



eGateway Configuration

Document Revision History

Version	Date	Author	Description
1	October 31, 2003	Laurance Stuntz	Initial Documentation
2	January 19, 2004	Laurance Stuntz	Added Batch Process Flow picture

Table of Contents

Purpose	3
eGateway Process Overview	3
Process Descriptions	4
File Arrives	4
eGateway Loads File	4
eGateway finds Transformation	5
eGateway Executes Actions	5
eGateway Action Configuration	6
Map Action	6
Validate Action	7
Log Action	8
Transport Action	8
<i>Direct Transport</i>	10
<i>FTP Transport</i>	11
<i>Batch Transport</i>	12
<i>Bulletin Board Transport</i>	13
<i>Command "Transport"</i>	14
Batch Thread Processing	15
eGateway Batch Process Flow	15
Transport*Batch	16
File Pickup Jobs	16
Exception Processing	17
Alert Listing	17
Alert Conditions	18

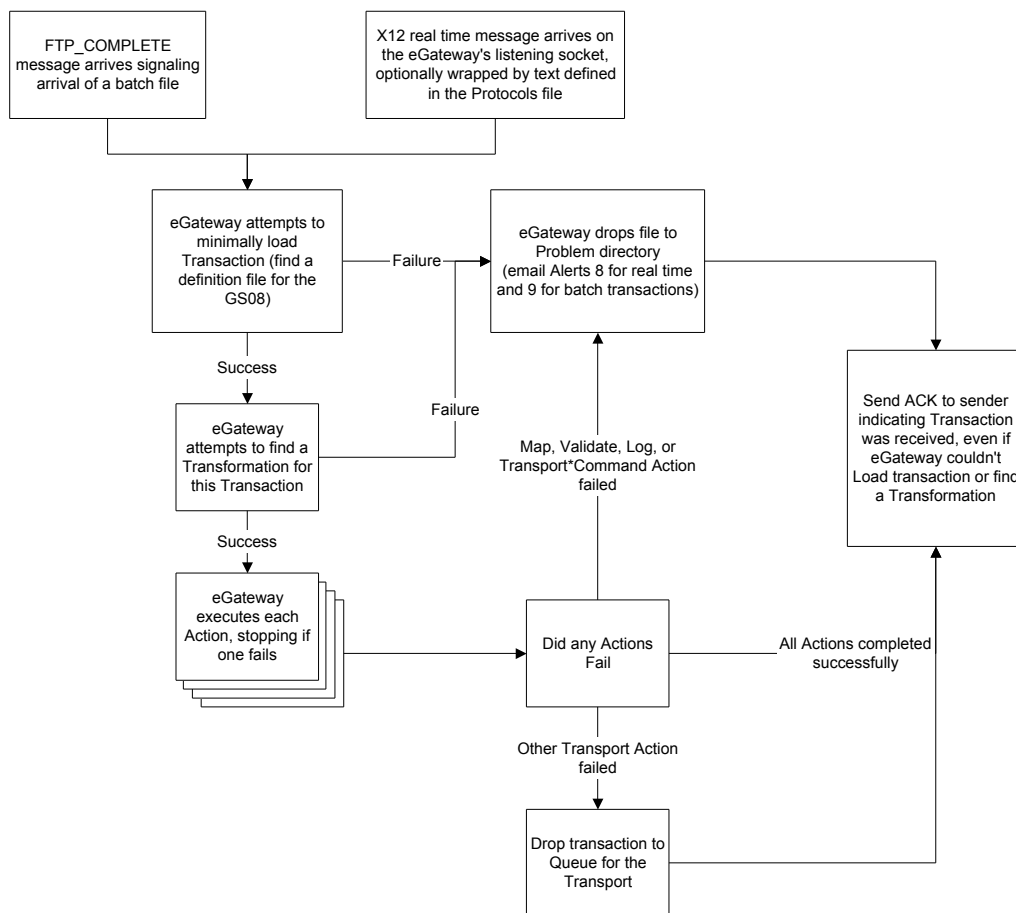
Purpose

This document describes the process that each transaction goes through when it arrives at the eGateway and the complete configuration of the eGateway Mapping, Partners, and Jobs files that allow a site to best manage delivery of X12 transactions.

eGateway Process Overview

The eGateway's Mapping.txt configuration file controls how the eGateway interacts with each transaction that arrives. The flowchart below shows the process each real time transaction goes through.

Process Overview



Process Descriptions

File Arrives

There are three mechanisms for delivering a file to the eGateway:

- ◆ Remote trading partner sends the file to the eGateway's listening socket
- ◆ Remote trading partner FTP's the file to a folder on the eGateway server and then sends an "FTP_COMPLETE" custom message to the eGateway's listening socket notifying the eGateway of the new file.
- ◆ File is transferred to a folder on the eGateway that is scanned by a "Transport Batch" Action. This transfer can happen through an external script, manual file copy, or use of a File Pickup Job. See the Batch Thread processing section for more information on this process.

After discovering that a file is available for processing, the eGateway reads the file into memory and strips any line-level protocol information (see the Transport Direct section for more information about line-level protocols). For files sent to the eGateway, either directly to the listening socket, or through FTP and subsequent notification, responsibility for processing the transaction is transferred to one of the "Worker Threads". The eGateway creates 2 worker threads per processor on the machine.

At this point the eGateway assumes that the transaction is an X12 transaction. In order to further validate the transaction, the eGateway tries to find a Definition for the transaction in the Load section.

eGateway Loads File

When the eGateway Loads a file that has arrived, it goes through several steps:

1. First, it validates the ISA and GS segments. The eGateway currently supports the 3040, 3070, and 4010 versions of these segments.
2. Next, it determines which Definition to load based on the value in the GSo8 and the value in the ST01. If there is no definition that matches these values, then the eGateway cannot process it and the file will be saved to the Problem folder.

After determining that a Definition for this file exists, the eGateway validates every segment in the file if the eGateway Validate registry entry is set to 1.

If the eGateway fails to Load the file, either because the Definition is not found, or the file fails validation, then it saves the file to the Problem directory.

eGateway finds Transformation

After Loading the file, the eGateway looks for a Transformation that matches the X12 transaction in memory. Each line in the Mapping.txt file corresponds to a Transformation, and the key to the transformation is the combination of the following information:

Transaction ID – the value from the ST01

- ◆ Implementation Guide Version ID – the value from the GSo8. This can be a ‘?’ if all transactions of this type for this sender/receiver pair should be dealt with in the same way.
- ◆ Sender ID – the value from either the ISA06 or the GSo2. The eGateway supports routing transactions at either the Interchange or the Application level. Typically, providers with multiple systems producing the same type of transaction will use Application level routing because when the responses come back, they can then be sent to the appropriate core system.
- ◆ Receiver ID – the value from either the ISA08 or GSo3
- ◆ Production/Test Flag – this can be a ‘?’ if both Test and Production transactions should be dealt with in the same way.

If the eGateway doesn't find a Transformation for the transaction, it logs an error message and saves the transaction to the Problem directory.

Note: *Because the eGateway does not currently have the ability to natively reformat and split apart or combine X12 transactions, it only supports a single functional group (GS to GE) within an interchange and it only supports a single interchange within a file. Entities that need to be able to transform large, complicated X12 interchanges that contain multiple transaction types or multiple functional groups should use a commercial translator for this task.*

eGateway Executes Actions

A Transformation is made up of a series of Actions that are run by the eGateway against the transaction. The order of the actions is from left to right within the Transformation line. Each Action is separated by a ‘,’ and the first word in the Action is the Action type, e.g. Map, Validate, Log, or Transport. Within an Action, the separator for parameters is ‘*’.

eGateway Action Configuration

There are currently four types of actions supported by the eGateway Transformation:

1. Map Actions – this class of action runs a “Map” that transforms an X12 file to meet the needs of the receiver.
2. Validate Action – this action matches the received transaction against the implementation guide for this transaction and creates a 997 Functional Acknowledgement
3. Log Action – this action extracts information from the transaction and saves it to a database
4. Transport Action – this type of action handles the guaranteed delivery of the transaction to a remote trading partner

Map Action

A map action takes the following form:

- ◆ MapTool
- ◆ MapName

This was originally designed to use third party mapping tools, but in practice all NEHEN members are only using homegrown NEHEN maps in the eGateway, so the syntax looks like this:

Map*[executable name, with full path]*NotALot

The default location of NEHEN maps is in the [NEHEN install]\eGateway\Maps folder.

When a Map runs, the eGateway saves a copy of the transaction to the eGateway\Temp folder with the name [filename].001 and the mapped transaction has the name [filename].002. After mapping the transaction the eGateway attempts to fully Load and validate the transaction to ensure that the resulting transaction is fully compliant with the appropriate standard. After successfully loading the mapped transaction into memory, the eGateway removes the temporary files and passes the mapped transaction to the next Action in the Transformation.

Sample Map Action:

Map*e:\Program Files\NEHEN\eGateway\Maps\Map4010to4010A1.exe*NotALot

English Description of this Map Action

For this transaction run the Map4010to4010A1.exe map tool, which is found in e:\Program Files\NEHEN\eGateway\Maps. The “NotALot” refers to the map name, but since Map4010to4010A1 just supports the single map, the NotALot in this case is just filler to meet the syntax requirements of the Map Action.

Validate Action

The validate action is designed to create a Functional Acknowledgement (997) and store it to a defined location. The eGateway can then be configured to separately transport the 997 to the appropriate entity.

There are three parameters for the Validate Action:

- ◆ SuccessAction – defines whether to create a 997 if the transaction passes validation, or only to create the 997 if the transaction fails validation. Valid values for this parameter are ‘ACK’ and ‘NoACK’
- ◆ FailureAction – defines whether to stop processing if the transaction fails validation, or to continue with warnings. Valid values for this parameter are ‘Error’ and ‘Warning’
- ◆ 997Destination – the directory where the 997 should be stored. In order to send the 997 back to the sender of the original transaction, a Transport Batch Action must be set up to handle the delivery of the 997. See the Samples section for an example of this.

Sample Validate Actions:

Validate*ACK*Error*e:\Program Files\eGateway\Batch\Partners\Outbound

English Explanation of this Validate Action:

Validate this transaction and produce a 997 for all transactions seen, whether or not the transaction passes HIPAA Validation Types 1 and 2. If the transaction fails validation, then stop processing, and don't continue with any other Actions in the Transformation. Save the 997 to the e:\Program Files\eGateway\Batch\Partners\Outbound directory.

Validate*NoACK*Warning*e:\Program Files\eGateway\Batch\MGH\Outbound

English Explanation of this Validate Action:

Validate this transaction every time, but if it passes HIPAA Validation Types 1 and 2 then don't bother creating a 997. If the transaction fails validation, then create a 997 with a status code of “Accepted, but Errors were noted”. Continue

processing any other Actions in the Transformation. Save the 997 to the e:\Program Files\eGateway\Batch\MGHs\Outbound directory.

Log Action

The Log Action saves information about the transaction to a database. In order to use the Log Action, you must have the following values defined in the Registry for the eGateway:

- ◆ DataSourceName
- ◆ DataSourceLogin
- ◆ DataSourcePassword

You can find more information about setting up these entries in the File Tracking Release Note.

As of the HIPAA Release of the NEHEN core software, the eGateway can only log 837s and the corresponding 997 functional acknowledgements.

There are two parameters for the Log Action:

- ◆ LogType – this should be “Claim” or “997”
- ◆ LogDetails – this value should be “Y” or “N”. This indicates to the logging system whether detailed information should be culled from the transaction and saved to the database. For Claims, this information includes total charges and the number of lines on the claim.

Note: *Logging transaction details significantly slows down processing of a transaction because of the increased number of database updates. For example, logging a typical professional 837 with 1,000 claims would involve 1,000+ database updates when logging details as opposed to a single update when not logging details.*

Sample Log Action:

Log*Claim*N

English explanation:

When running this transformation, save summary information about this file of claims to the database, but don't save detailed information about each claim.

Transport Action

Transport Actions are at the core of the NEHEN eGateway, and they handle delivery of the transaction from the eGateway server to a remote trading partner. A Transport

represents a connection to a remote trading partner and the different types of Transports represent different generic ways of connecting.

- ◆ Direct – This is a raw TCP/IP socket-based method for connecting. The only protocol wrapped around the data is the NEHEN-defined one. This is the preferred Transport for real time transactions
- ◆ FTP – This uses the standard file transport protocol for delivering files. This is typically used for delivering large files and for sending files to systems that don't support a socket-based interface.
- ◆ Batch – This Transport combines the FTP and Direct Transports. The Batch Transport is used to deliver large files (typically claims and remittances) over FTP and then it sends an "FTP_COMPLETE" message to the receiving eGateway using a Direct Transport.
- ◆ Bulletin Board – This Transport supports dialing a modem and logging into a bulletin board and running a series of predefined steps in order to upload or download files. Typically used for claims and the corresponding response reports or remittance advice transactions with non-NEHEN payers.
- ◆ Command – This Transport runs an executable and passes the transaction to it as a parameter. Typically used to post NEHENLite response transactions back to the NEHENLite database.

All Transport types except for Command also support Queueing of transactions, so if the initial delivery of a transaction fails, the eGateway will save the transaction to the Queue folder for that transport. After Queueing, a separate processor thread picks up the transaction, attempts to establish connectivity to the remote trading partner, and deliver the transaction.

Considerations when setting up a Transport:

- ◆ The RelayNode named in the configuration of the Transport must be defined in Partners.txt. If it is not, then the eGateway will not start.
- ◆ Transports are named according to the RelayNode parameter, and the eGateway creates only a single instance of each Transport. The first definition of the Transport found in the Mapping file is the one used. This is especially important for FTP Transports, which often use a different user ID/password or file name depending on the trading partner. If you wish to configure FTP Transports to deliver files to the same FTP server, but using different user IDs or different static names, you must set up two different trading Partners in Partners.txt, but point them at the same IP address and port.

Direct Transport

The Direct Transport is the basic transport the eGateway uses to support real time, TCP/IP socket-based communication. The following parameters are used to define a Direct Transport:

- ◆ RelayNode – A trading partner defined in Partners.txt. This name must match the second parameter on one of the lines in Partners.txt (the “NEHEN Identifier”)
- ◆ Protocol – In order to increase the reliability of the socket-based interface, the eGateway supports additional “line-level” or “application” protocols on top of TCP/IP. These protocols are defined in [NEHEN Install Directory]\NEHENEDI\Control\Protocols.txt and consist of the following information:
 - ◆ Protocol Name
 - ◆ Preamble – the prefix to send before sending the X12 interchange
 - ◆ Length – the length of the transaction that will be sent (-1 if the length is included in the preamble or the transaction is a standard X12 interchange)
 - ◆ Postscript – the text that will be sent immediately following the segment terminator after the IEA segment
 - ◆ ACK – the text that will be sent back if the receiving node successfully processed the transaction sent
 - ◆ NAK – the text that will be sent back if the receiving node didn’t successfully process the transaction sent
 - ◆ TimeOut – the length of time in seconds that the eGateway should wait for an ACK before declaring failure and Queueing the transaction.
- ◆ KeepAlive – Certain remote trading partners shut down their receiving socket after a period of inactivity and then cannot receive new socket connect requests. In order to alleviate this problem, the eGateway can be configured to periodically test the socket so that it remains open. Typically this is configured as 0.
- ◆ InitialConnect – This parameter tells the eGateway whether to establish a connection at startup. Valid values are 1 (Connect at Startup) and 0 (Don’t Connect at Startup).

Sample Transport Direct Action:

```
Transport*Direct*NEHEN003*NEHEN*0*0
```

English explanation:

Send this transaction over a socket to the IP Address/Port combination specified in the Partners.txt file by the NEHEN003 entry. Use the NEHEN protocol to further manage delivery of the transaction, don't worry about keeping the socket alive, and only connect when there is a transaction to deliver to NEHEN003.

FTP Transport

The FTP Transport is often used to transfer files to back-end core systems that don't support a direct socket interface. The following parameters are used in the FTP Transport:

- ◆ RelayNode – same definition as for all other transports.
- ◆ UserID – the user ID to use when logging in to the remote FTP server
- ◆ Password – the password to use when logging in to the remote FTP server
- ◆ FileName – the file name to use when “putting” the file onto the remote server. In addition to static text, there are several parameters that can be used to make the remote file name dynamic:
 - ◆ \$1 – use a unique system-generated ID. The ID consists of 8 bytes with a hexadecimal representation of the IP address that sent the transaction, 6 bytes representing the date (3 bytes for the years since 1900 and 3 bytes for the day of the year), 9 bytes for the time (HHMMSSmmm), and three bytes for a sequence number.
 - ◆ \$8 - use the first byte and the last 7 bytes of the system generated ID
 - ◆ \$d – use a date/time stamp. If no pattern is specified, the stamp is CCYYMMDD_HHMMSSmmm, but a pattern of MMDD is also supported by using the parameter \$d{MMDD}.
 - ◆ \$s – use a sequence number maintained in the Counter table in the database. This parameter has the syntax \$s{CounterName~CounterMaxValue}.
 - ◆ \$i – use the value from the ISA06 in the transaction being sent
 - ◆ \$g – use the value from the GSO2 in the transaction being sent
 - ◆ \$f – use the file name of the base file being sent. This parameter is used primarily in conjunction with the Batch Transports. For real time transactions that arrived at the eGateway through the listening socket, \$f is equivalent to \$1.
- ◆ Active/Passive Flag – There are two types of FTP, Active and Passive. Typically ftp clients should be set up to use passive mode, but because of network firewall issues

this may not be possible. Try setting up all connections with a P in this parameter and if it doesn't work, try switching to an A in this parameter.

Sample Transport FTP Action:

Transport*FTP*GXS*NEHEN001*xxxxyy*\$i_\$g_\$d.837*P

English explanation:

Send this transaction using ftp to the IP Address/Port combination specified in the Partners.txt file by the GXS entry. Create the connection in Passive mode and log into that ftp server using NEHEN001 as the user ID and xxxxyy as the password. After logging in put this transaction onto that server with a file name = [value from the ISA06]_[value from the GSo6]_[datestamp like 20031028_170834985].837

Batch Transport

The Batch Transport combines an FTP or Bulletin Board Transport and a Direct Transport with some additional parameters. The specific parameters for the Batch Transport are:

- ◆ DeliveryType – valid values are FTP or BulletinBoard
- ◆ LocalDirectory – The directory where files to be transferred using this Transport are found. The path is relative to the BatchRootDirectory registry entry for the eGateway.
- ◆ DeliveryParameters – Following the DeliveryType, the eGateway expects all of the appropriate parameters for an FTP or Bulletin Board Transport
- ◆ NotificationParameters – After the DeliveryParameters, the RelayNode and Protocol portion of the Direct Transport are needed to tell the Batch Transport who to notify of delivery. These parameters can be left blank if the Delivery is to a non-NEHEN node.
- ◆ LogFileDelivery – After successful delivery of the file, this parameter indicates whether to update the database with that fact.

Sample Transport Batch Action:

Transport*Batch*FTP*THP*NEHEN002_FTPSite*NEHEN001*xxxxyy*\$d{MMDD}_\$s{THPClaims~9999}.837*P*NEHEN002*NEHEN*Y

English explanation:

Look for files in the E:\Program Files\NEHEN\eGateway\Batch\THP directory. I know it is this directory because the Registry entry BatchRootDirectory has the value E:\Program Files\NEHEN\eGateway\Batch

and the LocalDirectory defined in this Transport is “THP”. Send this transaction using ftp to the IP Address/Port combination specified in the Partners.txt file by the NEHEN002_FTPSite entry. Create the connection in Passive mode and log into that ftp server using NEHEN001 as the user ID and xxxyyy as the password. Create the remote file name with an MMDD date stamp followed by a 5 digit sequence number defined by the THP Claims sequence in the Counter table, followed by .837. After successful delivery of the file, notify the NEHEN002 partner (defined in Partners.txt) of the delivery. And, as a final step, update the database with the fact that delivery was successful.

Bulletin Board Transport

The Bulletin Board Transport is used to transfer files to proprietary Bulletin Board Systems, such as Medicaid Claims and Medicare Part B Claims. Such systems don't support either direct socket interface or an ftp connection. Hence, the Bulletin Board Transport mechanism has been developed to login using dial-up (modem), navigate through a given bulletin board by replicating steps otherwise performed manually by a user, transfer a file and disconnect.

Configuration and setup of a Bulletin Board Transport is a combination of parameters entered in the Mapping.txt file and instructions (sequence of steps) recorded in a SQL database table.

The following parameters are used in the Bulletin Board Transport:

- ◆ RelayNode – The use of the RelayNode in the Bulletin Board Transport is nominal – it allows differentiating between multiple bulletin boards maintained by the same trading partner. The Partners.txt file will have 127.0.0.1 for the IP Address and 0 for the Port for each Relay Node used to transport data to a bulletin board.
- ◆ BulletinBoardName – name of a Bulletin Board to transfer the files.

Note: *It is important that this Bulletin Board name matches exactly the name used to identify sequence steps in the SQL database. It must also match the name of the HyperAccess configuration file that stores dial-up information. See the Bulletin Board Release Notes for more information on detailed setup of a Bulletin Board instance.*

Sample Transport Bulletin Board Action:

Transport*Batch*BulletinBoard*Outbound\DMA7384\Institutional*DMA7384_Institutional*Medicaid_Claims_Institutional*Y

English explanation:

Send a file from the [Batch Install Directory] \“Outbound\DMA7384\Institutional” folder using batch transport, specifically, bulletin board delivery mechanism to a bulletin board. The name of the BulletinBoard is Medicaid_Claims_Institutional. This name must also be setup through HyperAccess and stored in Medicaid_Claims_Institutional.haw. In addition, a series of rows in the BulletinBoardControl table will have the bb_name column value of Medicaid_Claims_Institutional. These rows make up the sequence of steps that will be run against the bulletin board to upload files. After delivery of the claims, update File Tracking with the fact that this claim has been Sent.

Command “Transport”

Transport Command isn't truly a Transport because it doesn't have the guaranteed delivery component that all other Transports have. Instead, Transport Command simply runs a program and passes the transaction to that program as a parameter. Optionally, Transport Command can take other parameters that are defined in the Mapping.txt and the transaction will be passed to the program as the final parameter.

Sample Transport Command Action #1

Transport*Command*e:\Program Files\NEHEN\NEHENLite\Bin\Receive271.exe

English explanation:

Save the transaction in memory to the eGateway\Temp folder then pass the temporary filename to the program Receive271.exe found in e:\Program Files\NEHEN\NEHENLite\Bin. The temporary file name is the only parameter passed in this case. After the program successfully completes, the eGateway deletes the file from the Temp folder. If the program doesn't successfully complete, the eGateway writes a message to the eGateway.txt log file and does not delete the file from Temp so that it can be analyzed later.

Sample Transport Command Action #2

Transport*Command*e:\Program Files\NEHEN\NEHENLite\Bin\Send270.exe -u jbloke -t

English explanation:

Save a temporary file as in Example 1, and run “Send270.exe -u jbloke -t [temp filename]” instead of “Receive271.exe [temp filename]” as was run in Example 1.

Batch Thread Processing

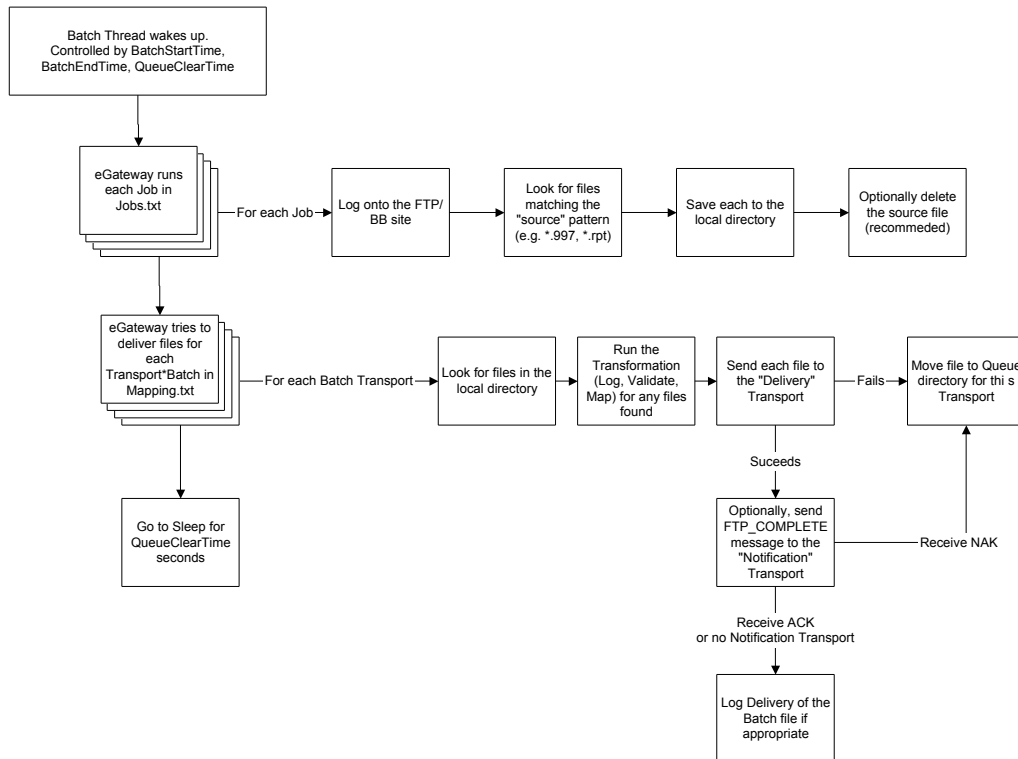
In addition to the Worker Threads that process transactions received either directly over the listening socket or through the FTP_COMPLETE notification, the eGateway also has a Batch Thread that wakes up on a scheduled basis and looks for work to do. There is a single Batch Thread running for each eGateway or eGateway clone.

The Batch Thread runs between the hours specified by the eGateway registry entries BatchStartTime and BatchEndTime. If these entries are not specified on an eGateway, then they default to 7 pm and 12 pm. If the Batch Thread is running, then it runs through the Batch Transports and File Pickup Jobs looking for work to do. After finishing any work, it sleeps for the number of seconds defined by the QueueClearTime entry in the eGateway registry.

eGateway Batch Process Flow

This process flow describes how the eGateway's batch thread sends and picks up files. The "run the Transformation" step is the same as processing actions for the Real Time transactions.

Process Flow



Transport*Batch

When the Batch Thread wakes up, it cycles through each Transport*Batch entry from Mapping.txt looking for files to deliver in the LocalDirectory defined for that Transport. As it finds one, it then Loads the transaction and processes it just as though it arrived through the listening socket or the FTP_COMPLETE notification.

File Pickup Jobs

File Pickup Jobs allow the eGateway to reach out to a remote ftp server and retrieve any files that are waiting. This is primarily used for picking up files that don't have routing information, like non-NEHEN payer submitter reports, or for picking up files from entities that cannot push the file to the eGateway, like internal billing systems or non-NEHEN entities.

There are four main parameters for the File Pickup Job and sub parameters within each. These parameters are:

- ◆ JobName – a unique name to identify the Job. This name is reported by the eGateway Control Panel.
- ◆ Schedule – a set of parameters that define when the Job should look for files, and how long to wait between searches. The times further define the overall Batch Thread start and end times. For example if the Batch Thread is running between 7 pm and 9 am, and a Job is set up to run between 8 am and 10 am, it will actually only run between 8 am and 9 am because the Batch Thread itself will stop processing at 9 am. The parameters consist of:
 - ◆ StartTime – the hour to start running the job
 - ◆ EndTime – the hours to stop running the job
 - ◆ Interval – the number of minutes to wait before running the job again
- ◆ Source – a set of parameters that define the source for the file. This set is made up of:
 - ◆ Type – either FTP or BulletinBoard. If it is FTP, then use these parameters (fully defined in Transport FTP below)
 - ◆ RelayNode, UserID, Password, Active/Passive Flag
 - ◆ SubDirectory – after logging into the site, navigate to this subdirectory
 - ◆ FilePattern – look for files matching this pattern
 - ◆ BulletinBoard parameters

- ◆ RelayNode – IP address and port aren't used
- ◆ BulletinBoardName – The name of the bulletin board to access. This is the key used to look up the sequence of steps to run from the BulletinBoardControl table in the database
- ◆ Destination – a set of parameters that defines what to do with the file after downloading it
 - ◆ TargetDirectory – a valid local directory that the LocalSystem Account has access to where the file should be saved.
 - ◆ TargetName – The file name to use in saving the file. See the FTP Transport documentation for a full description of the available parameters
 - ◆ DeleteSource – a Y or N that determines whether the Job should delete the file from the remote server after successfully downloading it. This is only used for the FTP-type Sources. For BulletinBoard Sources, deleting the original is accomplished through a step in the Bulletin Board sequence.

Exception Processing

There are two main mechanisms for dealing with exceptions. The primary mechanism is to simply log an error in the eGateway.txt log file and save the offending file to the eGateway\Problem folder. The problem with this is that then an administrator must periodically check the log file and the problem folder for errors.

In order to proactively notify system administrators about problems, with HIPAA Release 2 we introduced optional email alert notification. Configuration of Email Alerts is fully described in the Email Notification Release Note.

List updated 9/26 – based on HIPAA Release 2

Alert Listing

Alert ID	Alert Name	Alert Description
1	BAD_ROUTING_SEGMENTS	There's something wrong with the ISA, GS, or ST segments
2	INVALID_PARTNERS_IP_PORT_USER	Connection to a remote trading partner failed. This Alert will only be sent every EmailAlertFrequency (from the registry) seconds. If this registry entry is not defined, the alert will be sent once per hour.
3	CLAIM_TRACKING_LOG_FAILED	

Alert ID	Alert Name	Alert Description
4	DELIVERY_NOTIFICATION_FAILED	During batch file delivery, the sending eGateway notifies the receiving eGateway that a file has been sent. This Alert is sent when that notification doesn't get the expected "Ok" response
5	EDI_SIMPLE_SERVICE_FAILURE	An EDI Simple Service, like Medicaid or Medicare, has failed or automatically shut itself down.
6	TRANSACTION_VALIDATION_FAILED	
7	DUPLICATE_TRANSACTION_FOUND_DURING_TRACKING	
8	UNEXPECTED_DATA_RECEIVED_ON_LISTENING_SOCKET	An eGateway or other EDI Service listens for connect requests and data on a known socket. If non-X12 data is received on that socket, the service can report the issue.
9	BATCH_FILE_TRANSFORMATION_FAILURE	In the eGateway, the GetBatch() function failed to Load the received file or the Action list failed for some reason.

Alert Conditions

Condition	Alert Generated
File Tracking – Duplicate file found	7
File Tracking – Failed to update database	3
HIPAA Validation – File failed validation, if the 997 can be created, it will be attached to the email	6
Transaction Load – ISA segment not valid	1
Transaction Load – GS segment not valid	1
Validation – no SE segment seen	1
Transaction Load – GS doesn't immediately follow ISA	1
Check Addressing – Sending or Receiving Trading Partner not defined in Partners.txt	1
EDI Simple Service stopping – common situation is in the Medicare service where it automatically stops after three password failures to prevent password lockout	5
Connection failure to remote trading partner	2
eGateway - Batch transmission – FTP delivery succeeded, but delivery	4

Condition	Alert Generated
notification failed	
eGateway - Unexpected Data Seen on the Listening Socket (not an X12 transaction)	8
Check Addressing – Trading Partners and Transaction ID recognized, but not in the combination found in the interchange	1
Transaction Load - Multiple functional groups in the interchange. Because of the file transformation and splitting for routing and mapping it might involve, the eGateway doesn't support routing interchanges with multiple functional groups	1
eGateway – Batch delivery receipt failure. If the eGateway receives a “Batch file delivered” message, but the receiving eGateway can't find the alleged delivered file, or the file is the wrong size, it will send this message	4
eGateway – Batch file failed on Load or Transformation	9