

CHANGE LOG

Tab 2, Introduction

Section/ Paragraph	Area to Change/Comment	Suggested Change
Header	Replace WGQ with REQ	REQ for WGQ
17	Greater gas industry	Replaced with greater gas and electric industries
[all]	Most, but not all, references to WGQ	Replaced most instances of WGQ with REQ
17	Marketplace for natural gas	Replaced by marketplace for energy
[all]	Most references to Internet EDI/EDM and BATCH FF	Replaced with Electronic Delivery Mechanism (EDM)
18	TAB 7, TAB 8, TAB 9 and TAB 10	Replaced with TAB 7 Testing Guidelines and TAB 8 Appendix
18	Appendix C and Appendix D	Eliminated, and renamed Appendix E to Appendix C

CHANGE LOG

Tab 3, Executive Summary

Section/ Paragraph	Area to Change/Comment	Suggested Change
Header	Replace WGQ with REQ	REQ for WGQ
19	Added sentence introducing the concept of NAESB, as should be present in all Exec Summaries.	Selected sentence off NAESB website explaining NAESB.
[all]	Most, but not all, references to WGQ	Replaced most instances of WGQ with REQ
[all]	Numerous references to Batch FF/EDM	Removed references to Batch FF/EDM, following lead of Tab 6.
19,20	Spelled out all acronyms the first time they appeared, as should be the case with an Exec Summary	NAESB, all 4 quadrants, EDM, EDI, FTTF
19	Identified who benefits from standards	Added Gas and Electric utilities, changed the word 'bank' to 'financial institution.
20	More precise wording on who will require consensus	Added 'international community'.
20	Identify testing partners	Changed Gas to Electric
21	Concerns about reliability	Updated wording to more accurately describe the current state of affairs with the Internet



CHANGE LOG**Tab 6**

Section/ Paragraph	Area to Change/Comment	Suggested Change
[all]	Numerous references to WGQ,	Replaced WGQ with REQ
[all]	Numerous references to Batch FF/EDM	Removed references to Batch FF/EDM
Page 51	Extraneous “to”	Removed
Page 52	Common Code Identifier format	Tagged as OPEN ISSUE
[all]	Numerous references to gisb- acknowledgment-receipt	Tagged as OPEN ISSUE
Page 52	Description of input-format data element incorrect for REQ use	Removed reference to FF, added XML
[all]	Numerous references to CDI/script	Removed references to CGI/script
Page 52	request-status	Removed reference to decryption process
Page 53	time-c data element lacks time-zone indicator	Added time-zone indicator
Page 53	Transaction-set data element requires enhancement to remove gas industry specific references	Changed transaction-set to 16 character free form text field
Page 54	Diagram	Added EDM Server and simplified
Page 58	Reference to pipeline under Throughput Considerations	Removed pipeline reference
Page 58	HTTP Request Data Elements	Brief description added
Page 59	Incorrect description of transaction-set for REQ purposes	Provided new description for transaction-set, and tagged 8-character names as an OPEN ISSUE
Page 59	Writing a Batch Browser	Removed reference to NAESB home page
Page 59	Description of content type line	Rephrased to indicate that this referred to the specific example on page 59
Page 60	Description of content length	Rephrased to indicate that this referred to the specific example earlier on page 59
[all]	Several references to version 1.4	Changed to 1.6
Page 65	Reference to multipart POST	Tagged implementation as an OPEN ISSUE
Page 67	References to Central time zone	Removed restriction to use Central time
Page 67	Synchronization of client	Added reference to clock accessible via the Internet
Page 67	References to gas nominations	Removed references to gas

REVISIONS
CHANGE LOG
Tab 2, Introduction

Section/ Paragraph	Area to Change/Comment	Suggested Change
Header	Replace WGQ with REQ	REQ for WGQ
17	Greater gas industry	Replaced with greater gas and electric industries
[all]	Most, but not all, references to WGQ	Replaced most instances of WGQ with REQ.
17	Marketplace for natural gas	Replaced by marketplace for energy
[all]	Most references to Internet EDI/EDM and BATCH FF	Replaced with Electronic Delivery Mechanism (EDM)
18	TAB 7, TAB 8, TAB 9 and TAB 10	Replaced with TAB 7 Testing Guidelines and TAB 8 Appendix
18	Appendix C and Appendix D	Eliminated, and renamed Appendix E to Appendix C

INTRODUCTION

The North American Energy Standards Board (NAESB) is a voluntary non-profit organization comprised of members from all aspects of the greater gas and electric industries. NAESB Retail Electric Quadrant (REQ) Standards are a product of the North American Energy Standards Board. The NAESB mission is to take the lead in developing and implementing standards across the industry to simplify and expand electronic communication, and to streamline business practices. This will lead to a seamless North American marketplace for energy, as recognized by its customers, the business community, industry participants and regulatory bodies.

The standards are written as 'minimums,' which industry participants are encouraged to exceed (if they are not doing so already) through provision of value-added services and customized arrangements. NAESB defines 'exceed the minimum standard' to mean surpassing the standards without negative impact on contracting and non-contracting parties.

All of the standards have been adopted in the realization that as the industry evolves and uses the standards, additional and amended NAESB standards will be necessary. Any industry participant seeking additional or amended standards (including principles, definitions, standards, data elements, process descriptions, technical implementation instructions) should submit a request to the NAESB office, detailing the change, so that the appropriate process may take place to amend the standards.

TAB 1 Version Notes

Contains notes about this version, and, if appropriate, a brief summary of changes from the immediately preceding version.

TAB 2 Introduction

Provides a background statement about NAESB's Mission and the underlying concepts behind the design and use of this guide.

TAB 3 Executive Summary

Provides a brief outline of the industry business situation which is the basis for development of this guide.

TAB 4 Business Process & Practices

Provides a brief overview of the business process and the NAESB REQ approved principles, definitions and standards related to the business process covered by this guide.

TAB 5 Related Standards

Provides a reference to any related standards.

TAB 6 Technical Implementation - Electronic Delivery Mechanism (EDM)

Provides an overview of the business process for EDM.

Data Dictionary

Provides definition of the standard data elements and the usage requirements for each element.

Batch Flow Diagram

Sending Transactions

Provides instructions to develop mechanisms for sending of NAESB REQ standard format data files.

Receiving Transactions

Provides instructions to develop mechanisms for receiving of NAESB REQ standard format data files.

Security

Provides guidelines for data privacy, data integrity, authentication and non-repudiation of inbound and outbound transactions.

Other Considerations

Provides information regarding error notification and testing. Includes a reference guide and examples for repudiation and validation.

TAB 7 Testing Guidelines

TAB 8 Appendix

- Appendix A - Reference Guide
- Appendix B - Repudiation and Validation Examples
- Appendix C - Minimum Technical Characteristics for an EDM Server+

CHANGE LOG

Tab 3, Executive Summary

Section/ Paragraph	Area to Change/Comment	Suggested Change
Header	Replace WGQ with REQ	REQ for WGQ
19	Added sentence introducing the concept of NAESB, as should be present in all Exec Summaries.	Selected sentence off NAESB website explaining NAESB.
[all]	Most, but not all, references to WGQ	Replaced most instances of WGQ with REQ.
[all]	Numerous references to Batch FF/EDM	Removed references to Batch FF/EDM, following lead of Tab 6.
19,20	Spelled out all acronyms the first time they appeared, as should be the case with an Exec Summary	NAESB, all 4 quadrants, EDM, EDI, FTTF
19	Identified who benefits from standards	Added Gas and Electric utilities, changed the word 'bank' to 'financial institution.
20	More precise wording on who will require consensus	Added 'international community'.
20	Identify testing partners	Changed Gas to Electric
21	Concerns about reliability	Updated wording to more accurately describe the current state of affairs with the Internet

EXECUTIVE SUMMARY



The North American Energy Standards Board ~~Retail Electric Quadrant Whole Gas Quadrant~~ (NAESB ~~REQWGQ~~) has developed standards for accomplishing electronic commerce over the Internet. ~~using ANSI ASC X12 (EDI/EDM flat files (FF/EDM), and Customer Activities Web site presentations (EBB/EDM).~~ Technologies necessary for all Internet Electronic Delivery Mechanisms (EDM) to rapidly, reliably and safely move data across the Internet have been determined. For ~~data EDI and flat files, once~~ received from a trading partner via the Internet, the data is decrypted and moved through a translator or other appropriate processor for NAESB ~~WGREQ~~ standard file formats and forwarded to a back-end processing application. However, file format translation and back-end processing are outside the Internet EDM scope. ~~For NAESB WGQ Customer Activities and Informational Posting Web sites, requirements for data presentation, navigation and session security have been determined.~~

This document is a high-level guide to implementing various technologies necessary to communicate transactions using the standard protocols. As such, this guide is not intended to be a comprehensive, in-depth manual. Wherever possible, this guide points to more in-depth material. The Reference section provides locations on the Internet to obtain more information as well as books and periodicals that have been recommended.

Open Standards

There are several major topic areas related to Internet Electronic Delivery Mechanism covered in this manual. When looking to implement Internet EDM, one should become familiar with the following components of the implementation:

Communications Protocols

Sending of Transactions

Receipt of Transactions

Security

HTTP Transport for Secure EDI (a.k.a. IETF EDIINT AS2)

The "open" standard technologies selected by NAESB ~~WGREQ~~ to address these areas are designed to provide flexibility and scalability. There are business benefits gained from adherence to "HTTP Transport for Secure EDI" such as:

Allows potential to more readily, electronically trade with others (e.g., ~~electric gas~~ utilities, ~~financial institutions~~ banks, suppliers, retail customers)

Makes it more likely that packages can be purchased to replace custom written ~~applications~~ currently in place to support NAESB ~~WGQ REQ~~ EDM

Strengthens the surety of receipt and error notification

HTTP Transport for Secure EDI (AS2) is an emerging standard, largely based on the original

NAESB WGQ EDM, that is being developed by the Internet Engineering Task Force, the Internet standards body. Adherence with a formal, international Internet standard, such as AS2 ensures that the specification will not change without due process and any changes that do occur will be the result of a broad consensus in the international community. Individual companies and entire industries are free to use as much or as little of AS2 as they see fit, providing the maximum flexibility to meet business needs. The specific implementation of the standards is dependent upon what fits the trading partner's needs and available resources. A brief delineation of these components is covered at a high level in the Business Process and Practices (Major functions of Internet EDM covered by the Standards) section and in more detail in later sections.

Same Application Implementation For All Trading Partners

The basic assumption in designing and implementing the Internet EDM application is that it is not platform-specific. What is meant by this is that an organization's Internet EDM application serves the role of communicating with all trading partners in the electricgas industry no matter what hardware, operating system and programming languages they use at their site. For this reason, testing with other trading partners with a variety of platforms is very important in ensuring that eachyour EDM application is compatible with a range of platforms used by various trading partners.

Testing With ElectricGas Industry Internet EDM Participants

(THIS SECTION NEEDS TO BE REWRITTEN REFERRING TO OUR GUIDELINES (16 03))

To provide a way for parties interested in Internet EDM testing to initiate testing relationships, the NAESB home page will have a list of organizations willing to act as testing partners and their respective test coordinator. — The Future Technology Task Force (FTTF) meets on an intermittent basis be scheduled teleconference or in-person meetings to discuss issues, problems, further refinement of the standards. These discussions will provide a means to benchmark results and provide feedback to each other on possible enhancements to the participants' implementations. The FTTF realized that the technology being implemented is relatively new and all organizations can benefit from the sharing of research and technical information and the resolution of gas business issues integrated with the new technologies.

Importance of the Trading Partner Agreement When Using EDM

The expectations of who will perform what function and how it will be accomplished in Internet EDM should, at some level, be laid out in the trading partner agreement. This clarification in the agreement would help to expedite a smoother communication between the trading partners when first setting up their Internet EDM relationship. The newness of the Internet EDM standards and the various implementations of the applications between trading partners bring to the forefront a quandary of issues related to establishing the business rules associated with these standards. The specifications in the trading partner agreement should be tested before production implementation to formulate a solution to any problems revealed during testing well before reliance on the implementation.

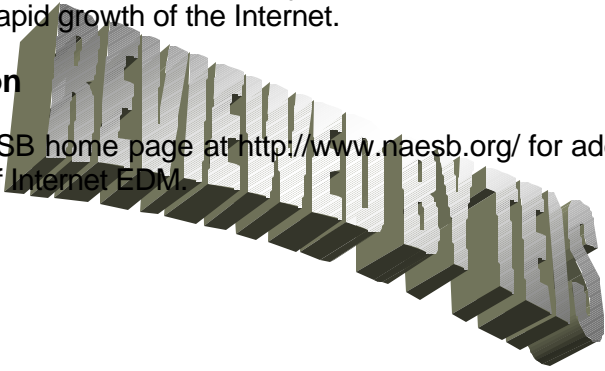
Concerns About Future Reliability of the Public Internet

The Internet has proven its viability as a business communication method in many different

industries, but Continued monitoring of the Internet's viability as a communications medium infrastructure will continue to take place. Increased traffic and potential lack of sufficient transmission capacity on the Internet is difficult to predict and quantify at this time. Concerns may be resolved by new Internet service providers and new communications technologies to compensate for the rapid growth of the Internet.

Further Information

Please see the NAESB home page at <http://www.naesb.org/> for additional useful information on the implementation of Internet EDM.



RELATED STANDARDS

Common Codes

~~A decision made in 1994 by a FERC-established standards development group (EBB Working Group 5) resulted in a location coding system which cross-references proprietary point codes to a common industry-supported location code. This common location code, called the GRID Code, was developed based on the American Petroleum Institute (API) well code model. The FERC, in Order 563-A, directed the industry to establish any necessary relationships and to proceed with the implementation of the GRID Code. To achieve this implementation, in August 1994 trade associations representing three segments of the natural gas industry entered into an agreement with Petroleum Information Corporation (PI) to develop and maintain the PI GRID™ Common Code database. As GISB prepared standards for capacity release (July 1995) and nominations (September 1995), GISB fully endorsed the use of the PI GRID™ common codes.~~

~~However, after extensive consideration by GISB's Common Code Subcommittee, GISB adopted, on September 30, 1996, a new Common Code for Gas Transaction Points, the NAESB WGQ/PI Data Reference Number (generally referred to as "DRN"). The DRN is a one-to-nine digit, non-intelligent number also assigned by IHS (successor to PI), which has a one-to-one relationship with the PI GRID™ Code. For more information, access the NAESB Web Page at www.naesb.org.~~

~~In keeping with the trends in other industries involved with EDI, EBB Working Group 5REQ TEIS recommended recommends the acceptance of the D-U-N-S®¹ Number as a common company identifier. (OPEN ISSUE 001) This recommendation was also adopted in FERC Order 563-A. The D-U-N-S® Number is assigned to companies by the Dun & Bradstreet Corporation (D&B). Similarly, as GISB NAESB prepared standards for capacity release (July 1995) and nominations (September 1995), GISB fully endorsed the use of the D-U-N-S® Number common code.~~

For NAESB ~~WGQ-REQ~~ Common Code purposes, an entity will use one and only one D-U-N-S® Number. Entity common codes should be "legal entities," that is, Ultimate Location, Headquarters Location, and/or Single Location (in ~~D&B Dun & Bradstreet Corporation ("D&B")~~ terms). However, in the following situations, a Branch Location (in D&B terms) can also be an entity common code:

- ~~1. 1-~~When the contracting party provides a D-U-N-S® Number at the Branch Location level;
of.
- ~~2. 2-to To~~ accommodate accounting for an entity that is identified at the Branch Location level.

~~-Since D&B offers customers the option of carrying more than one D-U-N-S® Number per entity, please refer to NAESB's Web Page at www.naesb.org for directions on determining the one and only one D-U-N-S® Number constituting the NAESB ~~WGQ-REQ~~ Entity Common Code. (OPEN ISSUE 001, OPEN ISSUE 021)~~

~~In the datasets, an asterisk by a data element means that it is a "common code," so the field will reflect the industry-supported common code for location or company.~~

1 D-U-N-S® is a registered trademark of Dun & Bradstreet, Inc.

~~NAESB WGQ Electronic Data Interchange Trading Partner Agreement~~

~~In 1998, GISB adopted Standard 6.3.3, the NAESB WGQ Electronic Data Interchange Trading Partner Agreement (TPA) for exchange of data within the gas industry. The NAESB WGQ TPA defines the relationship of the sender and receiver of NAESB WGQ Standard ASC X12 documents. This agreement represents a complete set of balanced terms which a company should accept whether it is sender or receiver of electronic documents. It has established all the data items necessary to exchange electronic documents in a step by step, fill in the blank model form. The use of the TPA minimizes preparation, negotiation and review time. This will allow more time for implementation of electronic commerce. Copies of this agreement may be obtained from the NAESB office or may be downloaded from the NAESB home page at www.naesb.org.~~

~~Party Roles~~

~~In all of the transaction sets, there are multiple parties that may be involved in the transaction. There are [distribution companies, end-users, regulatory entities, service providers, and suppliers](#). [\(OPEN ISSUE 022\) utilities, marketers, government entities, and various service providers](#). [The Utilities and Marketers are self explanatory. The government entities may be a state commission such as ERCOT in Texas. The service providers can be a CIS provider and/or ED/EDM providers.](#) There are the Transportation Service Provider (a.k.a. Pipeline or Transporter), the Service Requester (a.k.a. Shipper), Service Requester Agent (a.k.a. Shipper's Agent) and Third Party Service Provider (a.k.a. Third Party Agent). It is important to distinguish between the role of the Service Requester Agent and the Third Party Service Provider.~~

~~[The service providers normally will not be identified in the envelope or body of the transaction. The utility, marketer, and government entity will be identified both in the envelope, or ISA/GS segments and the body of the transaction in the N1 name segments.](#)~~

~~The Service Requester Agent is the party contractually authorized by the Service Requester to submit business transactions to the Transportation Service Provider on behalf of the Service Requester for a service requester contract. Once the Service Requester Agent is contractually authorized, the agent becomes the Service Requester for subsequent business transactions unless and until the agency relationship is terminated.~~

~~The Third Party Service Provider is the communications agent that the Service Requester or Service Requester Agent may subscribe to in order to send and receive transactions with the Transportation Service Provider.~~

~~It is possible that a single entity may, at times, provide the role of a Service Requester Agent for one party while providing the role of Third Party Service Provider for another party. Likewise, a single entity could be both Service Requester Agent and Third Party Service Provider for a single party.~~

~~In EDI implementation, the party that is authorized to send and receive transactions will be the party identified in the transmission envelope (ISA Header Segment). If the sending party is a Service Requester, Service Requester Agent or Third Party Service Provider, their appropriate~~

~~identifiers will appear here. In all cases, the Transportation Service Provider, Service Requester and Service Requester Agent (if applicable) will be identified in the body of the transaction (N1 Name Segment).~~

ANSI ASC X12 Standards

The NAESB ~~WGQ~~REQ standards reflect an industry utilization of the American National Standards Institute (ANSI) ASC X12 standards maintained by the Data Interchange Standards Association, Inc. (DISA). ~~The technical implementation documents included in this manual reflect the NAESB WGQ REQ subset of the ANSI ASC X12 standards versions.~~ It is recommended that any industry participant who wishes to utilize the ANSI ASC X12 standards should also have a copy of the ANSI ASC X12 Standards Reference document for a full understanding of the X12 requirements. NAESB members may purchase an ANSI reference document through NAESB by contacting the NAESB office. Non-NAESB industry participants may purchase the reference document by contacting:

Manager of Publications
DISA
333 John Carlyle Street, Suite 600
Alexandria, VA 22314
Voice: 703-548-7005
Fax: 703-548-5738
www.disa.org

As a member of ANSI, NAESB ~~WGQ~~REQ will utilize the ANSI ASC X12 standards and remain in full compliance. In all standards, occasions arise where the standard does not fully meet a need. NAESB ~~WGQ~~REQ recognizes this and will add interim usages and code values when required. When NAESB ~~WGQ~~REQ utilizes an interim solution, NAESB ~~WGQ~~REQ will apply to ANSI and the appropriate ANSI organizations for acceptance of the interim solution. ANSI's final solution may provide a usage or code value different than the interim solution. NAESB ~~WGQ~~REQ standards will be updated to reflect the final solution. ([OPEN ISSUE 023](#))

The architecture of ASC X12 is designed for end to end communications. The translator that generates the ASC X12 file and envelope will assign control numbers and counts that will appear within the ISA/IEA segments of the transaction and within the GS/GE segments of the transaction. These numbers and counts allow the translator to ensure that all of the segments in an envelope and all of the data elements in an envelope have been received and that the transmission was complete.

ISA contents

The ISA segment marks the beginning of an X12 document. It can be equated to an envelope that a paper document would come in via the mail. The envelope may contain one or more functional groups (defined by the GS segment) and one or more transaction sets.

The ISA is the interchange control segment to be utilized on all NAESB ~~WGQ~~REQ X12 standards. ([OPEN ISSUE 023](#)) The segment identifies the sender and receiver of the document. The Interchange Sender ID/Interchange Receiver ID is published by both the sender and receiver for other parties to use as the sender/receiver ID to route data to them. The

sender must always code the sender's ID in the sender element and the designated receiver's ID in the receiver ~~Element~~. Trading partners utilizing a password for their documents will use the Security Information element. The receiver of the document identifies a password for the sender to include in this element. ~~(OPEN ISSUE 024) This sender and receiver information is specified in the NAESB WGQ-REQ Electronic Data Interchange Trading Partner Agreement.~~

There are additional elements in the ISA segment. These elements are traditionally assigned by the sending party's translator. These elements inform the receiver of the date/time that the envelope was generated, the X12 version number being utilized, whether the transmission is for test or production purposes, and what characters were used to designate the end of a sub element, element or segment. Different characters must be chosen for the sub element, element and segment delimiters. These delimiting characters must never appear in the data.

For more information on the ISA segment and the possible values for its elements, contact DISA at the above address or consult the appropriate version of the ANSI ASC X12 Standards Reference document corresponding to the NAESB ~~WGQ-REQ~~ transaction set being sent/received. Information about control segments (including the ISA and IEA) can be found in the Overview/Introduction and Control Standards sections of the reference document. Specific information about the ISA and IEA segments and corresponding elements can be found in the Segment Directory and Data Element Dictionary sections.

GS contents

The GS segment indicates the beginning of a functional group and provides control information for the data that follows it. A functional group can be defined as a group of transactions related to one business application. Within a mailing envelope, there may be a bundle of information relating to ~~imbalances-enrollments~~ and a bundle of information relating to ~~measurement information usages~~. ~~(OPEN ISSUE 025)~~ Each of these 'bundles' is sent within its own (or a separate) GS Functional Group Header and a GE Functional Group Trailer in the X12 environment. The sender of a transmission provides the Application Sender's Code that the receiver of the transmission will reflect back on acknowledging documents. The receiver of a transmission provides the Application Receiver's Code that the sender will include in the transmission for the receiver to utilize in routing to internal applications. Group Control Numbers are originated and maintained by the sender of the document.

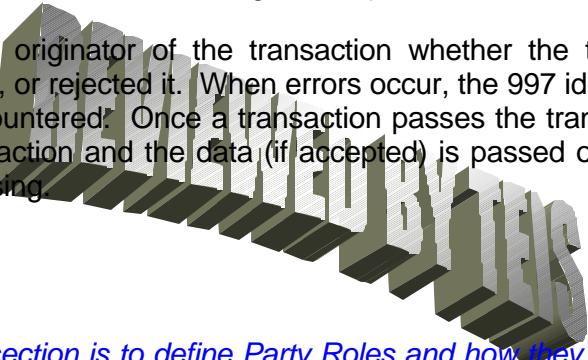
For more information on the GS segment and the possible values for its elements, contact DISA at the above address or consult the appropriate version of the ANSI ASC X12 Standards Reference document corresponding to the NAESB ~~WGQ-REQ~~ transaction set being sent/received. ~~(OPEN ISSUE 023)~~ Information about control segments (including the GS and GE) can be found in the Overview/Introduction and Control Standards sections of the reference document. Specific information about the GS and GE segments and corresponding elements can be found in the Segment Directory and Data Element Dictionary sections.

997 Usage

The 997 Functional Acknowledgment is used to indicate the results of the syntactical analysis of the X12 documents. The documents include the transaction sets and functional groups with an ISA/IEA envelope. This standard covers all of the X12 and NAESB ~~WGQ-REQ~~ standard criteria that the receiver of the document has incorporated into the receiver's translator. The translator may be set to accept all information into the receiver's application processing, it may be set to accept only ANSI ASC X12 compliant information into the receiver's application processing, or it

may be set to accept only ANSI ASC X12 and NAESB ~~WGQ~~REQ compliant information into the receiver's application processing. Compliance checking, in a translator, may be set to any of several levels. NAESB ~~WGQ~~REQ recommends that compliance checking be set to the element level in the Functional Acknowledgement. ([OPEN ISSUE 023](#))

The 997 informs the originator of the transaction whether the translator accepted the file, accepted it with errors, or rejected it. When errors occur, the 997 identifies the location and type of error that was encountered. Once a transaction passes the translator, the 997 is sent to the originator of the transaction and the data (if accepted) is passed on to the receiver's business application for processing.



[XML Standards](#)

{The purpose of this section is to define Party Roles and how they are implemented in the XML transactions. ([OPEN ISSUE 019](#))}

Hypertext Transfer Protocol (HTTP)

The Hypertext Transfer Protocol (HTTP) is an application-level protocol with the lightness and speed necessary for distributed, collaborative, hypermedia information systems. It is a generic, stateless, object-oriented protocol which can be used for many tasks, such as name servers and distributed object management systems, through extension of its request methods (commands). A feature of HTTP is the typing of data representation, allowing systems to be built independently of the data being transferred.

HTTP has been in use by the World-Wide Web global information initiative since 1990. Appendix A of the [NAESB REQ](#) Electronic Delivery Mechanism Related Standards manual contains a listing of the HTTP version(s) ~~supported by NAESB~~ ~~WGQ~~REQ.

~~HTTP transaction-set Code Values~~

~~The following table contains a list of code values to be used with the transaction-set data element, which is a mutually agreeable (MA) data element in the HTTP Request.~~

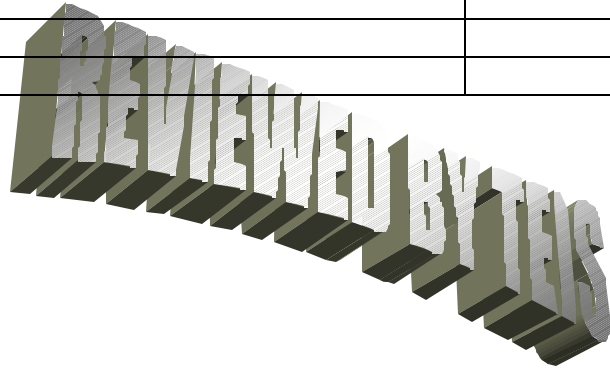
HTTP transaction-set Code Values	NAESB-WGQ Standard Number	Transaction Set Description
G873NMST	4.4.1	Nomination
G874NMQR	4.4.2	Nomination Quick Response
G873RQCF	4.4.3	Request for Confirmation
G873RRFC	4.4.4	Confirmation Response
G873SQTS	4.4.5	Scheduled Quantity
G873SQOP	4.4.6	Scheduled Quantity for Operator
G874CRQR	4.4.7	Confirmation Response Quick Response
G860PDAL	2.4.4	Pre-determined Allocation

HTTP transaction-set Code Values	NAESB-WGQ Standard Number	Transaction Set Description
G866PDQR	2.4.2	Pre-determined Allocation - Quick Response
G865ALLC	2.4.3	Allocation
G841IMBL	2.4.4	Shipper Imbalance
G867MSIN	2.4.5	Measurement Information
G867MAUS	2.4.6	Measured Volume Audit Statement
G844RQIN	2.4.7	Request for Information
G844RRIN	2.4.8	Response to Request for Information
G841TSIN	3.4.1	Transportation/Sales Invoice
G820PYRM	3.4.2	Payment Remittance
G822STAC	3.4.3	Statement of Account
G841SRCA	3.4.4	Service Requester Level Charge/Allowance Invoice
G840CROF	5.4.1	Offer Download
G843CRBR	5.4.2	Bid Download
G843CRAN	5.4.3	Award Download
G832CRRC	5.4.4	Replacement Capacity
G843CRWD	5.4.5	Withdrawal Download
G840UPWD	5.4.6	Withdrawal Upload
G840UDOF	5.4.7	Offer Upload
G843UDVL	5.4.8	Offer Upload-Quick Response
G840UDRC	5.4.9	Offer Upload Notification
G843UDBC	5.4.10	Offer Upload Bidder Confirmation
G824UDCV	5.4.11	Offer Upload Bidder Confirmation Quick Response
G567UDFD	5.4.12	Offer Upload Final Disposition
G840OAUC	5.4.13	Operationally Available and Unsubscribed Capacity
G846UPRD	5.4.14	Upload of Request for Download of Posted Datasets
G846RURD	5.4.15	Response to Upload of Request for Download of Posted Datasets
G864SWNT	5.4.16	System-Wide Notices
G864CRNS	5.4.17	Note/Special Instruction
G843BDUP	5.4.18	Bid Upload
G843BDQR	5.4.19	Bid Upload Quick Response
G997FNAK	N/A	Functional Acknowledgement

CHANGE LOG

Tab 6

Section/ Paragraph	Area to Change/Comment	Suggested Change
[all]	Numerous references to <u>WGQ</u> .	Replaced <u>WGQ</u> with <u>REQ</u>
[all]	Numerous references to Batch FF/EDM	Removed references to Batch FF/EDM
Page 51	Extraneous “to”	Removed
Page 52	Common Code Identifier format	Tagged as OPEN ISSUE
[all]	Numerous references to gisb- acknowledgment-receipt	Tagged as OPEN ISSUE
Page 52	Description of input-format data element incorrect for <u>REQ</u> use	Removed reference to FF, added XML
[all]	Numerous references to CDI/script	Removed references to CGI/script
Page 52	request-status	Removed reference to decryption process
Page 53	time-c data element lacks time-zone indicator	Added time-zone indicator
Page 53	Transaction-set data element requires enhancement to remove gas industry specific references	Changed transaction-set to 16 character free form text field
Page 54	Diagram	Added EDM Server and simplified
Page 58	Reference to pipeline under Throughput Considerations	Removed pipeline reference
Page 58	HTTP Request Data Elements	Brief description added
Page 59	Incorrect description of transaction-set for <u>REQ</u> purposes	Provided new description for transaction-set, and tagged 8-character names as an OPEN ISSUE
Page 59	Writing a Batch Browser	Removed reference to NAESB home page
Page 59	Description of content type line	Rephrased to indicate that this referred to the specific example on page 59
Page 60	Description of content length	Rephrased to indicate that this referred to the specific example earlier on page 59
[all]	Several references to version 1.4	Changed to 1.6
Page 65	Reference to multipart POST	Tagged implementation as an OPEN ISSUE
Page 67	References to Central time zone	Removed restriction to use Central time
Page 67	Synchronization of client	Added reference to clock accessible via the Internet
Page 67	References to gas nominations	Removed references to gas nominations



~~TECHNICAL IMPLEMENTATION - INTERNET EDM~~ ~~IMPLEMENTATION - INTERNET EDI/EDM & BATCH FF/EDM~~

Technologies Selected by NAESB ~~REQWGQ~~

The transport protocol for communication of future NAESB ~~WGQ-REQ~~ transactions should be TCP/IP. In addition, standard Internet protocols should be chosen for specific tasks. Various Internet protocols were considered to accomplish the delivery of a transaction at the application protocol level. The Hyper-Text Transfer Protocol (HTTP) was chosen.

Practical information systems require more functionality than simple retrieval, including search, front-end update, and annotation. HTTP allows an open-ended set of methods to be used to indicate the purpose of a request. HTTP is also used as a generic protocol for communication between user agents and proxies/gateways to other Internet protocols, allowing basic hypermedia access to resources available from diverse applications and simplifying the implementation of user agents.

There are two primary Internet software components involved in Web communications. The first is called a browser and runs as client software. The second is called a Web server, or HTTP server and usually runs on a dedicated server computer.

The standard data elements, each with element name and description, have been defined in the Section "Data Dictionary For Internet EDM". The following two sections identify what is involved in sending and receiving transactions. After that comes a discussion regarding the securing of the transactions to be sent. The remaining sections cover considerations for other aspects of the overall process. While these were not the focus of the Internet EDM process as mentioned above, selected topics that may affect your overall implementation are discussed.

Data Dictionary For Internet EDM

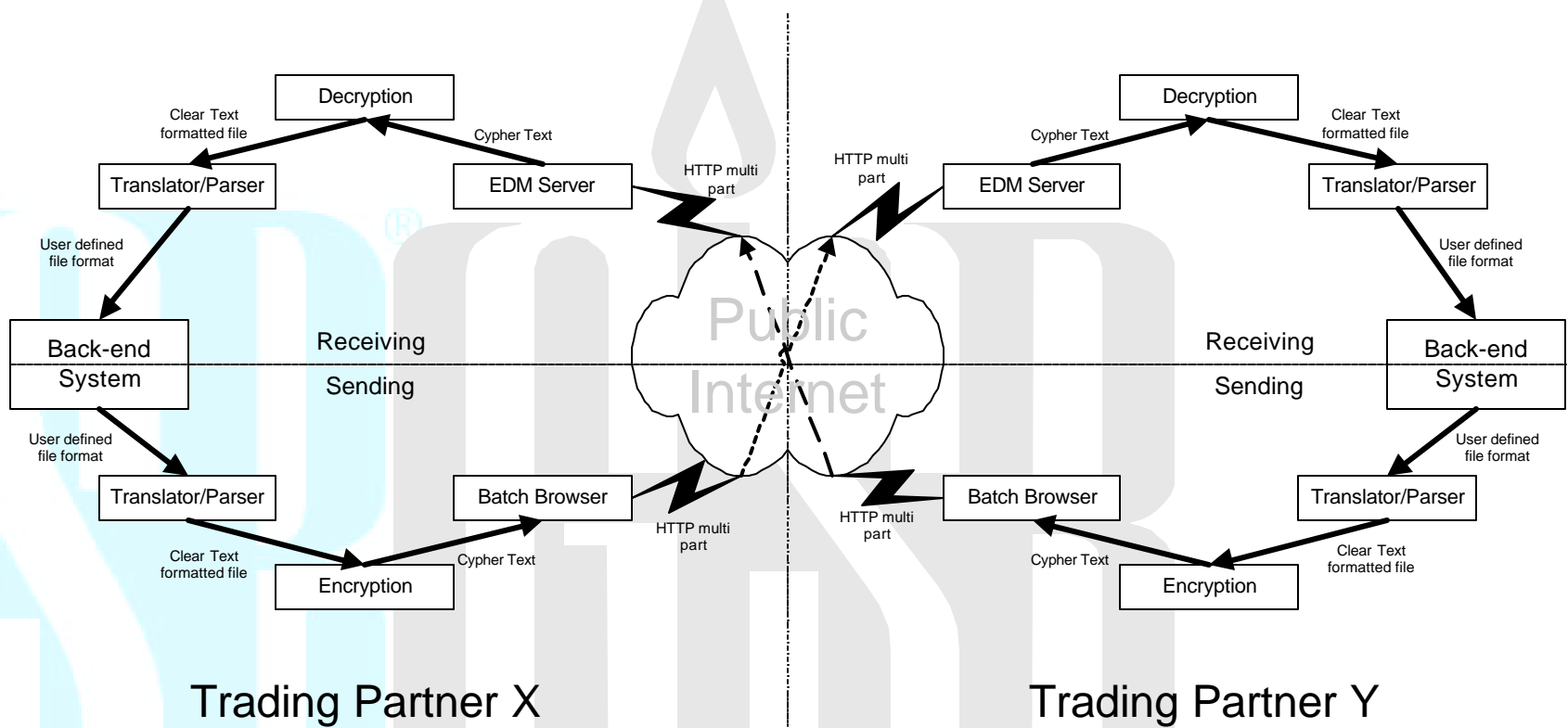
Business Name	Definition	Format	Usage*	Condition
from**	the party sending the transaction	Common Code Identifier format (<u>OPEN ISSUE 001</u>)	in Request; M	used in file transmittal; displayed in HTTP response; and, used in posting back decryption-related errors
input-data	the filename for the transaction data set transmitted	including drive letter and directory name with filename if needed	in Request; M	used in file transmittal of any transaction data sets; and, used for posting back all transaction value pairs for a transmittal that had decryption-related errors.
input-format	descriptor of the data format used for the file transmitted	X12 ; <u>XML;FF</u> ;error	in Request; M	“X12”, “XML”, “FF”, or other NAESB <u>REQWGQ</u> standard format indicator used in file transmittal; “error” used in posting back any decryption-related errors
receipt-disposition-to	the party to receive receipts, the value should be the same as the “from”	Common Code Identifier format (<u>OPEN ISSUE 001</u>)	in Request; M	used in file transmittal and in posting error notifications
receipt-report-type	type of receipt type being requested by sender	gisb-acknowledgement-receipt (<u>OPEN ISSUE 002</u>)	in Request; M	used in file transmittal and in posting error notifications
receipt-security-selection	used to request signed receipts	signed-receipt-protocol=required,pgp-signature;signed-receipt-micalg=required,md5	in Request; MA	used in file transmittal and in posting error notifications
refnum	used by the party to assign a unique message identifier for tracing purposes	maximum 40 character integer value	in Request; MA	May be used by sender to send tracking information to a recipient. Use of this data element is by mutually agreed. This data element is conceptually similar to a Message-ID filed within RFC 822.
request-status	status describing success or failure of transmission at recipient server	ok; EEDM###:error description; WEDM###:warning description. see Table A, “Internet EDM Standard Error Codes and Messages”	in Response; M	“ok” is returned if all is fine with the CGI/script processing; error messages/warnings and their related descriptions are returned if problems were encountered in <u>CGI/script</u> processing, or in the decryption process.
server-id	uniquely identifies the server and CGI/script processing the transaction	domainname or hostname.domainname; no embedded spaces allowed	in Response; M	displayed in the HTTP response and posted back for any decryption-related errors
time-c	the time file transfer is complete at the server, <u>where + or -ZZ indicates delta from UTC (ref ISO 8601)</u>	Y yyymmddhhmmss- <u>ZZ</u> ; yyymmddhhmmss+ <u>ZZ</u> <u>(OPEN ISSUE 010)</u>	in Response; M	displayed in the HTTP response and posted back for any decryption-related errors
to **	the party the transaction was sent to	Common Code Identifier format (<u>OPEN ISSUE 001</u>)	in Request; M	used in file transmittal and displayed in HTTP response and posted back for any decryption-related errors

Business Name	Definition	Format	Usage*	Condition
transaction-set	name of the document type being sent	8 character code; (OPEN ISSUE 003) examples are: G873NMST , G873RQCF , etc.; refer to NAESB REQWGQ Implementation Guide, Related Standards Tab, Hypertext Transfer Protocol (HTTP) section, HTTP transaction-set Code Values table.	in Request; MA	used in file transmittal
trans-id	sequential number assigned to the transaction by the server CGI/script upon processing before being passed to the decryption process	integer up to 15 characters in length	in Response; M	displayed in the HTTP response and posted back for any decryption-related errors
version	the NAESB REQWGQ EDM version being used by the sender	numeric, decimal notation (e.g. 1.64)	in Request; M	used in file transmittal and in posting error notifications

*The **Usage** column defines whether the element appears in the HTTP Request (Client-generated) or the HTTP Response (Server-generated), the order in which the element appears in the data stream, and whether the field is Mandatory (M) or Mutually-Agreed-To (MA).

** Common Code Identifier ([OPEN ISSUE 001](#))

Batch Flow Diagram



Batch Flow Diagram

SENDING TRANSACTIONS

General Flow

The following is an example of the steps necessary to send an [Internet EDI/EDM file: and batch FF/EDM file:](#)

1. Open HTTP-connection ([OPEN ISSUE 004](#))
2. Check connection status. If in error, re-queue file according to NAESB [REQWGG](#) standards (this check should be performed here and throughout the following processes)
3. Post
 - A. Authentication (password must be base64-encoded)
 - B. Send multipart form
 - C. Receive HTTP response data
4. Check connection status. If in error re-queue file according to NAESB [WGQ-REQ](#) standards
5. Check HTTP status code (200 is good, less than 300 may be acceptable). If status is not successful re-queue file according to NAESB [WGQ-REQ](#) standards
6. Close connection - wait for other end to close in a reasonable time
7. Parse HTTP response data elements
8. If request-status ok, then log success
9. If request-status error, then log error
10. If no valid request-status re-queue file according to NAESB [REQWGG](#) standards
11. Remove file from sending queue when successful or when failed completely

If trading partners agree to implement signed receipts then the sending party must include the "receipt-security-selection" data element in the posted data. The receiving party must digitally sign the gisb-acknowledgement-receipt ([OPEN ISSUE 002](#)) and encapsulate the gisb-acknowledgement-receipt ([OPEN ISSUE 002](#)) and digital signature body parts within a MIME envelope with a Content-type of application/pgp-signature.

HTTP Post

Most people think of the Web as the process of using a browser to fetch, or download, documents, not upload them. Indeed, this capability is most prevalent. HTML pages, text files, and other documents can be retrieved by a browser using HTTP, FTP, or other protocols. However Web browsers allow the user to input data to a server using HTML forms. Data is entered into the fields of the form and is transmitted to the server by pressing a pushbutton or hitting the enter key.

The HTTP protocol has two methods for transmitting a request to a server. Both methods return a response to the client, which may be a document retrieved from the server. Both methods can be used to transmit form data. The GET method is the simplest and is used for requests that pass a small amount of information. Data passed with the GET method must be translated into a special format known as "URL encoding." Furthermore, the data stream transmitted by the GET method has a limit of 1024 characters. The POST method, on the other hand, allows the upload of complete datasets without special encoding. It is this method which will be used to send NAESB [REQWGG](#) standard format transactions and receive the response from the server.

Using an Interactive Browser

When most of us think of Web surfing, we think of using an interactive browser. When you enter an HTTP Uniform Resource Locator (URL), the browser opens the HTML document identified by the URL. Basically, a URL is an “address” of an HTML document on a Web server. For purposes of NAESB ~~WGQ~~~~REQ~~ standards Uniform Resource Locator (URL) is as defined by the Internet Engineering Task Force (IETF).

In order to use an interactive browser to upload data, an HTML document must be created for that function. The HTML document can reside on either the server to which you are uploading or the client’s system. The “form” feature of HTML allows that within an HTML document, a form can be created which allows the client to type in any necessary data elements, such as to, from, and input format and then specify a file to be uploaded from the PC. Some type of “Send” button would be on the form and when selected, the form would cause an HTTP POST to be issued, thereby uploading the file. Below is an example of an HTML document with a form which specifies the POST method and contains the required data elements.

An HTML form like that described here could be used with any retail browser that supports multipart POST with a file upload. When choosing a packaged browser, it is mandatory that it supports multipart encoding.

Sample of HTML document with a form to perform a multipart post using an interactive browser:

```

<HTML>
<HEAD>
<TITLE>NAESB REQWGQ File Upload</TITLE>
<H1><CENTER>NAESB WGQREQ File Upload</CENTER></H1>
</HEAD>
<HR>
<BODY>
<form ENCTYPE="multipart/form-data" ACTION="http://www.target.server/cgi-bin/upload.exe"
METHOD=POST>
Enter Common Code Identifier for From and To (OPEN ISSUE 001)
From: <input TYPE="text" NAME="from" SIZE=20 VALUE=""><br>
To: <input TYPE="text" NAME="to" SIZE=20 VALUE=""><br>
NAESB WGQREQ EDM Version: <input TYPE="text" NAME="version" SIZE=5 VALUE="1.64"><br>
Deliver Receipt To: <input TYPE="text" NAME="report-disposition-to" SIZE=20 VALUE=""><br>
Receipt Type: <input TYPE="text" NAME="receipt-report-type" SIZE=30
VALUE="gisb-acknowledgement-receipt"><br>(OPEN ISSUE 002)

IF requesting signed receipts also include:

Receipt Type: <input TYPE="text" NAME="receipt-security-selection" SIZE=30 VALUE="signed-receipt-
protocol=required, pgp-signature; signed-receipt-micalg=required, md5"><br>

Format of this file: <input TYPE="text" NAME="input-format" SIZE=6 VALUE="X12"><br>
Send this file: <INPUT NAME="input-data" TYPE="FILE"><br>
<input TYPE="submit" VALUE="Send File"><br>
</form>
</BODY>
</HTML>

```

The non-bolded text in this example is the basic HTML required for a document and allows your page to show a title in the title bar. The bolded text is the form within the document and is described in more detail.

The important characteristics of the form within the HTML document are:

ENCTYPE= specifies the encoding type. The “multipart/form-data” encoding type is identified as the standard encoding methodology.

- ACTION= specifies the URL that will receive the uploaded data. The Trading Partner Agreement([OPEN ISSUE 008](#)) identifies the URLs for both parties.
- METHOD= specifies the HTTP protocol method. “POST” has been defined as the NAESB [WGQ-REQ](#) standard method.
- <input ...> Five input areas are specified on this form: from, to, file format, file name, “Send File” button.

NOTE: This document often refers to “multipart POST” which implies the encoding type and method as described in this example.

When a user selects the “Send File” button, the browser will take the values entered in the input fields and reformat them according to the encoding type into a data stream. For the file identified for upload, the file is opened and its contents are included in the data stream, rather than the file’s name. The data stream is then sent to the URL specified by **ACTION=**. The URL will indicate an HTTP server script or program written to receive the data.

For a smaller site only performing a few transactions or file transfers this manual process would be viable as a primary transmission tool. This method could also be considered a back-up method to any batch or automated process that may be implemented. If the client provides its own form, the form can be copied for each trading partner. The only change to the HTML would be to modify the URL shown for the **ACTION=** attribute.

Using a Batch Browser

For companies that have automated much of their back-end process and prefer to avoid unnecessary human involvement, a so-called “batch browser” is needed. This browser needs to be capable of program-based or script-based initiation. At this time, there are few off-the-shelf batch browsers which use the POST method. Most packaged batch browsers use the GET method.

However, a batch browser can be created using custom programming. The batch browser will be coded to perform all of the same formatting that the interactive browser performed to send a data stream which conforms to the HTTP protocol. A batch browser must be coded as a sockets program. See Section “Writing a Batch Browser”.

A sockets program can be written with various programming languages which offer the required library to achieve this function.

Authentication

HTTP basic authentication includes a userid and password. Interactive browsers include a basic authentication feature which automatically prompts for userid and password. In a batch browser, the authentication must be specifically coded. The userid and password are to be base64-encoded within the document header. Base64-encoding utilities are readily available on the Internet as either public domain software or commercial libraries.

Server Response

The receiving server will send a gisb-acknowledgement-receipt ([OPEN ISSUE 002](#)) as an HTTP response to the client before dropping the client's connection. If the transacting parties agree to use signed receipts, then the receiving server applies a digital signature to the gisb-acknowledgement-receipt ([OPEN ISSUE 002](#)) and encapsulates the entire package in a MIME envelope of Content-type: application/pgp-signature. The response returned from the Web server will contain timestamps that include a timestamp recorded when the final byte from the file upload is received and stored. This timestamp is the official timestamp regarding transaction turnaround deadlines defined in NAESB ~~WGQ-REQ~~ standards. This timestamp and all other pertinent file transmittal information should be logged when the posted file is stored on the receiving server as well as logged by the client. Likewise, any errors or warnings should be logged at both the server and client. ([OPEN ISSUE 010](#))

Throughput Considerations

The performance of the batch browser is one component critical in meeting deadlines. It is conceivable that it may be called many times for a busy site ~~(such as a pipeline sending quick responses)~~. It should therefore utilize whatever performance techniques ~~that~~ are possible. For example, it may be desirable to write a multithreaded version which can handle a certain number of requests simultaneously with a single copy of the program.

HTTP Request Data Elements

The HTTP Request will provide all required data elements in the order defined. Any mutually agreed to data elements will follow the required data elements in the data stream.

Required Data Elements (listed in the required order)

Data Element Name	Description
from	Common Code Identifier of sending/client company. (OPEN ISSUES 001)
to	Common Code Identifier of receiving/server company. (OPEN ISSUES 001)
version	The NAESB WGQ-REQ EDM version being used by the sender, in decimal notation (e.g. 1.64) The sending of the "version" data element is intended to assist in the early identification of EDM configuration errors and will not in itself dictate the version which a receiving party will support.
receipt-disposition-to	Common Code Identifier of the party to receive the acknowledgement receipt. (OPEN ISSUE 001)
receipt-report-type	Type of receipt requested "gisb-acknowledgement-receipt". (OPEN ISSUE 002)
input-format	Descriptor of the data format within the input data set.
input-data	The properly formatted file of electronic commerce data.

Mutually Agreed Upon Data Elements

Data Element Name	Description
-------------------	-------------

transaction-set	Descriptor of the transaction types included in the input-data. The values used must be from the unique 8-character names defined in the Implementation Standards. See the HTTP transaction-set Code Values table in the Hypertext Transport Protocol (HTTP) section Related Standards Tab for the various transaction types and their corresponding 8-character names. (OPEN ISSUE 003)
receipt-security-selection	Used to request signed receipts from the party receiving a file upload.

Writing a Batch Browser

A batch browser needs to simulate the actions of an interactive browser. As stated earlier, the interactive browser will take the HTML form and reformat the information according to the HTTP protocol before it sends the data stream to the HTTP server. The reformatting involves adding a header and placing field delimiters around the data items. A batch browser needs to produce the same kind of data stream and therefore, writing a batch browser requires some specific knowledge of the HTTP protocol. ~~See the NAESB WGQ home page for sources of HTTP protocol information.~~

First, consider the header:

Sample Example of a typical header sent to the HTTP server

```
POST /cgi-bin/AS2dispatcher HTTP/1.1
Referer: http://www.get.a.life/upl.htm
Connection: Keep-Alive
User-Agent: brow v0.1 XYZ Corp.
Host: localhost
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Content-type: multipart/form-data; boundary=-----87453838942833
Content-Length: 5379
```

This information is documentary in purpose. The parts that are important are:

The first line: *POST /cgi-bin/AS2dispatcher HTTP/1.1* indicating that the POST method is used and which program to call.

In the example above the content type line is:

Content-type: multipart/form-data; boundary=-----87453838942833

The content-type element indicates that the encoding method is multipart. It also identifies the character string used as the boundary. The boundary will appear between each field as a delimiter. **In this example**, the boundary is comprised of 27 hyphen characters followed by a number.

The boundary can be any character string that you choose except that it is required that it will not occur anywhere else in the form or in the transaction being sent. This is usually accomplished by using either the system clock or a random number so that even if by some remote chance the string appears in the document it would not appear in any re-transmission of the file. It is strongly recommended that a relatively long string be used as a boundary. The boundary when used as a separator requires two hyphen characters appended to the front of

the string as you can note by the lines between the data fields in the example. The last boundary required in the form is two hyphen characters appended to the back of the separator boundary, this is used to indicate to the server program that this is the end of the data.

In the example above, the content length is:

Content-Length: 5379

The content-length value should match the number of bytes contained in the entity body including the characters in the boundary lines, variable content, blank lines, etc. In essence, it tells the server how much is going to come after this point.

In this example, the data portion, or body, sent to the server program is as follows and assumes only required data elements are sent (not mutually agreed data elements):



```

-----87453838942833
Content-Disposition: form-data; name="from"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="to"

234567890 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="version"

1.64
-----87453838942833
Content-Disposition: form-data; name="receipt-disposition-to"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="receipt-report-type"

gisb-acknowledgement-receipt (OPEN ISSUE 002)
-----87453838942833
Content-Disposition: form-data; name="input-format"

x12
-----87453838942833
Content-Disposition: form-data; name="input-data"; filename="c:\temp\smallnom.bin"
Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-encrypted"

--8760
Content-Type: application/pgp-encrypted

Version: 1

--8760
Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----
Version: PGP 6.5

hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwYsHsz0e8
sb7ErC340MrNA/dw3taGMjml+CXRYF/PLEdg1NZE1ZCtNeL4YdIHAMLWwODGIQxhSuc
z8rMSqQ5mZzcOJwBdWLW70efgsu/9UJjuJYc1uZ6C03eFQv/43fkB+alATgydxX4g8QK6
64ad+Jo/XUICSmWBL66fqJR1KLeLf4wTaqGy174Aq48Wpwwg1Eh785zC03UAw0qg0ug
Mt86dPeyd91e2JigqwDYEf/DYEKD0J9BGiGpS/uAupNKj8Ocp2IWClxKOGUbxpVNOnt
qWHS/GntegvDE/7/ewCxDxsnmQS95pOI141QZ1RqbeNaqx2Dq/ra9g65HNchOCzjul5Vi8
HHf6Yhg2WnROe+npByyCue6rihqgNVOJwj0Cvzpb4JE+qMDf3q4ISUb1Fv7/+SSFHDdnh
dC5YTpqf1Bc3B07hiLmtTXqNit31EbX9UVEIObzSa9ZhxbC6/eSI7Nuf5ZTDsh9nrk+QQJ6
FeC9W4cqXLjIZySaRO8Vtff+4ktqeuH YusT4kSpnk027aw4O/5jomUkfb22CAe4=
=Oiuo
-----END PGP MESSAGE-----
--8760--
-----87453838942833--

```

The important characteristics of the above stream are:

- The boundary string appears at the beginning of each data field in the body.
- For each body data field, two identifiers define the contents of the data field. The Content-disposition identifier defines that “form-data” is contained in the element. The name identifier defines the name of the data element. These data element names must

match the name specified by NAESB ~~REQ~~WGQ. The name identifier is not completely relevant since the fields should be present in the correct order but this field should be checked to verify the validity of the form content.

- The actual data value of the field is always preceded by a line termination. This is typically used as a marker for the server program to indicate that a data value will follow. For example, note the blank line preceding "X12" in the above ~~ex~~sample. In most programming libraries and commercial products the starting delimiter is "\n\n\n" (C notation).
- The data field containing the NAESB ~~WGQ~~REQ standard file has two extra identifiers: first the name of the file sent from the source computer, filename="c:\temp\smallnom.bin", and second a content type identifier on a separate line. This line should always be constructed to reflect the content-type of the data being transmitted, in accordance with accepted Internet standards. If the data file contains clear text, X12 data, as shown in the above example, the content-type identifier follows the recommendations of RFC 1767, "MIME Encapsulation of EDI Data", and the "Content-Type:application/EDI-X12" is used. However, for security purposes it is recommended that all data be encrypted and digitally signed prior to transmission over the Internet. There are IETF standards for describing and packaging encrypted data files, most notably, "MIME Security with Pretty Good Privacy (PGP)", RFC 2015 and "MIME-based Secure EDI", RFC TBD ([OPEN ISSUE 005](#)).
- After the contents of the last data field, the boundary appears again as the last item of the form with the required two hyphen characters following the boundary at the end of the form to indicate the end of the data.

When the sender of a file intends to use encryption and digital signature functions to secure the contents of a data file the file must be prepared in accordance with the above mentioned IETF standards. ASC X12 data must first be prepared in canonical form as specified in RFC 1767. The ASC X12 data file would be concatenated with the MIME Content-type of application/EDI-X12 as the first line of the file.

For example below is a file before encryption:

```
Content-type: application/EDI-X12
ISA~00~ ~01~AAA6300300~14~1234567890000 ~14~2345678900000
... more data from the X12 file...
IEA~1~000003616
```

This file is encrypted, signed and packaged, which follows EDIINT AS1 and RFC 2015, which produces a file containing MIME headers and encrypted content as follows.

Below is the file after encryption:

Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-encrypted"

--8760

Content-Type: application/pgp-encrypted

Version: 1

--8760

Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----

Version: PGP 6.5

hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwYsHsz0e8sb7Er340MrNA/dw3taGMjml+CXYRF/PLEdg1NZE1ZCtNeL4YdlHAMLWwODGIQxhSucz8rMSgQ5mZzcOJwBdWLW70efgsu/9UljuJjYc1uZ6C03eFQv/43fkB+alATtgydxX4g8QK664ad+Jo/XUICSmWBL66fqJR1KLeLf4wTaqGy174Aq48Wpwvg1Eh785zC03UAW0qg0ugMt86dPeyd91e2JigqwDYEf/DYEKD0J9BGiGpS/uAupNKj8Ocp2IWCIXKOGUbxpVNOntqWHS/GntegvDE/7/ewCxDxsnmQS95pOI141QZ1RQbeNaqx2Dq/ra9g65HNchOCzjul5Vi8HHf6Yhg2WnROe+npByyCue6rihgqNVOJwj0cVzpb4JE+gMDf3q4ISub1Fv7/+SSFHDdnhdC5YTpqf1Bc3B07hiLmtTXqNit31EbX9.UVEIObzSa9ZhxbC6/eS17Nuf5ZTDsh9nrk+QQJ6FeC9W4cqXLj7IZySaRO8Vtff+4ktqeuHYusT4kSpnk027aw4O/5jomUkfb22CAe4=

=Oiuo

-----END PGP MESSAGE-----

--8760--

Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-encrypted"

--8760

Content-Type: application/pgp-encrypted

Version: 1

--8760

Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----

Version: PGP 6.5

hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwYsHsz0e8sb7Er340MrNA/dw3taGMjml+CXYRF/PLEdg1NZE1ZCtNeL4YdlHAMLWwODGIQxhSucz8rMSgQ5mZzcOJwBdWLW70efgsu/9UljuJjYc1uZ6C03eFQv/43fkB+alATtgydxX4g8QK664ad+Jo/XUICSmWBL66fqJR1KLeLf4wTaqGy174Aq48Wpwvg1Eh785zC03UAW0qg0ugMt86dPeyd91e2JigqwDYEf/DYEKD0J9BGiGpS/uAupNKj8Ocp2IWCIXKOGUbxpVNOntqWHS/GntegvDE/7/ewCxDxsnmQS95pOI141QZ1RQbeNaqx2Dq/ra9g65HNchOCzjul5Vi8HHf6Yhg2WnROe+npByyCue6rihgqNVOJwj0cVzpb4JE+gMDf3q4ISub1Fv7/+SSFHDdnhdC5YTpqf1Bc3B07hiLmtTXqNit31EbX9.UVEIObzSa9ZhxbC6/eS17Nuf5ZTDsh9nrk+QQJ6FeC9W4cqXLj7IZySaRO8Vtff+4ktqeuHYusT4kSpnk027aw4O/5jomUkfb22CAe4=

=Oiuo

-----END PGP MESSAGE-----

~~--8760--~~

This file is associated with the "input-data" data element of the multipart-form-data and is sent to the recipient using the HTTP POST method.

The HTTP POST data stream used to send this file would appear as follows:

```

-----87453838942833
Content-Disposition: form-data; name="from"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="to"

234567890 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="version"

1.46
-----87453838942833
Content-Disposition: form-data; name="receipt-disposition-to"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="receipt-report-type"

gisb-acknowledgement-receipt (OPEN ISSUE 002)
-----87453838942833
Content-Disposition: form-data; name="receipt-security-selection"

signed-receipt-protocol=required, pgp-signature; signed-receipt-micalg=required, md5
-----87453838942833
Content-Disposition: form-data; name="input-format"

X12
-----87453838942833
Content-Disposition: form-data; name="input-data"; filename="c:\temp\smallnom.bin"
Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-encrypted"

--8760
Content-Type: application/pgp-encrypted

Version: 1

--8760
Content-Type: application/octet-stream

-----BEGIN PGP MESSAGE-----
Version: PGP 6.5

hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwY
sHsz0e8sb7ErC340MrNA/dw3taGMjml+CXRYF/PLEdg1NZE1ZCtNeL4YdlHAML
    
```

WwODGIQxhSucz8rMSgQ5mZzcOJwBdWLW70efgsu/9UljUjYc1uZ6C03eFQv/4
3fkB+alATtgydxX4g8QK664ad+Jo/XUICSmWBL66fqJR1KLeLf4wTaqGy174Aq48
Wpwwg1Eh785zC03UAW0qg0ugMt86dPeyd91e2JigqwDYEf/DYEKD0J9BGiGpS/
uApNKj8Ocp2IWCIXKOGUbxpVNOntqWHS/GntegvDE/7/ewCxDxsnmQS95p
OI141QZ1RQbeNaqx2Dq/ra9g65HNchOCzjul5Vi8HHf6Yhg2WnROe+npByyCue6
rihqgNVOJwj0cVzpb4JE+gMDf3q4ISub1Fv7/+SSFHDdnhdC5YTpqf1Bc3B07hiL
mtTXqNit31EbX9UVEIObzSa9ZhxbC6/eSI7Nuf5ZTDsh9nrk+QQJ6FeC9W4cqXLj
7IZySaRO8Vfff+4ktqeuH Yust4kSpnk027aw4O/5jomUkfb22CAe4=
=Oiuo

-----END PGP MESSAGE-----

--8760--

-----87453838942833--



Although the specifications for multipart POST include several variations on this method, the NAESB [WGQ-REQ](#) standards do not include implementing them at this time ([OPEN ISSUE 006](#)). The most significant of these variations is to send several files in a single post. Additionally, sending a single file split into more than one post is not expected by the HTTP server.

The output from the browser is important to the understanding of the processing needed by the server script or program which must interpret the result. The complete data stream from the browser will look like:

```

POST /cgi-bin/AS2dispatcher HTTP/1.1
Referer: http://www.get.a.life/upl.htm
Connection: Keep-Alive
User-Agent: brow v0.1 XYZ Corp.
Host: localhost
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Content-type: multipart/form-data; boundary=-----87453838942833
Content-Length: 5379

-----87453838942833
Content-Disposition: form-data; name="from"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="to"

234567890 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="version"

1.46
-----87453838942833
Content-Disposition: form-data; name="receipt-disposition-to"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="receipt-report-type"

gisb-acknowledgement-receipt (OPEN ISSUE 002)
-----87453838942833
Content-Disposition: form-data; name="input-format"

X12
-----87453838942833
Content-Disposition: form-data; name="input-data"; filename="c:\temp\smallnom.bin"
Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-encrypted"

--8760
Content-Type: application/pgp-encrypted

Version: 1

--8760
Content-Type: application/octet-stream
    
```

-----BEGIN PGP MESSAGE-----

Version: PGP 6.5

hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwYsHsz0e8sb7ErC3
 40MrNA/dw3taGMjml+CXYRF/PLEdg1NZE1ZCtNeL4YdIHAMLWwODGIQxhSucz8rMSgQ5mZzcO
 JwBdWLW70efgsu/9UljuJjYc1uZ6C03eFQv/43fkB+alATtgydxX4g8QK664ad+Jo/XUICSmWBL66fqJ
 R1KLeLf4wTaqGy174Aq48Wpwvg1Eh785zC03UAw0qg0ugMt86dPeyd91e2JigqwDYEf/DYEKD0J9
 BGiGpS/uAupNKj8Ocp2IWClxKOGUbxpVNOntqWHS/GntegvDE/7/ewCxDxsnmQS95pOI141QZ
 1RqbeNaqx2Dq/ra9g65HNchOCzjul5Vi8HHf6Yhg2WnROe+npByyCue6rihqgNVOJwj0cVzpb4JE+g
 MDf3q4ISUb1Fv7/+SSFHDdnhdC5YTpqf1Bc3B07hiLmtTXqNit31EbX9UVEIObzSa9ZhxbC6/eSI7N
 uf5ZTDsh9nrk+QQJ6FeC9W4cqXLj7IZySaRO8Vtff+4ktqeuH YusT4kSpnk027aw4O/5jomUkfb22CA
 e4=

=Oiuo

-----END PGP MESSAGE-----

--8760--

-----87453838942833-----

Client Specifications

Each client should be synchronized to a clock in the network of atomic clocks that is accessible via the Internet. Central Time (Central Standard / Central Daylight) available at any of the sites on a synchronized network of atomic clocks. Each trading party should observe the client clock over a period of time to determine the amount of “drift” occurring throughout the day with respect to the atomic clock. The client should be synchronized as many times per day as necessary to ensure synchronization. The most important time period to ensure synchronization is just prior to the nomination deadline. Please refer to Appendix A, “Time Synchronization” for references on public sites for synchronization.

The HTTP Request will provide all required data elements in the order defined. Any mutually agreed to data elements will follow the required data elements in the data stream.

RECEIVING TRANSACTIONS

General Flow

The following is an example of the steps necessary to receive an Internet EDI/EDM file: and batch FF/EDM file:

1. Parse multi-part form
2. Validate HTTP request data elements
3. If HTTP request data elements in error, return appropriate standard error code in the HTTP response data elements
4. Save data
5. Create gisb acknowledgement receipt (OPEN ISSUE 002)
- 5.1 If using signed receipts:
 - 5.1.1 Produce a digital signature over the gisb acknowledgement receipt (OPEN ISSUE 002) created in step 5.
 - 5.1.2 Encapsulate the gisb acknowledgement receipt (OPEN ISSUE 002) and Digital Signature body parts in a content-type of application/multipart/signed envelope
6. Return HTTP response, the gisb acknowledgement receipt (OPEN ISSUE 002) object, back to server

7. Close connection
8. Log final results
9. Route data file to the next process based upon input format

Using a Web Server

As was stated above, the protocol HTTP using the POST method as the means to upload a transaction is the standard. (~~OPEN ISSUE 004~~). On the receiving side of this HTTP request is the Web server, the second primary component in Web technology. However, the Web server does not actually save the uploaded file. Instead, it hands this responsibility over to a special program which, in effect, extends the Web server's functionality with custom programming. ~~This special program is known as a Common Gateway Interface (CGI) program.~~ Besides storing the file, ~~the CGI~~this program has the task of parsing the incoming HTTP message, noting the date, time and time zone indicator so to create the timestamp, and creating an HTML response to the ~~sendering browser.~~ (~~OPEN ISSUE 010~~)

An EDM receiving program may be implemented using a variety of technologies and techniques. Some options include, Active Server Pages (ASP), Common Gateway Interface (CGI), Java Server Pages (JSP), Java Servlet technology and other technologies.

The NAESB ~~WGQ-REQ~~ standard places no particular requirements on the vendor for the Web server. Most commercially available Web servers will provide the needed functionality. However, please refer to comments regarding performance under "Throughput Considerations" later in this section. ~~While the current approach to security does not require a Secure Sockets Layer (SSL) or Secure Hyper Text Transfer Protocol (S-HTTP) capable server, one of these may be a requirement in the future.~~ Determine whether the product you are considering provides a secure version capable of either SSL or S-HTTP. (~~Unfortunately, it is too early to predict which of these, if either, will prevail as an emerging standard.~~)

~~Another capability you may wish to consider when choosing a Web server is whether it supports Binary Gateway Interface (BGI) capability. Specifically, this is the capability to run Dynamic Link Library (DLL) equivalents of CGI applications. Some vendors call this capability Internet Server Application Programming Interface (ISAPI) while others call it Netscape Application Programming Interface (NSAPI).~~

The CGI Receive Process

A CGI (or BGI) receiving program must be able to parse the multipart form. It accomplishes this by finding the boundary string in the Content-Type header and scanning for its occurrences further within the uploaded stream. Upon finding these boundary strings, the program must next determine the content-disposition for each data element. This allows detection of the required text elements as well as the NAESB ~~WGQ-REQ~~ standard format file.

The CGI receiving program is not concerned with the content of the NAESB ~~WGQ-REQ~~ standard format data. In fact, the standard format file will be encrypted (see the Security section). The CGI program will merely accept the standard format data and store it as a file. ~~The CGI~~It will use the Content-Length to determine how much data to expect in the body.

Throughput Considerations

It is critical that the Web server and the associated CGI receiving programs perform efficiently.

This is particularly true for ~~pipelines high volume sites which may expect to see a large number of nomination transactions come in close to the deadline processing time sensitive materials.~~ For the greatest possible throughput, the Web server should be multithreaded. The ~~CGI receiving~~ program should be multithreaded as well or be ~~as~~ small and efficient as ~~is~~ possible ~~with a C program.~~ ~~BGI programming may provide even better performance.~~ It is also suggested that a Web server and operating system be chosen that allow for scaling to a more powerful computer (possibly multi-CPU). Transaction volumes are likely to be light at first but may become heavy rather quickly.

Writing the ~~CGI~~ Developing the Receive Process

A ~~CGI receive~~ process ~~is requires the an~~ executable program or module that is called by the HTTP server when it is identified by a POST ~~or GET~~ operation. ~~(In this case we are only concerned with POST method operations.)~~

When the HTTP server receives a POST it will first read the header and populate environment variables before calling the ~~CGI receiving~~ program. A sample header is shown below.

```
POST /cgi-bin/AS2dispatcher HTTP/1.1
Referer: http://www.get.a.life/upl.htm
Connection: Keep-Alive
User-Agent: brow v0.1 XYZ Corp.
Host: localhost
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Content-type: multipart/form-data; boundary=-----87453838942833
Content-Length: 5379
```

The important point to note is that you will not specifically code the step of reading the header and populating the environment variables, the HTTP server performs it for you. The variables populated are usually listed with the HTTP server documentation.

After reading this header the server will buffer the remaining data transmitted and then call the ~~CGI receiving program process~~ specified in the POST statement. Do not assume that the ~~CGI process receiving program~~ is called as soon as the header is read. The more common implementations will buffer the entire transmission before calling the ~~program CGI~~. You may want to check your server implementation if this characteristic is important to you.

The ~~called CGI process receiving program~~ will have the following ~~input~~ stream available ~~in the standard input (stdin)~~, and ~~will have~~ most of the header data available in environment variables.

```
-----87453838942833
Content-Disposition: form-data; name="from"
123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="to"
234567890 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="version"
1.64
-----87453838942833
Content-Disposition: form-data; name="receipt-disposition-to"
123456789 (OPEN ISSUE 001)
```

```

-----87453838942833
Content-Disposition: form-data; name="receipt-report-type"
gisb-acknowledgement-receipt(OPEN ISSUE 002)
-----87453838942833
Content-Disposition: form-data; name="input-format"
X12
-----87453838942833
Content-Disposition: form-data; name="input-data"; filename="c:\temp\smallnom.bin"
Content-Type: multipart/encrypted; boundary=8760; protocol="application/pgp-encrypted"
--8760
Content-Type: application/pgp-encrypted
Version: 1
--8760
Content-Type: application/octet-stream
-----BEGIN PGP MESSAGE-----
Version: PGP 6.5
hQCMAzRG1pEOIOvdAQP+JMr0m/9+8yOL60Z9Vr6fFV81FCExB/o0xmwiMkiwYsHsz0e8sb7ErC340MrNA/d
w3taGMjml+CXYRF/PLEdg1NZE1ZCtNeL4YdIHAMLWwODGIQxhSucz8rMSgQ5mZzcOJwBdWLV70efgsu/
9UljYc1uZ6C03eFQv/43fkB+aIATgydxX4g8QK664ad+Jo/XUICSmWBL66fqJR1KLeL4wTaqGy174Aq48
Wpwvg1Eh785zC03UAW0qg0ugMt86dPeyd91e2JigqwDYEf/DYEKD0J9BGiGpS/uAupNKj8Ocp2IWCixKOG
UbxpVNOntqWHS/GntegyDE/7/ewCxDxsnmQS95pOI141QZ1RQbeN.aqx2Dq/ra9g65HNchOCzjul5Vi8HH
6Yhg2WnROe+npByyCue6rihqgNVOJwj0cVzpb4JE+gMDf3q4ISUb1Fv7/+SSFHDdnhdC5YTpqf1Bc3B07hiL
mtTXqNit31EbX9UVEIObzSa9ZhxbC6/eSI7Nuf5ZTDsh9nrk+QQJ6FeC9W4cqXLj7IZySaRO8Vtff+4ktqeuYU
sT4kSpnk027aw4O/5jomUkfb22CAe4=
=Oiuo
-----END PGP MESSAGE-----
--8760--
-----87453838942833-----

```

This process should check for basic validity in the environment variables and the data stream. It will parse the variables/data from the format. The data validations should include:

- The “REQUEST_METHOD” environment variable is “POST”.
- The “CONTENT_TYPE” environment variable should be “multipart/form-data” and a boundary, which is unique in that it cannot appear anywhere in the transaction being sent (see above stream for an example).

The input stream should ~~be support in~~ binary mode to accommodate encrypted files.

- Each data element ~~is should~~ be preceded by the boundary with the required two hyphen characters appearing before it.
- Each data element should contain the correct name on the *Content-Disposition* line.
- Each data element should have \r\n\r\n (~~Ce~~ notation) before the start of the data.
- In the receiving program, all tag values in the HTTP header should be evaluated in a case insensitive manner.

Finding the end of the stream using both content length and the boundary end mark (the boundary with two required hyphen characters in front and behind) is usually the best method to detect improperly formatted input.

Immediately after the ~~CGI-receiving program~~ validates (as above), parses, and saves the data, ~~the CGI~~ should record the time and construct a gisb acknowledgement receipt (~~OPEN ISSUE 002~~) described in the following section. This ~~gisb acknowledgement~~ receipt is ~~usually~~ sent from the ~~CGI-receiving program by writing to the standard output (stdout) of the CGI processto the sending program~~. If using signed receipts, the receiving party must produce a digital signature of the gisb acknowledgement receipt (~~OPEN ISSUE 002~~) and send both the gisb acknowledgement receipt (~~OPEN ISSUE 002~~) and digital signature body parts within a multipart/signed MIME envelope.

Receive Process URL/CGI Implementation Guidelines

NAESB ~~WGQ-REQ~~ standard 4.3.12 (~~OPEN ISSUE 007~~) states

"As a minimum, with a trading partner agreement, one designated site for receipt should be identified for each trading partner. That site should be identified by a specific Uniform Resource Locator (URL). This does not preclude multiple designated sites being mutually agreed to between trading partners." (~~OPEN ISSUE 008~~)

~~This standard specifies that e~~ach company must offer at least one URL (~~URL is a one-to-one association with CGI~~) to accept ~~EDI/EDM and FF/EDM~~ files using Internet EDM. However, a maximum number of URLs per company is *not* included so that companies that wish to offer additional URLs will not be held back from doing so. Though companies are free to construct an ~~EDI/EDM and FF/EDM~~ Web site with multiple "single-purpose" URLs, NAESB ~~WGQ-REQ~~ recommends the use of one "general-purpose" URL.

Error notifications include errors that occur some time after the gisb acknowledgement receipt (~~OPEN ISSUE 002~~) is sent (such as a file decryption error) as well as errors on the transactions. A general-purpose URL would handle all error notifications.

Companies that wish to offer multiple URLs must negotiate additional URLs with their trading partners. All URLs that will be required for use in the ~~EDI/EDM and FF/EDM~~Internet EDM process must be agreed to and defined in the Trading Partner Agreement (TPA) (~~OPEN ISSUE 008~~) signed by both companies. ~~An~~ For example, a separate URL may be used for customized processing, such as high or low priority transactions. ~~of a company that would define multiple URLs in the TPA is a company that comes to agreement with its partners that all nominations-related transactions are sent to a URL offered by an out-sourcing vendor. All other transactions are sent to a URL offered on its own Web server.~~

~~A company can also offer additional URLs which have a special purpose without defining the URL in a TPA. Such additional URLs would be a way of offering additional customer service. The trading partners would have the option of using the additional URL. An example of a company that offers a URL for additional customer service is a company that offers a URL to accept capacity release information requests with immediate turnaround while the general-purpose URL is set up to postpone all capacity release information requests until 4 p.m. that day. This company wishes to keep its primary Web server available for nominations requests while other information requests are handled on a secondary Web server.~~

To those companies who wish to offer multiple URLs, NAESB ~~WGQ-REQ~~ strongly recommends that you divide URL usage along transactional grouping lines, ~~such as nominations or capacity release~~. Create groupings that are likely to correlate to business functions in a company, ~~within the gas industry~~. Do not divide URL usage along an arbitrary internally-understood group such as region of the country. Remember that the intent of not specifying a maximum number of

URLs is to allow companies the freedom to offer services, not to further complicate the ~~EDI/EDM and FF/ Internet~~ EDM process.

~~Some companies have raised a question of offering a “default” URL. The default URL would be used when the trading partner was not able to determine the proper URL from the trading partner agreement. NAESB WGQ does not recommend that any company offer a default URL. When situations arise where the TPA does not fully define the appropriate URL, the partners should communicate the situation, agree to the appropriate URL usage, and revise the TPA.~~

Server Specifications

The HTTP server should be synchronized to ~~Central Time (Central Standard / Central Daylight)~~a clock in the network of atomic clocks that is accessible by the Internet. ~~available at any of the sites on a synchronized network of atomic clocks.~~ Each trading party should observe the ~~server~~ clock over a period of time to determine the amount of “drift” occurring throughout the day. The server should be synchronized as many times per day as necessary to ensure synchronization. ~~The most important time period to ensure synchronization is just prior to the nomination deadline.~~ Please refer to Appendix A, “Time Synchronization” for references on public sites for synchronization. (R)

The HTTP server will provide an HTTP response to the client according to NAESB ~~WGQ~~REQ standards.

All data element names of the HTTP request and response fields will be in lower case. Note that the NAESB ~~WGQ~~REQ standard format file contained in the request and response may follow a different standard.

Carriage returns and line feeds will be ignored in all files.

A field delimiter of “*” will be used in the HTTP response. Please refrain from displaying a “*” anywhere else in the response so as not to confuse programs that need to parse on this basis.

No spaces should surround the equal sign or the field delimiter.

The required data elements must appear first in the response.

Additional information can be included after the required elements at the server’s discretion.

The gisb acknowledgement receipt ([OPEN ISSUE 002](#)) must be enveloped in a multipart/report, as specified in EDIINT AS2 ([OPEN ISSUE 005](#)) following the rules for Generalized Receipts. If signed receipts are used, the gisb acknowledgement receipt ([OPEN ISSUE 002](#)) (including the multipart/report envelope) is digitally signed, producing a application/pgp-encrypted body part. Both the multipart/report (gisb acknowledgement receipt([OPEN ISSUE 002](#))) and the application/pgp-signature body parts are placed in a multipart/signed envelope and the entire package is returned to the sender.

The first occurrence of the field name within the response will contain the value.

If an HTML response is given, all data must be presented in a user-readable fashion. For example, if the required machine-readable fields are embedded in comments, another representation of these fields must be presented to the user.

Servers should be configured to use one of the allowable TCP ports listed in Appendix E([OPEN ISSUE 009](#)).



HTTP Response Data Elements

Required Data Elements (listed in the required order)

Data Element Name	Description
time-c	the time of transfer completion at the server. The format will be <i>yyyymmddhhmmss-<u>ZZ</u></i> . The <u>-ZZ</u> indicates the difference, in hours, between local time and Coordinated Universal Time (UTC). For example, 20030107142004-06, reflects a timestamp that was issued by a machine set to the Central time zone. In cases where a machine is within a time zone that is ahead of UTC the (-) would change to a (+). (OPEN ISSUE 010)
request-status	a text status indicator by the server. The only defined value at this time is "ok" for a successful transfer. The server should supply a descriptive indication of the error detected following the standards for error codes and messages presented in Table A, "Internet EDM Standard Error Codes and Messages".
server-id	a <i>domainname</i> or <i>hostname.domainname</i> uniquely identifying the server associated with the CGI program that received and processed the file.
trans-id	a number (integer) up to 15 characters in length uniquely identifying the received transaction file at the server. The trans-id will uniquely identify the file only at the receiving server. A client may receive non-unique trans-ids across multiple servers.

[Samples-Examples](#) of HTTP Response Required Data Elements:

successful, [plain-text/multipart](#) format:

```
Content-Type: multipart/report; report-type="gisb-acknowledgement-receipt"; boundary="GISB7867" \(OPEN ISSUE 002\)

--GISB7867
Content-type: text/html

<HTML><HEAD><TITLE>Acknowledgement Receipt Success</TITLE></HEAD> <BODY><P>
time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=ok*
server-id=coolhost*
trans-id=234423897*
</P> </BODY></HTML>
--GISB7867
Content-type: text/plain

time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=ok*
server-id=coolhost*
trans-id=234423897*
--GISB7867--
```

or

error, [plain-text/multipart](#) format:

```
Content-Type: multipart/report; report-type="gisb-acknowledgement-receipt"; boundary="GISB7866" \(OPEN ISSUE 002\)

--GISB7866
Content-type: text/html
```

```
<HTML><HEAD><TITLE>Acknowledgement Receipt Error</TITLE></HEAD> <BODY><P>
time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=EEDM106: Invalid To Common Code Identifier*
server-id=coolhost*
trans-id=234423897*
</P> </BODY></HTML>
--GISB7866
Content-type: text/plain

time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=EEDM106: Invalid To Common Code Identifier*
server-id=coolhost*
trans-id=234423897*
--GISB7866--
```

or

warning, plain text/multipart format:

```
Content-Type: multipart/report; report-type="gisb-acknowledgement-receipt"; boundary="GISB7866" \(OPEN ISSUE 002\)

--GISB7866
Content-type: text/html

<HTML><HEAD><TITLE>Acknowledgement Receipt Warning</TITLE></HEAD> <BODY><P>
time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=WEDM100: Transaction Set Sent, Not Mutually Agreed*
server-id=coolhost*
trans-id=234423897*
</P> </BODY></HTML>
--GISB7866
Content-type: text/plain

time-c=19960619082855* \(OPEN ISSUE 010\)
request-status= WEDM100: Transaction Set Sent, Not Mutually Agreed *
server-id=coolhost*
trans-id=234423897*
--GISB7866--
```

or, as a more elaborate response to a successful transmittal,

```
Signed Receipt
Content-Type:multipart/signed; micalg=pgp-md5; protocol="application/pgp-signature";
boundary=8760

--8760

Content-Type: multipart/report; report-type="gisb-acknowledgement-receipt"; boundary="GISBB7867"
\(OPEN ISSUE 002\)

--GISB7867
Content-type: text/html

<HTML><HEAD><TITLE>Acknowledgement Receipt Success</TITLE></HEAD> <BODY><P>

time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=ok*
server-id=coolhost*
```

```

trans-id=234423897*

</P> </BODY></HTML>

--GISB7867
Content-type: text/plain.
time-c=19960619082855* \(OPEN ISSUE 010\)
request-status=ok*
server-id=coolhost*
trans-id=234423897*
--GISB7867--
--8760
Content-Type: application/pgp-signature

-----BEGIN PGP MESSAGE-----

Version: 2.6.26.5

iQCVAwUBMJrRF2N9oWBghPDJAQE9UQQAtI7LuRVndBjrK4EqYBIb3h5QXIX/LC//JV5bNvkZIGPcEmI5iF
d9boEgvpHtIREEqLQRkYNoBActFBZmh9GC3C041WGquMbrxc+nls1TIKIA08rVi9ig/2Yh7LFrK5Ein57U/
W72vgSxLhe/zhdfoIT9BrnHOxEa44b+EI=
=ndaj

-----END PGP MESSAGE-----

--8760—
    
```

HTML format (this example is for a successful transmittal):

```

HTML format (this example is for a successful transmittal): <html> <head> <title>
Upload OK</ title></ head> <!-- time- c= 19960123203618*-->_ <!-- request-
status= ok* --> <!-- server- id= coolhost*--> <!-- trans- id= 232323897*--> <h1>
Upload OK </ h1>< br> <body> <B> File Saved at (time- c): </ B>
19960123203618< br> <B> Status (request- status): </ B> ok< br> <B> Server
(server- id): </ B>coolhost< br> <B> Transaction ID (trans- id): </ B>
232323897< br> </ body> </ html> \(OPEN ISSUE 010\)
    
```

Using a Service Provider for Web Hosting

If you do not wish to install and maintain a Web server, you may wish to contact an Internet Service Provider (ISP) to provide the hosting service for you. Consider the following when selecting an ISP for Web hosting:

- limit on storage space for receiving files
- ability to meet NAESB [WGQ-REQ](#) standards for HTTP response
- accommodation for [GGI-receiving program](#) to meet NAESB [WGQ-REQ](#) standards for validation and processing.

SECURITY

Security Concepts

The security requirements include the current four primary security aspects: data privacy, data integrity, authentication, and non-repudiation.

- Data privacy: unauthorized parties cannot decipher the content of the data.
- Data integrity: unauthorized parties cannot modify or corrupt the data.
- Authentication: the receiver is certain of the identity of the sender.
- Non-repudiation: the sender cannot deny ownership of the transaction if it was sent with his/her digital signature.

In general, these needs are met by using the Basic Authentication capability of the Web server and the encryption and digital signature capability of the PGP and OpenPGP security application for securing transactions.

Understanding PGP or OpenPGP

Pretty Good Privacy (PGP) is the name of the chosen security application. OpenPGP is the Internet Engineering Task Force standard version of PGP which excludes all patented algorithms, allowing free commercial use of the standard. See the NAESB [WGQ-home pageWeb site](#) for information on software packages to implement the PGP or OpenPGP security application. Both OpenPGP and PGP utilizes a public key/private key pair to accomplish secure file transfers. The private key must be known only to the company which generated it. The public key counterpart is shared with trading partners.

Each company must generate its public key and private key pair. The RSA key generation algorithm should be chosen for versions of PGP which offer alternatives. Implementers of OpenPGP should choose DSA and El Gamal when creating their key pair. The public keys will be distributed using a secure method (e.g., courier mail) to the company's trading partners. You must use the utmost care in protecting your private key. If it is compromised, the security is broken. It is recommended that a key size of 1024 be chosen when generating the key pair. This provides a significantly secure transaction.

When a company wishes to send transactions to its trading partner, it will use the partner's public key to encrypt the file. Encryption provides data privacy. Only the private key counterpart can decrypt this file. Hence, the need to guard your private key.

When the sending party encrypts the file, it also uses its own private key to "sign" the transaction. The receiving party can use the sender's public key to verify the signature. The digital signature provides non-repudiation.

Encryption / Digital Signature

Encryption and signatures are applied to files already translated to a NAESB [WGQ-REQ](#) standard data format, and before the data is sent to the batch browser. (Use of internal encryption such as X12.58 encryption is outside the scope of NAESB [WGQ-REQ](#) encryption)

standards but does not conflict with PGP.)-

Encryption and signatures can be accomplished manually for each file using the on-line PGP or, on a mutually agreed basis, OpenPGP software, or in an automated (or “batch”) fashion using programs to encrypt and sign. Whether encrypting in a manual or automated fashion, it is essential that the correct public key of the trading partner be used to encrypt and just as essential that the correct sender’s own private key be used to digitally sign the file.

Digital signatures may also be applied, on a mutually agreeable basis, to the HTTP response by the receiver of the transaction.

Decryption / Signature Verification

After a transaction is received and processed by the ~~CGI~~receiving program, it is ready to be decrypted and have its signature verified. PGP and OpenPGP software will utilize the appropriate key pair when encrypting, signing, and decrypting if given the correct userID in the key ring identifying the trading partner. Upon request for signature verification, the PGP and OpenPGP software will return a human-readable ~~company name~~descriptive text. (OPEN ISSUE 011)

It is recommended that all implementers create a process where the ~~name~~descriptive text is used to look up the ID of the ~~company~~trading partner in a database table. If the ID is passed along with the decrypted file, a process could be created to verify that the ~~company~~trading partner which sent the transaction corresponds to the ~~company~~trading partner identified within the file, once the data has been translated. (OPEN ISSUE 011)

When digital signatures are applied, on a mutually agreeable basis, the HTTP response received by the sender of the transaction may be verified to ensure non-repudiation of receipt of the transaction.

Throughput Considerations

Encryption, digital signing, decryption and signature verification are all very CPU intensive. It is not recommended that decryption or signature verification be performed within the ~~CGI~~program that receives and processes the file. In fact, it would not be a good idea to have these steps performed on the same computer that is attempting to receive transactions at a time close to a deadline. Therefore, it is recommended that the secured or to-be-secured transaction be passed to a separate computer for security processing. This “passing” would ~~likely~~ be accomplished by using the File Transfer Protocol (FTP) or other similar process. The security processing computer should be optimized for CPU and memory.

Implementers of Internet EDM sites should review and evaluate Domain Name Server (DNS) cache refresh intervals so as to ensure trading partner address changes are recognized on a timely basis. A refresh interval of 24 hours or less is common.

Because decryption and signature verification are not handled at the time the file is received, the sender will get an HTTP response of successful transfer but doesn’t know if the file can be decrypted by the receiver. Guidelines for communicating the status of the decryption step have been developed. See Section “Sending Error Notification Transactions” and Table A, “Internet EDM Standard Error Codes and Messages”.

Security Requirements

Basic Authentication

Basic authentication, also known as realm one security, has been defined as one of the security standards for transmission on the Internet. The userid and password will be assigned by the server party according to site standards. The ~~Trading party-Partner a~~Agreement must identify the userid and password for this security as well as procedures for changing the password, if applicable. (OPEN ISSUE 008 and OPEN ISSUE 012)

PGP or OpenPGP File Encryption

File encryption of the EDI file is also selected as a security standard for transmission on the Internet. The encryption software employed is required to be compatible with PGP ~~2-66.5~~ or greater (using keys generated with the RSA algorithm) or the OpenPGP standard, specified in IETF RFC 2440, on a mutually agreed basis. There are freely available software implementations of the OpenPGP standard available at <http://www.gnupg.org/>.

General Security Recommendations

Firewall

A firewall is one or more computers running special software which is designed to provide control of communications between two networks. Its purpose is to limit the types of services between these two networks. Often, a company's connection to the Internet is intended to provide several other services to its employees who are connected by an internal network such as a Local Area Network or Wide Area Network (LAN or WAN). Examples of these services include access to the World Wide Web, use of e-mail, use of file transfer capabilities and publishing content intended for viewing by the external world on a Web server. In addition, the internal network will likely have connections to host computers which provide internal services such as file and print sharing, fax and database capabilities. So that availability of these services and confidential internal data are not compromised by unwelcome intruders from the Internet, there should exist a protective mechanism between the internal network and the public Internet, the firewall.

There are two general mechanisms employed by firewalls to provide this control: packet filtering and proxy services. Packet filtering examines important components of the messages such as the address of the sending and target computers and the designator (port number) for a specific application running on the target computer. By doing this, it can prevent access to specific computers or programs on those computers. It can also reject messages from certain computers. Proxy servers have various capabilities. They can act as relay agents that can examine attempted use of certain features within an application thus limiting access to these features. They can also hide (by substituting its own address) the internal addresses of clients communicating with external hosts. This hiding makes it difficult for potential attackers to focus on specific internal hosts.

Because firewalls are designed to deal with a broad set of security issues, which may vary at each organization, and are not specific to the use of HTTP, this guide does not attempt to provide specific implementation information. Deciding on a specific firewall architecture, organizational security policies, and choosing between numerous products may require outside

resources to address these issues.

SENDING ERROR NOTIFICATION TRANSACTIONS

Error Notification

When a client sends a file to a server, the server responds to the receipt of the file. Though the file may be received correctly, some further processing must be done, such as decryption and X12 translation. The decryption step which will have a pass/fail status and then the X12 general translation step which will have a pass/fail status. The X12 general translation is merely the check that the file meets the X12 standards and has not been corrupted. Further translation and processing of specific transactions and elements is outside the Internet EDM scope.

When a file passes the decryption step ~~and passes the general translation step~~, no notifying communication is sent back to the client. However, if ~~either the decryption step or the general translation step~~ fails, an error notification must be sent to the client.

In general, this standard format for error notification applies to the posting of an error message after sender's session has been disconnected. This error notification has the potential of occurring only after the original HTTP Response is returned with an "ok" or a warning (WEDM999 format) for the request-status value, not an error (EEDM999).

Additionally, trading partners are permitted to utilize digitally signed error notifications, if both parties mutually agree to do so.

Error Notification Data Elements

The data elements for the error notification are the same as those described in Section "Sending Transactions", with the exception of the "input-format" and "input-data" elements. The file containing the data elements for error notification should not be encrypted.

Required Data Elements for Error Notification (listed in the required order)

Data Element Name	Description
from	Common Code Identifier of sending/client company, the server company which detected the error (OPEN ISSUE 001)
to	Common Code Identifier of receiving/server company, the client company which sent the data set in error (OPEN ISSUE 001)
input-format	"error"
input-data	<p>A text block containing the following items:</p> <ul style="list-style-type: none"> orig-from The "from" value from the original transmission orig-to The "to" value from the original transmission. orig-input-format The "input-format" value from the original transmission. resp-time-c The "time-c" value from the original response. resp-server-id The "server-id" value from the original response. resp-trans-id The "trans-id" value from the original response. request-status The new status of the transaction based on some process beyond CGI the receiving process such as decryption; see Table A, "Internet EDM Standard Error Codes and Messages". comments Any comments the original receiving server wishes to include.

Mutually Agreed Upon Data Elements for Error Notification

none defined at this time

Error Notification "input-data" Element Specifications:

The file containing the data elements for error notification should not be encrypted.

All data element names will be in lower case in the Error Notification.

Carriage returns and line feeds will be ignored in all files.

A field delimiter of "*" will be used in the Error Notification. Please refrain from displaying a "*" anywhere else in the error notification so as not to confuse programs that need to parse on this basis.

No spaces should surround the equal sign or the field delimiter.

The required data elements must appear first in the response.

Additional information can be included after the required elements at the server's discretion.

The first occurrence of the field name within the response will contain the value.

If an error notification is given, a NAESB WGQ-REQ Error Notification contains two body parts nested within a multipart/report outer envelope, containing a type parameter with the value "gisb-error-notification". (OPEN ISSUE 014) The first body part contains human readable content in HTML. The second body part contains machine readable content in HTML plain text. Additionally, consenting trading partners can mutually agree to digitally sign error notifications. If digital signatures are used, the multipart/report containing the NAESB WGQ-REQ Error Notification is used to create a digital signature body part, identified by a content-type of application/pgp-signature. Both the multipart/report NAESB WGQ-REQ Error Notification and application/pgp-encrypted digital signature body parts are combined in a multipart/signed envelope.

Error Notification Example:

```
POST /cgi-bin/AS2dispatcher HTTP/1.1
Referer: http://www.get.a.life/upl.htm
Connection: Keep-Alive
User-Agent: brow v0.1 XYZ Corp.
Host: localhost
Accept: image/gif, image/x-xbitmap, image/jpeg, image/pjpeg, */*
Content-type: multipart/form-data; boundary=-----87453838942833
Content-Length: 1958
-----87453838942833
Content-Disposition: form-data; name="from"

234567890 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="to"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="version"
```

1.46

-----87453838942833
Content-Disposition: form-data; name="receipt-disposition-to"

123456789 (OPEN ISSUE 001)
-----87453838942833
Content-Disposition: form-data; name="receipt-report-type"

gisb-acknowledgement-receipt (OPEN ISSUE 002)
-----87453838942833
Content-Disposition: form-data; name="input-format"

error
-----87453838942833
Content-Disposition: form-data; name="input-data"; filename="c:\temp\error.not"
Content-Type: multipart/report; report-type="gisb-error-notification"; boundary="GISB7868" (OPEN ISSUE 014)

--GISB7868
Content-type: text/html

```
<HTML><HEAD><TITLE>Error Notification</TITLE></HEAD> <BODY><P>  
orig-from=123456789* (OPEN ISSUE 001)  
orig-to=234567890* (OPEN ISSUE 001)  
orig-input-format=X12*  
resp-time-c=19960619102855* (OPEN ISSUE 010)  
resp-server-id=coolhost*  
resp-trans-id=234423897*  
request-status=EEDM601: Public Key Invalid*  
comments=Please contact 1-800-555-1212 for correct public key*  
</P> </BODY></HTML>
```

--GISB7868
Content-Type: text/plain

```
orig-from=123456789* (OPEN ISSUE 001)  
orig-to=234567890* (OPEN ISSUE 001)  
orig-input-format=X12*  
resp-time-c=19960619102855* (OPEN ISSUE 010)  
resp-server-id=coolhost*  
resp-trans-id=234423897*  
request-status=EEDM601: Public Key Invalid*  
comments=Please contact 1-800-555-1212 for correct public key*  
--GISB7868--
```

-----87453838942833---

Signed Error Notification

Content-Type:multipart/signed; micalg=pgp-md5; protocol="application/pgp-signature";
boundary=8760

--8760

Content-Type: multipart/report; report-type="gisb-error-notification"; boundary="GISB7868" (OPEN ISSUE 014)

--GISB7868
Content-type: text/html

```
<HTML><HEAD><TITLE>Error Notification</TITLE></HEAD> <BODY><P>  
orig-from=123456789* (OPEN ISSUE 001)  
orig-to=234567890* (OPEN ISSUE 001)
```

```

orig-input-format=X12*
resp-time-c=19960619102855* (OPEN ISSUE 010)
resp-server-id=coolhost*
resp-trans-id=234423897*
request-status=EEDM601: Public Key Invalid*
comments=Please contact 1-800-555-1212 for correct public key*

</P> </BODY></HTML>

--GISB7868
Content-Type: text/plain

orig-from=123456789* (OPEN ISSUE 001)
orig-to=234567890* (OPEN ISSUE 001)
orig-input-format=X12*
resp-time-c=19960619102855* (OPEN ISSUE 010)
resp-server-id=coolhost*
resp-trans-id=234423897*
request-status=EEDM601: Public Key Invalid*
comments=Please contact 1-800-555-1212 for correct public key*

--GISB7868--
--8760

Content-Type: application/pgp-signature
-----BEGIN PGP MESSAGE-----

Version: 2.6.26.5

iQCVAwUBMJrRF2N9oWBghPDJAQE9UQQA7LuRVndBjrk4EqYBib3h5QXIX/
LC//JV5bNvkZIGP1cEmI5iFd9boEgypirHtlIREEqLQRkYNoBActFBZmh9GC3C041
WGquMbrbxc+nls1TIKIA08rVi9ig/2Yh7LFrK5Ein57U/W72vgSxLhe/zhdfoIT9BrnH
OxEa44b+EI=
=ndaj

-----END PGP MESSAGE-----

--8760--

```

Pre-validation before Decryption

Proper trapping of the range of decryption process errors listed in Table A (Internet EDM Standard Error Messages and Codes) may require program code which is external to the decryption algorithm. Some versions of the PGP software do not explicitly discriminate between EEDM601, EEDM602, EEDM603, and EEDM699 type errors. Under such a circumstance, files inbound to the decryption process should be preprocessed to trap the errors not identified by the PGP version being used. For example, searching the file for the text strings “BEGIN PGP MESSAGE” and “END PGP MESSAGE” can quickly identify “EEDM602 File not encrypted” and “EEDM603 Encrypted file truncated” type errors when the implemented PGP version only identifies decryption success, invalid public key (EEDM601), and decryption failure (EEDM699).

CHECKLIST OF TESTING STEPS (OPEN ISSUE 015)

Purpose

Preliminary steps in testing are helpful before the full batch browser and server applications are

completed. This checklist is intended to provide a series of small achievements leading up to the complete solution.

Client/Browser

NOTE: Throughout all transfer tests, compare files stored on the server against the source file to ensure that the file transferred intact. While transferring to another company's server, you may have to contact that company to send the file back to you so that you can perform the compare.

1. Install an interactive browser. Identify an existing Web server from among NAESB WGQ compliant servers offering interactive upload for test. See the NAESB WGQ home page for a list of organizations willing to act as testing partners. These organizations should have a URL complete with the CGI program name to which a tester may send test files. File content does not need to be X12 or other NAESB WGQ standard format to accomplish this step in testing.
2. Develop or acquire a batch browser that uses multipart for the encoding methodology. Transfer the same test file as in step 1 to the URL not requiring Realm One security.
3. Add Realm One security to your file transfer, and change the URL to the secure URL. Continue transfer tests with your batch browser.
4. Acquire and install PGP or OpenPGP compliant software. Generate your public and private key pair. Make sure to choose the RSA key generation algorithm for PGP or DSA and El Gamal for OpenPGP. Download the test server's test public key. Encrypt your data file using this key. Modify your file transfer to send the encrypted file. Continue transfer tests. Request that the test server contact decrypt your file.

HTTP Server and CGI

1. Install Web server. Establish an Internet connection to your server. Ensure that you have ample storage space for transferred files. Ensure that permissions are granted to the directories.
2. As an optional preliminary step, acquire or develop an HTML page for interactive file upload (sample code is earlier in this document). Test interactive file upload to your own server using an interactive browser.
3. Acquire or develop a CGI program to receive file transfers and process according to NAESB WGQ standards. Test transfers to your CGI using your batch browser.
4. Transfer a X12 or other NAESB WGQ standard format dataset to your server and process it through your translator or other appropriate processes.
5. Copy the CGI to a "secure" directory where Realm One security, or basic authentication, is enabled. Using your batch browser, transfer to both URLs, with and without authentication. Thoroughly test using the incorrect userid and password against the secure directory.
6. Generate a second public/private key pair. Use the second key to encrypt a file and transfer the file to your server. Decrypt the file.
7. Once your site security is established, contact a trading partner to test transfers against your server.
8. Test with various file sizes to ensure that your CGI can process small and large files.
9. Request that several other trading partners and/or several clients within your own company transfer concurrently to ensure that your server can withstand the load.
10. Test application with various simulated errors in both file transfers and in PGP or OpenPGP decryption.



FREQUENTLY ASKED QUESTIONS

As an end user, do I need a continuously connected internet Web server to participate in the Internet EDM in the ~~gas~~-energy industry, or can I just use a dial-up connection to my ISP and my favorite shrink-wrapped browser software?

An interactive browser connection is not enough to actively participate in the system. It is not necessary to have a private Web server, you can use a service, however the system requires that you have access to a permanent internet connection which is capable of both sending and receiving files (~~with CGI or BGI~~) without operator intervention.

If we use ANSI X12.58 encryption do we still need to use PGP or OpenPGP encryption?

The use of internal encryption such as X12.58 is outside the scope of the NAESB ~~WGQ~~-REQ encryption standards.



TABLE A - Internet EDM Standard Error Codes And Messages

These errors and warnings are strictly related to problems found in the ~~recipient CGI-receiving program~~ or decryption levels of processing before translation. Errors and warnings generated by the client batch browser are assumed to be documented at the client site to distinguish them from problems occurring in the ~~recipient CGI-receiving program~~ or decryption. Numbering schemes and descriptions should aid in this distinction.

Note: For HTTP-related information refer to the HTTP section of Appendix A error codes see the ~~NAESB WGQ home page~~ for information sources.

EEDM### standard error format with ### representing a numeric value; further processing will not take place

WEDM### standard warning format with ### representing a numeric value; further processing will take place

The string for the error or warning should appear in the following format:

Validation Code:Description;supplemental message to be defined by the issuing site up to 80 characters

Internet EDM Standard Error Codes and Messages

Validation Code	Description	Data Element	Data Element Required <u>or vs.</u> Mutually Agreed
EEDM100	Missing "from" Common Code Identifier code	from	required
EEDM101	Missing "to" Common Code Identifier	to	required
EEDM102	Missing input format	input-format	required
EEDM103	Missing data file	input-data	required
EEDM104	Missing transaction set	transaction-set	mutually agreed
EEDM105	Invalid "from" Common Code Identifier	from	required
EEDM106	Invalid "to" Common Code Identifier	to	required
EEDM107	Invalid input format	input-format	required
EEDM108	Invalid transaction set	transaction-set	mutually agreed
EEDM109 <u>(OPEN ISSUE 016)</u>	No parameters supplied	parameter string	required
EEDM110	Invalid "version"	version	required
EEDM111	Missing "version"	version	required
EEDM112	"receipt-security-selection" not mutually agreed	receipt-security-selection	mutually agreed

Validation Code	Description	Data Element	<u>Data Element Required</u> or vs. Mutually Agreed
EEDM113	Invalid "receipt-security-selection"	receipt-security-selection	mutually agreed
EEDM114	Missing "receipt-disposition-to"	receipt-disposition-to	required
EEDM115	Invalid "receipt-disposition-to"	receipt-disposition-to	required
EEDM116	Missing "receipt-report-type"	receipt-report-type	required
EEDM117	Invalid "receipt-report-type"	receipt-report-type	required
EEDM118	Missing "receipt-security-selection"	receipt-security-selection	mutually agreed
EEDM119	Mutually agreed element, refnum, not present	refnum	mutually agreed
EEDM601	Public key invalid (R)	file itself	required - security
EEDM602	File not encrypted	file itself	required - security
EEDM603	Encrypted file truncated	file itself	required - security
EEDM604	Encrypted file not signed or signature not matched	file itself	required - security
EEDM699	Decryption Error		required for general decryption errors not specifically identified by PGP or OpenPGP messages or exit codes
EEDM701 (OPEN ISSUE 017)	EDM party not associated with EDI party		required
EEDM702 (OPEN ISSUE 017)	Data Structure Error		required if the translator does not handle this exception
EEDM703 (OPEN ISSUE 017)	Data set exchange not established for Trading Partner		required if the translator does not handle this exception
EEDM999	System error		required for general system errors to indicate severe errors in processing at the receiving site
WEDM100	Transaction set sent not mutually agreed	transaction-set	mutually agreed
WEDM102	"receipt-security-selection" not mutually agreed	receipt-security-selection	mutually agreed
WEDM103	Missing "receipt-security-selection"	receipt-security-selection	mutually agreed
WEDM104	Element refnum received, not mutually agreed; ignored	refnum	mutually agreed

APPENDIX A - Reference Guide

~~CGI~~ Receiving Program

Options include Active Server Pages (ASP), Common Gateway Interface (CGI), Java Server Pages (JSP), Java Servlet technology and other technologies.

Information on ASP may be found on Microsoft's web site (www.microsoft.com).

An ~~excellent~~ source on CGI is a book entitled "Special Edition Using CGI" by Jeffrey Dwight and Michael Erwin.

Information on JSP and Servlet technology may be found at SUN's web site (<http://java.sun.com>).

Firewall Security

An ~~excellent~~ source which covers this topic in detail is a book entitled "Firewalls and Internet Security: Repelling the Wily Hacker" by William Cheswick and Steven Bellovin.

NAESB

NAESB Web Site: (~~<http://www.naesb.org>~~ www.naesb.org) Primary reference for ~~natural gas energy~~ industry standards.

~~General NAESB WGQ FTF Reference Page: (<http://www.naesb.org/ftf.htm>). This location provides pointers to samples and further documentation.~~

HTTP

The NAESB ~~REQWGQ~~ EDM architecture is based on HTTP 1.1, and all implementations should be compatible with this version.

W3C WorldWide Web Consortium. All aspects of HTTP, HTML, and other Web-related topics are documented at:

<http://www.w3.org/pub/WWW/>

General information regarding HTTP with basic terminology included are documented at: <http://www.w3.org/pub/WWW/Protocols/HTTP/1.1/spec.html>

Syntax information for multipart can be found in IETF RFC1341 section 7.2. (www.ietf.org)

HTML

~~Before April 24, 1998, the recommended standard from the WorldWide Web Consortium was HTML 3.2. The specification for this standard can be found at:~~

~~http://www.w3.org/pub/WWW/TR/REC_html32.html~~

~~Effective April 24, 1998, the WorldWide Web Consortium has made a recommendation for HTML 4.0. Information on HTML 4.0 may be found at <http://www.w3.org/TR/REC-html40/>. (OPEN ISSUE 018)~~

~~<http://www.ncsa.uiuc.edu/General/Internet/WWW/HTMLPrimer.html>~~

~~<http://www.interlink-2000.com/guide-to-publishing-html.html>~~

~~Special Edition Using HTML, Second Edition, Mark Brown, John Jung, and Tom Savola, Que Corporation, 1996.~~

PGP Software

PGP is available for a variety of operating systems and platforms. For more information contact Network Associates (<http://www.nai.com>) or PGP Corporation, (www.pgp.com)

OpenPGP Software

The IETF OpenPGP standard is available at <http://www.ietf.org/rfc/rfc2440.txt>

Software implementations of the OpenPGP standard are freely available for commercial use from the Free Software Foundation at <http://www.gnupg.org>.

Time Synchronization

Testing has shown that the clocks on all computer systems drift. ~~It has also been surprising to see just how much they do.~~ Time synchronization is required to assure that all trading partners transaction times are accurate. Time accuracy is dependent on how much a system's clock drifts, how frequently it is resynchronized and the accuracy of the source used for synchronization.

Authoritative time synchronization is now being provided by governmental agencies around the world based on a synchronized network of atomic clocks. In the United States this includes the U. S. Naval Observatory and the National Institute of Standards and Technology.

An easy way to obtain the current time is from the U. S. Naval Observatory's Web site at <http://tycho.usno.navy.mil/cgi-bin/timer.pl>. The output from this page can easily be edited and reformatted to set a local system's time. Commercial, shareware and public domain packages are also available to synchronize system times. Among them are NTP (which is an internet standard), internet daytime, nisttime / usnotime.

Further information on time synchronization may be found at the following Web sites:

~~[http://_www.eecis.udel.edu/~mills/ntp/test.html](http://www.eecis.udel.edu/~mills/ntp/test.html)~~

~~<http://tycho.usno.navy.mil/ntp.html>~~

~~<http://www.ccd.bnl.gov/xntp>~~

~~<http://www.txdirect.net/users/sfisher/clock.html>~~

~~<http://www.is.co.za/resources/ftpsite/tucows/softsync.html>~~

Java

A source on Java is a book entitled “Java & XML, Second Edition: Solutions to Real-World Problems”, by Brett McLaughlin.

The official site for the Java Language is <http://java.sun.com>.

XML

A source on XML is a book entitled “XML in a Nutshell: A Desktop Quick Reference”, by Elliotte Rusty Harold, W. Scott Means.

The official web site for XML standards is located at www.w3.org/XML/



APPENDIX B - Repudiation and Validation Examples

Repudiation and Validation examples:

When a transaction file is received using the EDM mechanism there are several questions that typically must be answered:

- 1.) Is the HTTP sender (from) valid to send to the HTTP 'to' party?
- 2.) Does the HTTP sender match the party who encrypted and signed the file?
- 3.) Does the HTTP sender match the sender within the file?
- 4.) Is that sender with the data valid to 'speak' for the parties transacting business?

Is the HTTP sender (from) valid to send to the HTTP 'to' party?

The first validation, determining that a party is a valid sender must be done in the receiving process during CGI execution. This is simply a 'look up' verification that the Common Code Identifier 'from' is recognized as a valid sender.

Does the HTTP sender match the party who encrypted and signed the file?

The next validation, determining that the HTTP sender is the same as the signer, requires that the following information be available:

The 'from' common code identifier (9 digit D-U-N-S@ Number (OPEN ISSUE 001)). This is the second field in the HTTP post message command sent to the CGI receiving program. This information must be preserved from that earlier process and passed to the 'post CGI' for use in the receiving process.

The Pretty Good Privacy (PGP) or OpenPGP User ID associated with that same party. (OPEN ISSUE 011)

To compare these items a 'table' would most likely be established that would allow the post-CGI receiving process to identify that there is a correlation between these identifiers. The origin of the 'from' identifier is the HTTP POST 'from' field. The origin of the PGP or OpenPGP user ID is the decryption process. The PGP or OpenPGP User ID of the signer is a byproduct of file decryption on a signed file. If PGP or OpenPGP software is executed from the command line the output would be presented in a format like:

```
Good signature from user "ABC CORP".  
Signature made 1997/05/13 2003/04/01 19:30 GMT  
Plaintext filename: test3
```

If PGP or OpenPGP is executed using a program interface the User ID that signed the file will be provided in a buffer. Comparing this buffer to the expected User ID would serve to verify this value.

Does the HTTP sender match the sender within the file?

The data file itself indicates (in the case of X12 data) the sender and the intended recipient within the ISA segment. Although this may be the same (D-U-N-S® Number) as the 'from' data these fields are not standardized. This may require the use of a 'table' to relate these identifiers. [\(OPEN ISSUE 019\)](#)

Consider also that, although it is strongly recommended that only a single ISA be contained within a file, that the process should account for the possibility of several ISA segments. This comparison will ensure that the parties used during translation are in fact the parties that sent, encrypted and signed the data.

Is that sender with the data valid to 'speak' for the parties transacting business?

This last validation is listed here only to complete the chain of identity. The process that would evaluate this relationship would typically be the business application. Since we have checked the identity through each step of this process this is the point at which the identity of the sender would finally be verified as having a business relationship to conduct the business specified.

APPENDIX C - Minimum Technical Characteristics and Guidelines for the Developer and User of the Customer Activities Web Site²

~~Browser Characteristics (includes defined NAESB WGQ current versions):~~

~~Features as supported by the latest generally available (GA) versions of both Netscape³ and Internet Explorer³ within 9 months of such GA version becoming available, including—~~

~~Frames & Nested Frames
Tables & Nested Tables
HTML
Cookies
JavaScript
SSL 128-bit RSA Encryption
Style Sheets~~

~~Plug-ins (Generally Available (GA) versions within 9 months of such GA versions becoming available)~~

~~JAVA[®]
ActiveX⁴ (Plug-in for Netscape[®])
Adobe Acrobat PDF Reader⁵
Systems Incorporated
Independent Computer Architecture (ICA[®]) — Protocol used for remote control access to an application~~

~~Operating Systems:~~

~~Operating systems on a client workstation should be multithreaded and pre-emptive.~~

~~Hardware:~~

~~CPU —————>=500 MHz
Memory —————>=256 MB Physical
Display Resolution —————>=1024 x 768, 16K colors
Connection —————>=56 KB (v.90)~~

² — Configuration shown indicates a minimum except where a specific level is established. 'Minimum' implies a level where a reasonable experience for the user may be achieved. These levels also indicate the level that a user may expect that a client has been tested. Results may be less than satisfactory, or may preclude use of a site, if the user chooses to use anything less than those levels shown.

³ — Netscape[®] is a registered trademark of Netscape Communications Corp.

³ — Internet[®] Explorer is a registered trademark of Microsoft Corporation.

⁴ — ActiveX[®] is a registered trademark of Microsoft Corporation.

⁵ — Adobe Acrobat[®] is a registered trademark of Adobe.

Example Configuration¹

Hardware:	CPU: P500 MHz or higher Memory: 256MB Physical Display Resolution: 1024 x 768, 16K colors Pointing Device with left and right click capability
Operating Systems:	Windows² 98 Windows² NT 4.0 Windows² 2000
Connection:	56KB (v.90) modem ISDN Direct Connect (T1, Fractional T1, etc.) DSL Cable-Modem
Browser:	Netscape[®] Communicator/Navigator Microsoft[®] Internet Explorer
Plug-ins:	JAVA[®] ActiveX[®] (Plug-in for Netscape[®]) Adobe Acrobat Reader[®] ICA[®]

~~Memory - Users who want to have multiple applications or EBBs open simultaneously should consider more memory.~~

~~CPU Speed - Users should be aware that higher CPU speeds may result in better performance.~~

¹ _____ Specific products should be reviewed prior to implementation for Year 2000 compliance. Examples provided represent a non-comprehensive set of configurations that a client may use. This example list in no way should be construed as an endorsement by NAESB WGQ of any specific products. Other products meeting the minimum technical characteristics of the client workstation may be used.

² _____ Windows[®] is a registered trademark of Microsoft Corporation.

~~APPENDIX D - Minimum and Suggested Technical Characteristics and Guidelines for the Developer and User of the Informational Postings Web Site~~

User technical characteristics provide specifications to the developer on the user environment for which the application will be designed and tested. Likewise, they will serve as guidelines to the user when purchasing the appropriate hardware and software to enable him/her to use the application.

~~Informational Postings Web Site User Technical Characteristics~~

	Minimum	Suggested
Connection Device:	28.8-KB	Direct Connect
Operating System:	Multi-threaded & Preemptive	
RAM:	128 MB	>128 MB
Browser Capabilities:	Cookies & JavaScript Frames & Nested Frames Tables & Nested Tables HTML 3.2	
Display Resolution:	1024x768, 16k colors	>1024x768, 16k colors

~~Definitions:~~

~~Minimum user technical characteristics-~~

~~The environment and components for which the Web site application is designed and tested. This should include:~~

- ~~-a client environment comprised only of characteristics listed above, and,~~
- ~~-support for all mandated functions in accessing Informational Postings.~~

~~Suggested user technical characteristics-~~

~~Environment or components not required to perform all mandated functions in accessing Informational Postings, but could provide an enhanced user experience.~~

~~Examples of User Workstations Meeting Criteria of Informational Postings Web Site User Characteristics⁴~~

	<u>Minimum</u>	<u>Suggested</u>
Hardware:	Pentium® ² 200MHz or equivalent	Pentium® 500MHz or greater
RAM:	128 MB	>128 MB
Communication Device:	28.8	Direct Connect ISDN Satellite 56 KB modem DSL Cable Modem
Monitor:	12" Laptop 15" Desktop	>12" Laptop >15" Desktop
Display Capabilities:	1024x768 16k colors	>1024x768 >16k colors
Operating System:	Windows® 95 System 7® ³ Solaris® ⁴ 2.5	Windows® XP Windows® 98 Windows® NT 4.0 Solaris® 2.6 System 8® Windows® 2000 Windows® ME Linux
Browser:	Microsoft Internet Explorer®	Microsoft Internet Explorer®

¹ _____ Technical implementations above represent a non-comprehensive set of choices which an implementer may use. This list in no way should be construed as an endorsement by NAESB WGQ of any specific products. Other products supporting technical implementation may be used.

² _____ Pentium® is a registered trademark of Intel Corporation.

³ _____ System 7® and System 8® are registered trademarks of Apple Computers, Inc.

⁴ _____ Solaris® is a registered trademark of Sun Microsystems, Inc.

~~Netscape®
Communicator~~

~~Netscape®
Communicator~~

~~Informational Postings Web Site Developer Technical Characteristics~~

~~User's environment supporting the above minimum characteristics should be able to access all NAESB ~~WGQ~~ standardized features of Informational Postings Web Sites.~~

~~Any other Web technologies may be considered for use by the developer as long as they can be used by the client without requiring special actions including firewall rule changes, use of a specific browser, logons and downloads of special helper applications such as plug-ins, viewers or readers.~~



APPENDIX E - MINIMUM TECHNICAL CHARACTERISTICS FOR AN EDM SERVER

Allowable TCP Ports (not UDP ports)

HTTP HTTPS 80, 443, 5713, 6112, 6304, 6874, 7403

ICA® 1494

RMI(Java®) 1099-1100

Java® Telnet 31415

TCP Optional 8001-8020**

SMTP 25

Allowable UDP Ports (not TCP ports)

Secure ICA 1604

There are other technologies available that would require additional ports to be opened, such as FTP and Telnet. If and when NAESB ~~REQ~~WGQ approves such technologies, FTTF will modify this list of allowable ports accordingly. The client-side firewall implementation and client browser settings should permit the downloading and installation of NAESB ~~WGQ-REQ~~ approved plug-ins and modules. ~~Please refer to the NAESB WGQ defined Minimum Technical Characteristics for Accessing Customer Activities Web Sites for the listing of NAESB WGQ approved plug-ins and modules.~~ [\(OPEN ISSUE 020\)](#)

**The reservation of 20 optional ports was to provide room for implementations such as DCE, IIOP, and load balancing implementations. ~~TSPs~~ [The parties](#) should endeavor to minimize the usage of these ports.

ICA® is a registered trademark of Citrix Systems Inc.

JAVA® is a registered trademark of Sun Microsystems, Inc.