	Meaning
`true`	The PWK segment exists.
`false`	The PWK segment does *not* exist.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent information in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlRes.semanticValidation:validateX094A1Provider

Use this service to find semantic errors in the Provider part of the 278 Response transaction set.

Input Variables

Provider_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider information in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HlRes.semanticValidation:validateX094A1Service

Use this service to find semantic errors in the Service part of the 278 Response transaction set.

Input Variables

Service_HL	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Service information in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1Header

Use this service to find syntax errors in the Header of the 278 Response transaction set. The Header contains segments ST and BHT.

Input Variables

Header	**Document** Header segment data to validate.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1UMO

Use this service to find syntax errors in the UMO (Source) part of the 278 Response transaction set.

Input Variables

Umo_Data	**Object** UMO (Source) HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

UMO	**Document** Values in the segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the UMO (Source) part in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1Requester

Use this service to find syntax errors in the Requester part of the 278 Response transaction set.

Input Variables

receiverData	**Object** Requester HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Requester_HL	**Document** Values in the segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Requester part in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1Subscriber

Use this service to find syntax errors in the Subscriber part of the 278 Response transaction set.

Input Variables

subscriberData	**Object** Subscriber HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Subscriber_HL	**Document** Holds the segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber part in a278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1Dependent

Use this service to find syntax errors in the Dependent part of the 278 Response transaction set.

Input Variables

dependentData	**Object** Dependent HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variables

delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Dependent_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent part in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1Provider

Use this service to find syntax errors in the Provider part of the 278 Response transaction set.

Input Variables

providerData	**Object** Provider HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Provider_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider part in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:validateX094A1Service

Use this service to find syntax errors in the Service part of the 278 Response transaction set.

Input Variables

serviceData	**Object** Service HL segment data of the document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

Service_HL	**Document** Segments that are available in that loop.
syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Service part in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HIRes.syntaxValidation:HIdatecheck

Use this service to find syntax errors in the HI segment in the 278 Response transaction set.

Input Variables

HI	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the HI segment in a 278 Response transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 278 Response transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN

This folder includes services related to the HIPAA 277 Health Care Claim Status Response transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HN.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HN.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HN.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HN.codeSourceValidation:validateTRN

Use this service to find code source errors in the TRN segment list of the 277 transaction set.

Input Variables

Claim_TRN	**Document** List of segments to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the TRN segment list in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.codeSourceValidation:validateSVC

Use this service to find code source errors in the SVC segment list of the 277 transaction set.

Input Variables

Claim_SVC	**Document** List of segments to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SVC segment list in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HN.semanticValidation:validateSubscriber

Use this service to find semantic errors in the Subscriber part of the 277 transaction set.

Input Variables

Subscriber_HL **Document** List of segments to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

dependentExists **String** Indicates whether the Subscriber is the patient.

Value	Meaning
`0`	The Subscriber is the patient.
`1`	The Subscriber is *not* the patient.

addendum **String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber information in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.semanticValidation:validateTRN

Use this service to find semantic errors in the TRN segment list of the 277 transaction set.

Input Variables

Claim_Trace **Document** Segment to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

dependentExists **String** Indicates whether the Subscriber is the patient.

Value	Meaning
`0`	The Subscriber is the patient.
`1`	The Subscriber is *not* the patient.

semanticErrors **Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors **Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the TRN segment list in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 277 transaction set. The Header contains segments ST and BHT.

Input Variables

Header **Document** Header segment data to validate.

delimiters **Document** Segment, field, and subfield delimiters from the incoming document.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateProvider

Use this service to find syntax errors in the Provider part of the 277 transaction set.

Input Variables

Provider_Data	**Object** Provider HL segment data of the document.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider part in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateReceiver

Use this service to find syntax errors in the Receiver part of the 277 transaction set.

Input Variables

Receiver_Data	**Object** Receiver HL segment data of the document.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Receiver part in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateSource

Use this service to find syntax errors in the Source part of the 277 transaction set.

Input Variables

Source_Data	**Object** Source HL segment data of the document.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Source part in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber part of the 277 transaction set.

Input Variables

Subscriber_HL	**Document** List of segments to validate.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber part in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateDependent

Use this service to find syntax errors in the Dependent part of the 277 transaction set.

Input Variables

Dependent_HL	**Document** List of segments to validate.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
addendum	**String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent part in a 277 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateTRN

Use this service to find syntax errors in the Claim Submitter Trace information segments of the 277 transaction set.

Input Variables

Claim_Trace **Document** List of segments to validate.

delimiters **Document** Segment, field, and subfield delimiters from the incoming document.

addendum **String** Indicates whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

dependentExists **String** Indicates whether the Subscriber is the patient.

Value	Meaning
`0`	The Subscriber is the patient.
`1`	The Subscriber is *not* the patient.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Claim Submitter Trace information segments in a 277 transaction set. The WmHipaaTransactions package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HN.syntaxValidation:validateSVC

Use this service to find syntax errors in the SVC segment list of the 277 transaction set.

Input Variables

Claim_SVC **Document** List of segments to validate.

delimiters **Document** Segment, field, and subfield delimiters from the incoming document.

Input Variables

hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
dependentExists	**String** Indicates whether the Subscriber is the patient.

Value	Meaning
`0`	The Subscriber is the patient.
`1`	The Subscriber is *not* the patient.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SVC segments in a 277 transaction set. The WmHipaaTransactions package demonstrates how this service is used when validating a 277 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP

This folder includes services related to the HIPAA 835 Claim Payment/Advice transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HP.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HP.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HP.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HP.codeSourceValidation:validateHeader

Use this service to find code source errors in the Header of the 835 transaction set. The Header contains segments ST, BPR, TRN, CUR, REF list, DTM and N1 list.

Input Variables

Header835	**Document** Header segment data to validate.

Output Variables

codeSourceErrors	**Document List** Code source errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.codeSourceValidation:validateCLP

Use this service to find code source errors in the CLP segment of the 835 transaction set.

Input Variables

DetailCLP	**Document** Segment to validate.
clpStartSegmentCnt	**String** Position of the first CLP segment in the CLP loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the CLP segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.codeSourceValidation:validateSVC

Use this service to find code source errors in the SVC segment of the 835 transaction set.

Input Variables

DetailSVC	**Document** Segment to validate.
clpStartSegmentCnt	**String** Position of the first CLP segment in the CLP loop.

Input Variables

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

codeSourceErrors **Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors **Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SVC segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HP.semanticValidation:validateHeader

Use this service to find semantic errors in the Header of the 835 transaction set. The Header contains segments ST, BPR, TRN, CUR, REF list, DTM, and N1 list.

Input Variables

Header835 **Document** Header segment data to validate.

Output Variables

semanticErrors **Document List** Semantic errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.semanticValidation:validateCLP

Use this service to find semantic errors in the CLP segment of the 835 transaction set.

Input Variables

DetailCLP	**Document** Segment to validate.
clpStartSegmentCnt	**String** Position of the first CLP segment in the CLP loop.
payeeIDCode	**String** Payee Identification Code for the transaction set being validated.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the CLP segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.semanticValidation:validateSVC

Use this service to find semantic errors in the SVC segment of the 835 transaction set.

Input Variables

DetailSVC	**Document** Segment to validate.
clpStartSegmentCnt	**String** Position of the first CLP segment in the CLP loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SVC segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 835 transaction set. The Header contains segments ST, BPR, TRN, CUR, REF list, DTM, and N1 list.

Input Variables

Header835	**Document** Header segment data to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

payeeIDCode	**String** Payee Identification Code for the transaction set being validated.
BPR02	**String** Value of the BPR02 element.
syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validateLXHeader

Use this service to find syntax errors in the LX Header of the 835 transaction set. The LX Header contains segments LX, TS2, and TS3.

Input Variables

DetailLXHeader — **Document** Header to validate.

delimiters — **Document** Record, field, and subfield delimiters for the transaction set being validated.

addendum — **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors — **Document List** Existing syntax errors for the transaction set being validated.

Input Variables

balancing **String** Whether the segment is to be validate for Balancing errors.

Value	Meaning
`true`	The segment will be validated for Balancing errors.
`false`	The segment will *not* be validated for Balancing errors.

Output Variables

syntaxErrors **Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the LX Header in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validateCLPHeader

Input Variables

DetailCLPHeader **Document** CLP loop to validate.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

clpSyntaxErrors **Document List** Syntax errors for the CLP loop being validated.

Usage Notes

You can use this service to validate the CLP loop in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validateCLP

Use this service to find syntax errors in the CLP segment of the 835 transaction set.

Input Variables

DetailCLP **Document** Segment to validate.

clpStartSegmentCnt **String** Position of the first CLP segment in the CLP loop.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

balancingErrors **Document List** Existing Balancing errors for the transaction set being validated.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

balancing **String** Whether the segment is to be validated for Balancing errors.

Value	Meaning
`true`	The segment will be validated for Balancing errors.
`false`	The segment will *not* be validated for Balancing errors.

Output Variables

CLP03Data **String** Value of the CLP03 element. This is needed for balancing.

CLP04Data **String** Value of the CLP04 element. This is needed for balancing.

balancingErrors **Document List** Balancing errors for the segment being validated. If the input variable *balancingErrors* is not null, the segment Balancing errors are appended to the transaction set errors.

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the CLP segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validatePLB

Use this service to find syntax errors in the PLB segment of the 835 transaction set.

Input Variables

Summary	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

providerLevel Adjustment	**String** Provider-level adjustment amount needed for transaction level balancing.
syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the PLB segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:transactionBalancing

Use this service to find transaction balancing errors in the 835 transaction set.

Input Variables

sumProviderLevel Adjustment	**String** Sum of all provider-level adjustment amounts.
BPR02	**String** Value of the BPR02 element.
sumCLP04forLXList	**String** Sum of all CLP04 elements in the transaction set being validated.

Output Variables

txnBalancingError	**Document** Transaction Balancing errors for the segment being validated.

Usage Notes

You can use this service to validate the transaction balancing rule for the 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:claimBalancing

Use this service to find balancing errors in the claim loop of the 835 transaction set.

Input Variables

DetailCLP	**Document** Claim loop data to validate.
totSVCAdjustmentAmt	**String** Value of the SVC adjustment amount.
clpStartSegmentCnt	**String** Position of the C:P segment in the CLP loop.

Output Variables

clmBalancingError	**Document** Balancing errors for the CLP loop being validated.
CLP03Data	**Document** Value of the CLP03 element.
CLP04Data	**Document** Value of the CLP04 element.

Usage Notes

You can use this service to validate the CLP loop in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:getProviderAdjustmentAmount

Use this service to find transaction balancing errors in the 835 transaction set.

Input Variables

Summary	**Document** PLB loop data to validate.

Output Variables

providerLevelAdjustment	**String** Value of the Provider adjustment amount.

Usage Notes

You can use this service to validate the PLB loop in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:serviceLineBalancing

Use this service to find balancing errors in the claim loop of the 835 transaction set.

Input Variables

DetailSVC	**Document** SVC loop data to validate.
clpStartSegmentCnt	**String** Position of the CLP segment in the CLP loop.

Output Variables

svcBalancingError	**Document** Balancing errors for the SVC loop being validated.
svcAdjustment Amount	**String** Value of the SVC adjustment amount.

Usage Notes

You can use this service to validate the SVC loop in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validateBPR

Use this service to find balancing errors in the BPR segment of the 835 transaction set.

Input Variables

BPRDT	**Document** BPR segment data to validate.

Output Variables

bprErrors	**Document** Balancing errors for the BPR segment being validated.
BPR02	**Document** Value of the BPR02 element.

Usage Notes

You can use this service to validate the BPR segment in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HP.syntaxValidation:validateSVC

Use this service to find syntax errors in the SVC loop of the 835 transaction set.

Input Variables

DetailsSVC **Document** SVC loop data to validate.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

balancing **String** Whether the segment is to be validated for Balancing errors.

Value	Meaning
`true`	The segment will be validated for Balancing errors.
`false`	The segment will *not* be validated for Balancing errors.

clpStartSegmentCnt **String** Position of the CLP segment in the CLP loop.

Output Variables

svcSyntaxErrors **Document List** Syntax errors for the SVC loop being validated.

svcAdjustment Amount **String** Value of the SVC adjustment amount.

svcBalancingError **Document** Balancing errors for the SVC loop being validated.

Usage Notes

You can use this service to validate the SVC loop in an 835 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 835 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR

This folder includes services related to the HIPAA 276 Health Care Claim Status Request transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HR.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HR.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HR.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HR.codeSourceValidation:validateSVC

Use this service to find code source errors in the SVC segment list of the 276 transaction set.

Input Variables

Claim_SVC	**Document** List of segments to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segments being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SVC segment list in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HR.semanticValidation:validateSubscriber

Use this service to find semantic errors in the Subscriber part of the 276 transaction set.

Input Variables

Subscriber_HL	**Document** List of segments to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Output Variables

semanticErrors	**Document List** Semantic errors for the segments being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber information in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.semanticValidation:validateTRN

Use this service to find semantic errors in the TRN segment list of the 276 transaction set.

Input Variables

Claim_Trace	**Document** Segment to validate.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
dependentExists	**String** Indicates whether the Subscriber is the patient.

Value	Meaning
`0`	The Subscriber is the patient.
`1`	The Subscriber is *not* the patient.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the TRN segment list in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 276 transaction set. The Header contains segments ST and BHT.

Input Variables

Header	**Document** Header segment data to validate.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateProvider

Use this service to find syntax errors in the Provider part of the 276 transaction set.

Input Variable

Provider_Data	**Object** Provider HL segment data of the document.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variable

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segments being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Provider part in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateReceiver

Use this service to find syntax errors in the Receiver part of the 276 transaction set.

Input Variables

Receiver_Data	**Object** Receiver HL segment data of the document.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Receiver part in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateSource

Use this service to find syntax errors in the Source part of the 276 transaction set.

Input Variables

Source_Data	**Object** Source HL segment data of the document.
delimiters	**Document** Segment, field, and subfield delimiters from the incoming document.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Source part in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber part of the 276 transaction set.

Input Variables

Subscriber_HL	**Document** List of segments to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variables

addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber part in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating z 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateDependent

Use this service to find syntax errors in the Dependent part of the 276 transaction set.

Input Variables

Dependent_HL	**Document** List of segments to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variables

addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segments being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the segment unrecognized and unexpected segments errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent part in a 276 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateTRN

Use this service to find syntax errors in the Claim Submitter Trace information segments of the 276 transaction set.

Input Variables

Claim_Trace	**Document** Segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variables

dependentExists **String** Indicates whether the Subscriber is the patient.

Value	Meaning
`0`	The Subscriber is the patient.
`1`	The Subscriber is *not* the patient.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
`true`	The addendum version will be validated.
`false`	The addendum version will *not* be validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Claim Submitter Trace information segments in a 276 transaction set. The WmHipaaTransactions package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HR.syntaxValidation:validateSVC

Use this service to find syntax errors in the SVC segment list of the 276 transaction set.

Input Variables

Trace_SVC **Document** List of segments to validate.

delimiters **Document** Segment, field, and subfield delimiters from the incoming document.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the SVC segments in a 276 transaction set. The WmHipaaTransactions package demonstrates how this service is used when validating a 276 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS

This folder includes services related to the HIPAA 270 Health Care Eligibility Benefit Inquiry transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.HS.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.HS.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HS.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.HS.codeSourceValidation:validateReceiver

Use this service to find code source errors in the receiver of the 270 transaction set.

Input Variables

Receiver_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the receiver HL segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.codeSourceValidation:validateSubscriber

Use this service to find code source errors in the subscriber of the 270 transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the receiver HL segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.codeSourceValidation:validateEQ

Use this service to find code source errors in the EQ loop of the 270 transaction set.

Input Variables

EQ	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
true	The addendum version will be validated.
false	The addendum version will *not* be validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the EQ loop segments in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.codeSourceValidation:validateDependent

Use this service to find code source errors in the dependent of the 270 transaction set.

Input Variables

Dependent_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the dependent HL segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.HS.semanticValidation:validateReceiver

Use this service to find semantic errors in the receiver HL segment of the 270 transaction set.

Input Variables

Receiver_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the receiver HL segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.semanticValidation:validateEQ

Use this service to find semantic errors in the EQ loop segment of the 270 transaction set.

Input Variables

EQ	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
semanticErrors	**Document List** Existing semantic errors for the transaction set being validated.

Output Variables

semanticErrors	**Document List** Semantic errors for the segment being validated. If the input variable *semanticErrors* is not null, the segment semantic errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the EQ loop segments in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 270 transaction set. The Header contains segments ST, BHT.

Input Variables

Header270	**Document** Header segment data to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation:validateSource

Use this service to find syntax errors in the source HL segment of the 270 transaction set.

Input Variables

sourceData	**String** holds the HL Segment data of the source
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segment being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the unrecognized and unexpected segment errors are appended to the transaction set errors.
source_HL	**Document** Segment as output for next service.

Usage Notes

You can use this service to validate the source HL segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation:validateReceiver

Use this service to find syntax errors in the Receiver segments of the 270 transaction set.

Input Variables

receiverData	**String** HL Segment data of the Receiver.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.

Input Variables

delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
addendum	**String** Whether the addendum version is to be validated.

Value	Meaning
true	The addendum version will be validated.
false	The addendum version will *not* be validated.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.
unrecog_and_ unexpected_segments	**Document List** Existing unrecognized and unexpected errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.
unrecog_and_ unexpected_segments	**Document List** Unrecognized and unexpected errors for the segment being validated. If the input variable *unrecog_and_unexpected_segments* is not null, the unrecognized and unexpected segment errors are appended to the transaction set errors.
Receiver_HL	**Document** Segment as output for next service.

Usage Notes

You can use this service to validate the receiver segments in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation:validateSubscriber

Use this service to find syntax errors in the Subscriber HL segment of the 270 transaction set.

Input Variables

Subscriber_HL	**Document** Segment to validate.
hlStartSegmentCnt	**String** Position of the first HL segment in the HL loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Input Variables

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
true	The addendum version will be validated.
false	The addendum version will *not* be validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Subscriber HL segment in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation:validateEQ

Use this service to find syntax errors in the EQ loop segment of the 270 transaction set.

Input Variables

EQ **Document** Segment to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

addendum **String** Whether the addendum version is to be validated.

Value	Meaning
true	The addendum version will be validated.
false	The addendum version will *not* be validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

Input Variables

dependentExists **String** Indicates whether the dependent in the 2000D loop exists.

Value	Meaning
true	The dependent exists.
false	The dependent does not exist.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the EQ loop segments in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.HS.syntaxValidation:validateDependent

Use this service to find syntax errors in the Dependent loop segment of the 270 transaction set.

Input Variables

Dependent_HL **Document** Segment to validate.

hlStartSegmentCnt **String** Position of the first HL segment in the HL loop.

delimiters **Document** Record, field, and subfield delimiters for the transaction set being validated.

syntaxErrors **Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors **Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the Dependent loop segments in a 270 transaction set. The WmHipaaSample package demonstrates how this service is used when validating a 270 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA

This folder includes services related to the HIPAA 820 Payment Order/Remittance Advice transaction set and contains the following validation sub-folders:

- wm.ip.hipaa.transaction.X12.V4010.RA.codeSourceValidation
- wm.ip.hipaa.transaction.X12.V4010.RA.semanticValidation
- wm.ip.hipaa.transaction.X12.V4010.RA.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.RA.codeSourceValidation

wm.ip.hipaa.transaction.X12.V4010.RA.codeSourceValidation:validateHeader

Use this service to find code source errors in the Header of the 820 transaction set. The Header contains segments ST, BPR, TRN, CUR, REF list, DTM, and N1 list.

Input Variables

Header820	**Document** Header segment to validate.

Output Variables

codeSourceErrors	**Document List** Code source errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA.codeSourceValidation:validateENT

Use this service to find code source errors in the ENT segment of the 820 transaction set.

Input Variables

ENT	**Document** Header segment to validate.
codeSourceErrors	**Document List** Existing code source errors for the transaction set being validated.

Output Variables

codeSourceErrors	**Document List** Code source errors for the segment being validated. If the input variable *codeSourceErrors* is not null, the segment code source errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the ENT segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA.semanticValidation

wm.ip.hipaa.transaction.X12.V4010.RA.semanticValidation:validateHeader

Use this service to find semantic errors in the Header of the 820 transaction set. The Header contains segments ST, BPR, TRN, CUR, REF list, DTM, and N1 list.

Input Variables

Header820	**Document** Header segment to validate.

Output Variables

semanticErrors	**Document List** Semantic errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA.syntaxValidation

wm.ip.hipaa.transaction.X12.V4010.RA.syntaxValidation:validateHeader

Use this service to find syntax errors in the Header of the 820 transaction set. The Header contains segments ST, BPR, TRN, CUR, REF list, DTM, and N1 list.

Input Variables

Header820	**Document** Header segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.

Output Variables

BPR02	**String** Value of the BPR02 element. This element is needed for balancing.
syntaxErrors	**Document List** Syntax errors for the Header being validated.

Usage Notes

You can use this service to validate the Header segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA.syntaxValidation:validateENT

Use this service to find syntax errors in the ENT segment of the 820 transaction set.

Input Variables

ENT	**Document** Header segment to validate.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the ENT segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA.syntaxValidation:validateRemittance

Use this service to find syntax errors in the RMR segment of the 820 transaction set.

Input Variables

Remittance	**Document** Segment to validate.
rmrStartSegmentCnt	**String** Position of the first RMR segment in the RMR loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
remittanceType	**String** Whether the remittance is Individual or Organizational.

Value	Meaning
`Individual`	The remittance is Individual.
`Organizational`	The remittance is Organizational.

syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the RMR segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

wm.ip.hipaa.transaction.X12.V4010.RA.syntaxValidation:validateNM1

Use this service to find syntax errors in the NM1 segment of the 820 transaction set.

Input Variables

NM1DT	**Document** Segment to validate.
rmrStartSegmentCnt	**String** Position of the first RMR segment in the RMR loop.
delimiters	**Document** Record, field, and subfield delimiters for the transaction set being validated.
syntaxErrors	**Document List** Existing syntax errors for the transaction set being validated.

Output Variables

syntaxErrors	**Document List** Syntax errors for the segment being validated. If the input variable *syntaxErrors* is not null, the segment syntax errors are appended to the transaction set errors.

Usage Notes

You can use this service to validate the NM1 segment in an 820 transaction set. The WmHipaaSample package demonstrates how this service is used when validating an 820 transaction set.

webMethods

APPENDIX A

Code Source List

Important! Code sets are constantly changing. The *webMethods6\IntegrationServer*\packages\WmHipaaCodeSource\data\codesets.zip file contains the version of each code set that was available when the HIPAA Module was released. Please check the webMethods Advantage Download site for the latest version of these code sets.

The following table provides a list of all HIPPA Code Sources, whether they are provided with the webMethods HIPAA Module, and, if provided, the file name of the Code Source in the *webMethods6\IntegrationServer*\packages\WmHipaaCodeSource directory.

Important! Some of the code sources provided by webMethods need to be licensed. Please make sure you have a license to use these code sources.

Number	Name	Provided	File Name
4	ABA Routing Number		
5	Countries, Currencies and Funds	✔	5_Country_Currency_Codes.txt
16	D-U-N-S Number		
22	States and Outlying Areas of the U.S	✔	22_States_and_Outlying_Areas.txt
41	Universal Product Code		
43	FIPS-55 (Named Populated Places)	✔	43_Fips_Populated_Places.txt
51	ZIP Code	✔	51_Zip_Codes.txt
60	DFI-Identification Number		
77	X12 Directories	✔	Implemented in Schema

Number	Name	Provided	File Name
91	Canadian Financial Institution branch and Institution Number		
94	International Organization for Standardization (Date and Time)	✔	Implemented in Schema
102	Languages	✔	102_Languages_Codes.txt
121	Health Industry Identification Number		
130	Health Care Financing Administration Common Procedural Coding System	✔	130_HCPCS_Codes.txt
131	International Classification of Diseases Clinical Mod (ICD-9-CM) Procedure	✔	131_ICD_CM_9_Codes.txt
132	National Uniform Billing Committee (NUBC) Codes	✔	132_Nubc_Type_Codes.txt
133	Current Procedural Terminology (CPT) Codes	✔	133_Cpt_Codes.txt
134	National Drug Code	✔	134_National_Drug_Codes.txt
135	American Dental Association Codes	✔	135_Cdt_Codes.txt
139	Claim Adjustment Reason Code	✔	139_Claim_Adjustment_Reason_Codes.txt
229	Diagnosis Related Group Number (DRG)	✔	229_DIAG_Related_Codes.txt
230	Admission Source Code	✔	230_Admission_Source_Codes.txt
231	Admission Type Code	✔	231_Admission_Type_Codes.txt
235	Claim Frequency Type Code	✔	235_Claim_Frequency_Type_Codes.txt
236	Uniform Billing Claim Form Bill Type	✔	236_Uniform_ Billing_Type_Codes.txt
237	Place of Service from Health Care Financing Administration Claim Form	✔	237_Place_of_Service.txt
239	Patient Status Code	✔	239_Patient_Status_Codes.txt
240	National Drug Code by Format	✔	240_National_Drug_Codes.txt
245	National Association of Insurance Commissioners (NAIC) Code	✔	245_NAIC_Codes.txt
307	National Association of Boards of Pharmacy Number		
359	Treatment Codes	✔	359_Treatment_Codes.txt
411	Remittance Remark Codes	✔	411_Remittance_Remarks_Codes.txt
457	NISO Z39.53 Language Code List	✔	457_Language_Codes.txt

Number	Name	Provided	File Name
507	Health Care Claim Status Category Code	✔	507 Claim_Status_Catagory_Codes.txt
508	Health Care Claim Status Code	✔	508_Claim_Status_Codes.txt
513	Home Infusion EDI Coalition (HIEC) Product/Service Code List		
522	Health Industry Labeler Identification Code		
530	National Council for prescription drug programs Reject/Payment Codes		
537	Health Care Financing Administration National Provider Identifier		
540	Health Care Financing Administration National Plan ID		
DOD1	Military Rank and Health Care Service Region		
DOD2	Paygrade		

webMethods

APPENDIX B

webMethods HIPAA Module Sample

Overview

The sample provided with the webMethods HIPAA Module illustrates how to validate a HIPPA message and generate acknowledgements and any resulting error reports. The sample does not demonstrate sending the acknowledgements and reports to the trading partner.

To use the sample, you need to set up the HIPAA Module on your Integration Server.

Setting Up the Sample

Prerequisite

You will need an Integration Server with the HIPAA Module installed. For instructions about how to install the HIPAA Module, see Chapter 2, "Installing the webMethods HIPAA Module" in this guide.

Set Up the My Enterprise Profile

To set up the My Enterprise profile

1 Using the Trading Networks Console, create the My Enterprise profile. For detailed instructions, see the chapter about defining your profile in the *webMethods Trading Networks--Building Your Trading Network*.

When defining the My Enterprise profile, be sure to define the following:

Profile Tab	Field	Value
Corporate	External ID Type	DUNS
	Value	987654321
	External ID Type	Mutually Defined
	Value	PROVIDER

Set Up the Trading Partner Profile

To set up the trading partner profile

1 Using the Trading Networks Console, create a partner profile for the sender trading partner. For detailed instructions, see the chapter about defining partner profiles in the *webMethods Trading Networks--Building Your Trading Network*.

When defining the partner profile, be sure to define the following:

Profile Tab	Field	Value
Corporate	**External ID Type**	DUNS
	Value	123456789
	External ID Type	Mutually Defined
	Value	PAYER

Install the Necessary TN Document Types

You can test processing for any of the supported transaction sets.

To install the necessary TN document types

1 Install the X12 4010 *XXX* TN document type for each transaction set that you want to process, where *XXX* is the transaction set value (for example, X12 4010 835). You can install more than one transaction set TN document type.

2 Install the X12 4010 997 TN document type.

For detailed instructions to install TN document types, see "Step 1: Install TN Document Types for HIPAA Transactions" on page 32.

Note: When you select to install TN document types for the X12 4010 *XXX* and X12 4010 997 transactions, the EDI Module also installs the TN document types for the X12 Envelope and X12 Group. Additionally, the EDI Module installs flat file schemas for the selected transaction and the 997 functional acknowledgement.

The X12 TA1 TN document type is installed automatically when you install the HIPAA Module.

Import the Sample Processing Rules

Note: You can import and use these sample processing rules, or you can use processing rules that you have created.

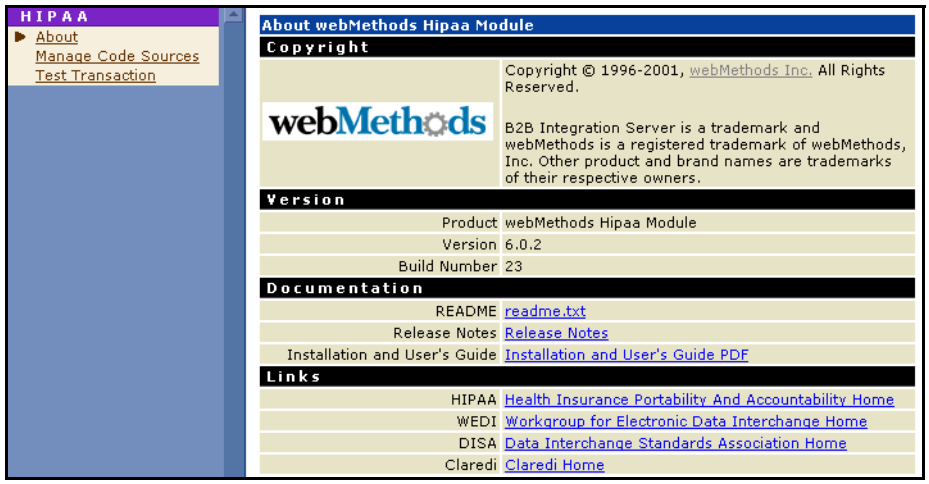
1. Using the Trading Networks Console, Select **File** ▸ **Import**.
2. Click the **Select File** icon.
3. Browse to and select the following file, and then click **Open**.

   *webMethods6**IntegrationServer*\packages\WmHipaaSample\data\processingRules.xml
4. Move the following processing rules from the **Available Items** list to the **Selected Items** list:
   - **Process X12 TA1**
   - **Process X12 997**
   - **Process X12 Envelope**
   - **Process X12 Group**
5. Click **OK**.
6. Move the new processing rule above the Default processing rules.

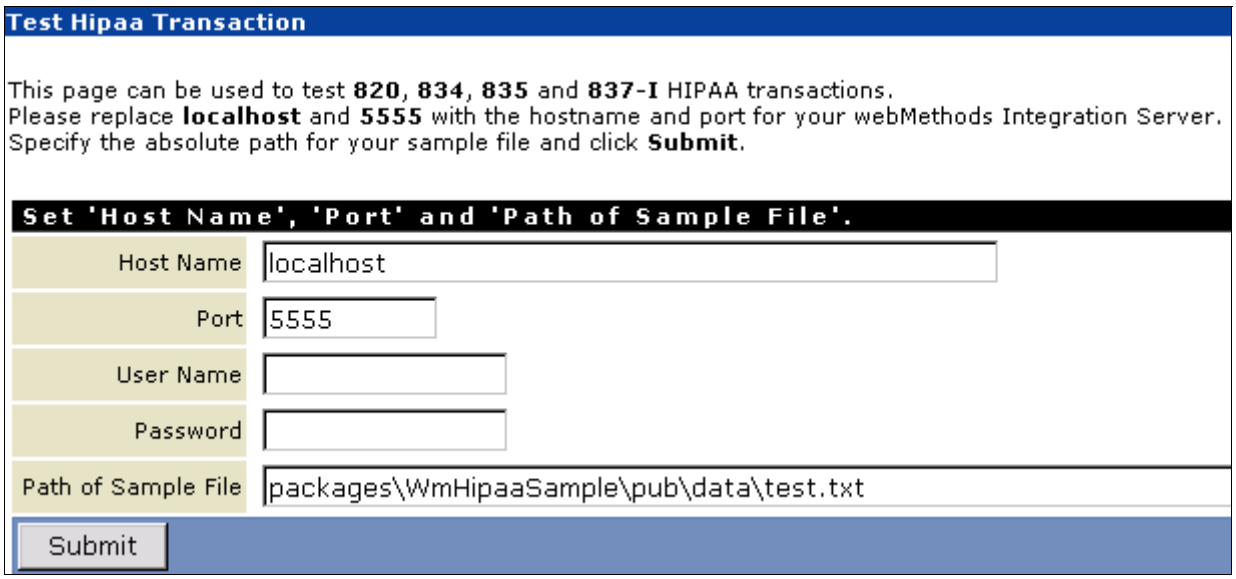
Running the Sample

To run the HIPAA message sample

1 From the Server Administrator, under **Adapters**, click **HIPAA**. The Server Administrator displays the HIPAA Module home page.



2 On the HIPAA Module home page, click **Test Transaction**.



3 In the **Host Name** field, specify the host name to which you want to send the HIPAA message.

4 In the **Port** field, specify the port to which you want to send the HIPAA message.

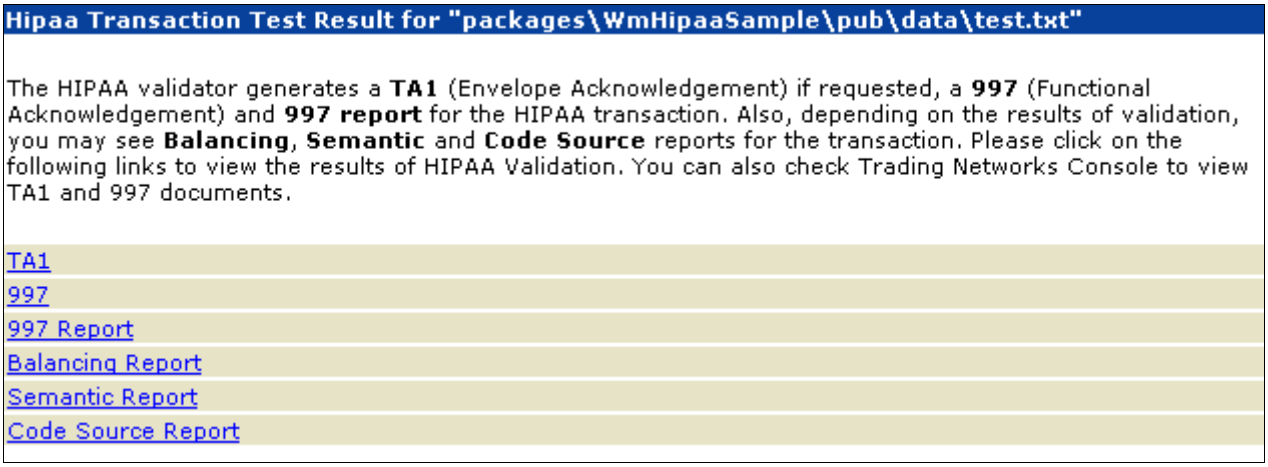
5 In the **User Name** field, specify the user name for the host.

6 In the **Password** field, specify the password for the user name.

7 In the **Path of Sample File** field, specify the file containing the HIPAA message.

Note: A test message is provided in the *webMethods6\IntegrationServer*packages\WmHipaaSample\pub\data\test.txt file.

8 Click **Submit**. Depending on how you have configured your processing rule for this transaction type, any acknowledgements generated for this transaction display. If errors are encountered, error reports display as well.



9 Click the desired link to view the acknowledgment or error report. You also can view the acknowledgements (TA1 and 997) using the Trading Networks Console.

Index

Numerics

A

B

C

D

E

F

G

H

I

L

M

N

O

P

R

S

T

U

V

W