Report to the Uniform Code Council (UCC)

ebXML Interoperability & Conformance Validation

Final Status

Test Round ebXML–4Q01

March 1, 2002

Report to the Uniform Code Council (UCC)

ebXML Interoperability & Conformance Validation

Final Status

Test Round ebXML–4Q01

v2

Prepared By:
DRUMMOND GROUP, INC.
www.drummondgroup.com

Test Participants

DRUMMOND GROUP, Inc. is pleased to announce that the following participants in the ebXML Interoperability & Conformance Validation Test 4Q01 have completed all requirements and passed tests (*see Final Test Results*) between each product demonstrating interoperability and conformance to the ebXML-MS v2.0 document.

To fully understand what completing the test means in the use of the products in production, please read this document carefully.

For the specified tests, the participating products are in full compliance with the ebXML-MS v2.0 specification. In some cases, it was necessary to extend the specification to provide functionality required by the marketplace. These issues are documented below (*see Interoperability Caveats*).

Sincerely,

Rik Drummond
CEO Drummond Group Inc.

bTrade, Inc  www.btrade.com Product Name: **ebXML Connector version: 1.0**	***Cyclone Commerce***  www.cyclonecommerce.com Product Name: **Cyclone Activator version: 4.2** **Cyclone Interchange version: 4.2**
Sterling Commerce, Inc  www.sterlingcommerce.com Product Name: **STERLING Integrator™ version: 1.2.0**	***Sybase, Inc***  www.sybase.com Product Name: **Web Services Integrator(WSI), version 2.5**

Drummond group

Table of Contents

Abstract

This is the first round of testing for ebXML and it focused primarily on the Messaging architecture. The test was originally slated to work with v1.0, but as work on v1.1 progressed, it became increasingly obvious that this new version would not be backward compatible with the version approved in Vienna in May, 2001. A decision was made to conform to the version in work – v1.1 which has now been renamed to v2.0 – and the test paralleled and participated in the development of this new specification. Many of the members of this test are a part of the OASIS ebXML-MS Technical Committee in addition to actively participating in other OASIS ebXML efforts. This test not only validated those efforts, but discovered and corrected a number of problems in the specification. In some cases, the members of this Interoperability test were able to test or develop concepts prior to their adoption by the committee.

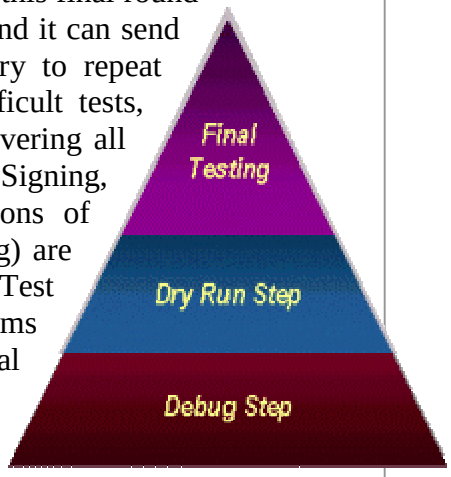
This Interoperability test was timed to complete in tandem with the release of the ebXML-MS v2.0 so confidence is high concerning compliance with this next version of the specification. Although the purpose of the test was specifically focused upon Interoperability, every effort was made by all test participants to assure 100% conformance to the specification. This effort was enhanced by the scrutiny of the output of each participants code by all other participants of the test.

While conformance to the specification was strictly adhered to, there was one functional requirement where the testing participants agreed to extend the specification – Encryption/Confidentiality. This new functionality was developed as a joint effort of the Testing group and presented to the ebXML-MS team. The team has asked that the results of this testing and implementation effort be presented to the TC after the completion of the test. Since this is an extension to the specification, this test was made optional. The write-up for this new functionality can be obtained from: http://www.drummondgroup.com/pdfs/Encapsulation.pdf.

The Test

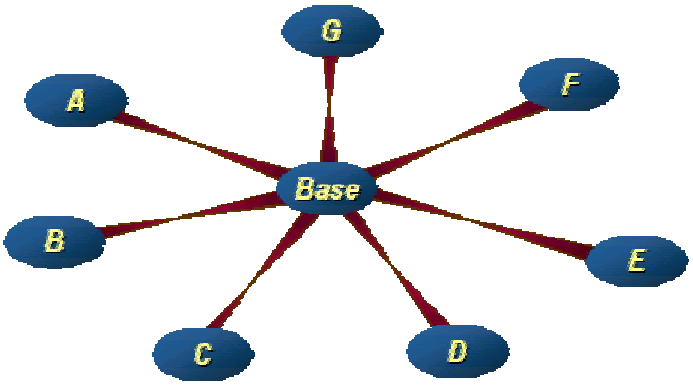
The test is structured so that the early tests are simple transfer tests for small data files. This allows the participants to work out problems focusing on path issues – firewalls, proxies – without complicating such issues with more complex problems encountered later. The next test then builds on the success of these early tests, progressing from small data files to larger data files, adding security (signatures and encryption), requesting Acknowledgments and performing retries (Reliable Messaging). The result is a testing process, which builds from the simple to the complex, correcting problems as they arise along the way. Once problems are

identified and corrected, the resultant system is fully interoperable between all participants in the test. However, since there have been changes along the way, one more final tests is required to provide assurance that previously run tests have not become invalidate. It is not necessary to repeat all tests in the testing program in this final round of tests. If a system can send large files correctly, and it can send small files with signatures, then it is not necessary to repeat simple small file tests. A dozen of the most difficult tests, covering all aspects of the testing procedure and covering all phases of the testing program (Large files, Signing, Encryption, Receipts, Signed Receipts, Combinations of these features, & Error Handling/Reliable Messaging) are selected and designated as the Final Test. The Final Test is run twice, once to make sure there are no problems and that everyone will pass – a Dry Run, then a formal Final test.



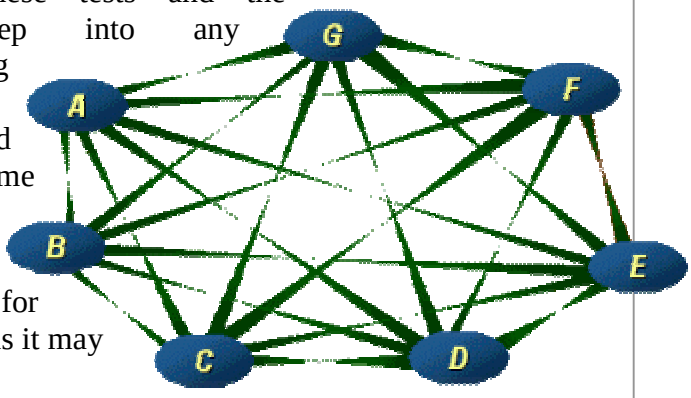
Full Matrix Testing

The usual method of testing systems such as these has historically centered on exercising each of the systems against a base Reference Implementation. While this may assure a measure of conformance to the specification in question, it cannot assure that the systems in question interoperate. Interoperability is not commutative: If A⇔B and B⇔C this does not assure that A⇔C.



There is a way to make this assurance? Yes – Full Matrix Testing.

However, if the systems are to be rigorously tested against each other, then is testing against a base Reference Implementation any longer needed? In order to remain completely neutral concerning the results of these tests and the inconsistencies, which creep into any implementation, including Reference Implementations, DGI has not provided for this added requirement since it has become unnecessary.



Full matrix testing means there are many, many sends/receives for each test. This is not as difficult as it may

seem at first glance. Participants allow direct access to their systems from the Internet. Each system then sends to each other participating system. The test is closely regulated so that one, or at most two related tests are being sent at any one time. For a single test there will be $(n-1)^2$ sends and receives but each participant is only required to keep track of their own sends and receives (n-1). For 12 participants, there will be 121 sends/receives of the same test. Each participant keeps track of 11 of these and reports success/failure to the moderator. It is then the DGI moderator's task to collate the results from all participants and report to the group.

Reporting

Reporting is done using spreadsheet procedures. Since each test has a sender and a receiver, each test run has two reports. If the sender reports success and the receiver reports success (for the same test) then the test is considered complete. By this method of double reporting, the authenticity of each successful report is cross-checked and verified. This reporting is done for each test, with each participant.

To maintain control of the tests and to synchronize the efforts of the participants, DGI provides an eMail list server to which only the tests participants have access, and DGI requires participation in daily conference calls where the results of the last test are discussed and the order and timing of the upcoming tests are announced. The moderator is responsible to slow down the test, to keep the participants together and to make sure any problems are solved prior to proceeding to more difficult tests.

The Interoperability Test

What is the ebXML Compliance Validation Test?

The process for each **Test Round** has three, interrelated Test-Steps. Each **Test-Step** has a series of **Tests**.

Test Round ebXML-4Q01 *(October 2001)*

Test-Step-1 (Debug Step)

Complete all tests from each test to allow Code Check & Debug. The tests comprise tests A through K.

Test-Step-2 (Dry Run)

Software installed from scratch following "written" install procedure as it appears in the Product-with-version Installation Manual.

Run representative subset of the tests (e.g. Tests A, E.1, E.2, E.3, E.4, E.6, F.4, G.3, J.1, J.4)

Test-Step-3 (Final Conformance Validation)

Final Verification and official interoperability & conformance test. One or Two Day Event (no product fixes or code debug activities will be allowed). Successful test completion demonstrates Interoperability to UCC/DGI satisfaction.

Test Round ebXML-xQ02 *(Date TBD)*

There will be other rounds of testing annually (or semi-annually as required). Continued UCC sanctioned ebXML **Interoperability & Conformance Validation** *will require each Product-with-Version to participate in future Testing Rounds to retain their conformance rating.*

Each subsequent Test-Round is composed of the same three Test-Steps as described above.

Test-Rounds

A Test-Round is designed to help companies implement interoperable products by conducting a series of product verification tests. Testing is conducted periodically – usually at least once a year or when three or more new products become available to establish a new Test-Group.

Interoperability & Conformance Validation

UCC *Conformance Validation* of a product-with-version will be issued when the following have been completed to UCC/DGI's satisfaction: The product-with-version has passed Test-Step-3 as defined in this document. This means that each product must exchange information as described in Tests A through K between all products-with-version that have previously passed Test-Step-3 tests.

Interoperability Caveats

Testing Conditions

Interoperability is highly dependant upon test conditions and specifications. Altering or exceeding the conditions under which the test is performed may significantly alter the interoperability results. The three primary impediments to interoperability are Firewalls, Proxy Servers and Certificate Configurations. The former two must be configured locally to allow access to and from the Internet. The later, Certificates, should conform to X.509 standards and any system should ignore extensions not understood. However, since most vendor products incorporate security toolkits, it is not entirely within the control of the ebXML software vendors to support all possible certificate fields or extensions. For this reason, certificates should be kept as simple as possible, with one field or Set per Sequence (*see description below*). Creativity is not encouraged when building certs.

Although higher levels of security are available, it was deemed sufficient and prudent to perform the tests using the following certificate attributes:

Client Certificates (signing/encryption)

- 128/1024 bit encryption
- Triple DES
- SHA1

Server Certificates (HTTP/S)

- SSL port 443 (unless otherwise specified)
- Server Side Authentication only
- No Basic Authentication (not necessary although supported in most cases)

Data Types

Testing consisted of transporting a variety of test data types, EDI-X12, EDIFACT and XML of a variety of sizes. The following MIME Content-Types were used although no payload parsing was tested:

```
Content-type: multipart/related
Content-type: application/PKCS7-mime
Content-type: application/XML
Content-type: application/EDI-X12
Content-type: application/EDIFACT
Content-type: image/jpeg
Content-type: text/XML
```

Final Test Results

This is the final result of the last step. To understand what each test means, please read the Test Descriptions below. Also note that test G.3 is OPTIONAL. This test involves and extension of the ebXML-MS specification for Encryption.

To read the table, first find a participant and read across to the appropriate test column. For instance, if participant C sent and received test E.3 then the they reported successfully receiving the test from participants a, b & d (abd) and reported successfully sending the test to participants A, B & D (ABD). *As discussed above, not all test were included in the Final Test.*

Completed

STATUS	A	E1	E2	E3	E4	E6	F4	G3*	J1	J4	STATUS
A *bTrade*	bcd BCD	bcd BCD	bcd BCD	bcd BCD	bcd BCD	bcd BCD	bcd BCD	bd BD	E	d BCD	**DONE**
B *Cyclone Commerce*	acd ACD	acd ACD	acd ACD	acd ACD	acd ACD	acd ACD	acd ACD	ad AD	E	ad	**DONE**
C *Sterling Commerce*	abd ABD	abd ABD	abd ABD	abd ABD	abd ABD	abd ABD	abd ABD	abd ABD	E	ad	**DONE**
D *Sybase*	abc ABC	abc ABC	abc ABC	abc ABC	abc ABC	abc ABC	abc ABC	ab AB	E	a ABC	**DONE**
E *DGI*									abcd		

Note: Caps mean sent, Lower case mean received

* Test G3 is OPTIONAL and therefore NOT REQUIRED for any certification.

Uppercase indicates successful send. Lower case indicates successful receive. All participants submit a cumulative status sheet after each set of tests (see below) and the results are summarized on the table above.

Test G Encryption is optional. This test was added to provide Encryption functionality requested by some implementors and customers. Details on this process: http://www.drummondgroup.com/pdfs/Encapsulation.pdf.

Test J was comprised of two Reliable Messaging events. Reliable Messaging is a method of dealing with a particular class of errors. Since a perfectly working system will never invoke Reliable Messaging, a staged set of errors needs to be induced in order to test this functionality.

Test J1 tests Retry functionality. To artificially create an environment where the Retry feature would be activated, DGI set up an HTTP server to accept data, but not respond. When a message is sent to this system with Reliable Messaging enabled, the sending system expects an Acknowledgment. When no Acknowledgment Message is forth coming, the sending system should retry the same, exact message again. This

continues until the number of retries equals the value of Retries in the system configuration. By viewing the log on the DGI system, the successful sends and retries may be observed at the proper time intervals.

Test J4 involves testing the response of a system when receiving multiple, identical messages, as would occur when the sending system is issuing retries. Having two participants send a normal message to all other participants, and then trigger a retry of the identical message to all participants again, created this scenario. In this case, the test for compliance is for the sending system to receive exactly the same Acknowledgment on the retry as on the original send (same MessageId, and Timestamp). As shown in the table above and in the report sheet below, the two sending participants for this test were Sybase and bTrade. In the results table, for example, Sybase sent to "ABC" (bTrade/Cyclone/Sterling) and received from "a" (bTrade). In the table below, Cyclone received from bTrade and Sybase.

Test results were gathered directly from the participants.

Sample Test Report Sheet

STATUS		A	E1	E2	E3	E4	E6	F4	G3*	J1	J4
A	*bTrade*	S/R	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa		S/Rpa
B	*Cyclone Commerce*									S/Rpa	
C	*Sterling Commerce*	S/R	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa			
D	*Sybase*	S/R	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa	S/Rpa		Rpa

**Note*: *This is a sample report sheet for Cyclone, i.e. no data in the Cyclone row.*

Only success was reported. Since each Test involved a full duplex (both directions i.e. a send and a reply) for each participant and each participant captured and reported receiving and responding to each test, all tests were reported in duplicate. If a participant believed they sent data and received receipts correctly, that result must be correlated with the corresponding report from the participant at the receiving end of that transaction. In this way, we gather full interoperability testing results from all participants and to all participants (full matrix). This gives a positive interoperability result on each test data and for each testing scenario.

While not every possible situation may be tested, a large portion of the expected real-world scenarios are represented.

Optional Tests

Note that in the table above, an OPTIONAL test was included in the Final test. This is one of several optional tests included in the testing. Other

tests included more testing of Encrypted File Transfer and Message Services – Ping/Pong and Message Status. Test numbering is not always sequential. The most likely reason for this is the adjustment of the test plan during the testing procedure – this is the first time this test has been administered and the specification being tested was not always stable at the time of the test. Some tests, such as Foreign Language tests, were conceived but later removed due to testing difficulties or impracticality. Some tests, such as *Delivery Failure* or *Delivery Failure Notifications*, apply specifically to Multi-Hop and are not appropriate to this test, and thus were removed. Some of these test may be reworked and included in future test rounds.

One particular test (E.5) will be of particular importance as the Business Process nature of ebXML is developed. This test included a Signed Acknowledgment with a returned payload. However, without a more mature choreography, this test cannot yet be administered.

Test	Test Description	Test Data
A	Certificate exchange	
E1	Unsigned Data/Unsigned Acknowledgment	Small X.12
E2	Unsigned Data/Signed Acknowledgment	Small EDIFACT
E3	Signed Data/Unsigned Acknowledgment	Small JPEG (Bridge)
E4	Signed Data/Signed Acknowledgment	Small XML
E6	Signed Data/Signed Acknowledgment HTTP/S	Small X.12
F4	Multiple Payload Signed with Signed Acknowledgment – five payloads	4 files above + 10MB XML
G3/Optional	Signed & Encrypted Data/Signed Acknowledgment	Small X.12
J1	Retries, RetryInterval	Small XML
J4	Duplicate Detection	Small XML

Note: For test J.1, all participants will send to the DGI test system as Clients.
For test J.4, bTrade and Sybase will act as Clients and send to all participants.

Test Descriptions

Test Setup Conditions:

Each participant should obtain three (3) personal certificates and one (1) Server/SSL certificate. Each participant should obtain a persistent Internet connection and support HTTP and HTTP/S connections. These URLs will be eMailed to the eMail List as part of Test A.

This test focuses on basic Messaging Functionality – ebXML-MS v2.0 specification. Although this test will not include Business Processes, CPPA or Reg-Rep specification testing, certain basic functionality from those specifications must be supported in order to run this test.

It will be up to each participant, how they will create an environment, which can best support this test. Messaging, by itself, does not create a closed-loop to facilitate testing. There must be some supporting applications, such as encryption/decryption modules, signature verification, applications to provide basic reply and receipt support, etc. Each participant should anticipate the test environment and create appropriate, simple test applications. There will be no payload parsing tests so, the test application should not issue errors related to payload content. The only payload testing we will do is, after decryption, comparisons of the received data file (payload) with the known payload. This is to ensure no changes – no added bytes at the end or changed bytes within the payload body.

Test A Certificate Exchange

Test Description:

The three main impediments to interoperability are Firewalls, Proxies and Certificates. Prior to initiation of Interoperability Testing, Firewalls and Proxies must be configured to allow HTTP and HTTP/S connections from the Internet. In addition, Certificates must be exchanged between all participants and installed on the receiving system.

A.1 Personal Certificate

Test Setup:

The test suite requires the use of security and connection information. This test is a basic way of establishing communication between the test participants. An eMail will be prepared containing the requisite information with two (or three) certificates attached. The eMail will be sent to the List Serve.

All test participants should install their systems from scratch and clean all certificates from their test system certificate database. Certificates obtained from the List Server should then be installed into the certificate database. Information obtained from the eMail should be used to create entries in a CPA database (or equivalent) including a Delivery Channel for each transfer protocol. This is not intended to be a test of the CPPA system and equivalent functionality is acceptable.

The following RECOMMENDED settings should apply:

- HTTP Timeouts set to at least 30 minutes (*we are going to send some big files*)

- SyncReply *not present for SMTP and present for HTTP and HTTP/S*
- Retries *set to* 3
- RetryInterval *set as required. TBD*
- TimeToLive *must comply with:*

```
TimeToLive > currentTime + (Retries * RetryInterval)
```

- PersistDuration *should be very long and must comply with:*

```
PersistDuration > Retries * RetryInterval
PersistDuration + SendTime > TimeToLive
```

- duplicateElimination *set to* `false` (default)
- MessageOrderSemantics *set to* `NotGuaranteed`
- MessageId *recommend set to* `date` & `time` & `dailyIndex@CompanyURL`
- ConversationId *recommend set to* `date` & `time` & `dailyConversationIndex`

Expected Results:

Successfully install certificates from each of the other participants. Create CPA entries for each of the other participants. This information must be present before any data transfer may begin.

Notes:

It is important to include the entire chain-to-root for each certificate. The method of obtaining these certificates is left to the participants. These may be obtained from any public Certificate Authority or they may be self-generated certificates. Any standard or extension fields may be populated.

It is the goal of this test to include as wide a variety of certificate configurations as possible. Implementations should support as many of the certificate fields as possible. If the field or entry is not understood, it should be ignored. It is not uncommon for particular extensions to be unrecognized by some security toolkits. These are the problems this test is designed to discover.

Test B Simple Data Transfer

Test Description:

Successfully send a small file to each test participants. This tests basic connectivity and port configurations. This also ensures basic ebXML structure conformance. Systems MUST be able to support multiple PartyID values in both From and To elements.

B.1 HTTP Data Transfer
B.2 HTTP/S Data Transfer

Test C Large File Transfer

Test Description:

The ability to send and receive large files will be a deciding point for some customers. However, the limits of many test platforms will be reached with too large a file. It is the goal of this test to stress the software platforms without engaging in performance assessment of the test hardware or Internet connection. However, systems are not necessarily scalable, i.e. a small file may work while larger files may expose problems not experienced with the smaller transfer. Unfortunately, many test systems are not robust enough for full performance testing. As a compromise, a 10MB test file will be used for this test.

Participants are cautioned to set timeouts very high for HTTP. Participants should also carefully monitor disk-space availability during this test.

C.1 HTTP Large File Send

Test D Data Security

Test Description:

Data security and sender authentication will be a market requirement as it is with all B2B systems. Authentication is performed through digital signatures (XMLdsig).

D.1 Signed Data
D.2 Signed Data Secure Channel (HTTP/S)

Test E Acknowledgments

Test Description:

An Acknowledgment provides *Transport* level acknowledgement that a message was received by the To Party MSH. An Acknowledgment element may be present on any message.

An Acknowledgment may be requested by creating an AckRequested element in the originating message:

The ds:Reference element(s) should always be present in a signed Acknowledgment to provide NRR (Non-Repudiation of Receipt).
Unsigned Data/Unsigned Ack

E.1 Unsigned Data/Signed Ack
E.2 Signed Data/Unsigned Ack
E.3 Signed Data/Signed Ack
E.4 REMOVED: Signed Data/Signed Ack with Returned Payload
E.5 Signed Data/Signed Ack Secure Channel

Received data should be identical to Reference data.

Original Sender should receive an Acknowledgment message containing an Acknowledgment element. RefToMessageId should correspond to the original message. If the Acknowledgment is signed, the ds:Reference element should contain a hash of the original message which should be identical to the hash of the Reference Data and the hash of the header information.

Test F Multiple Payload Handling

Test Description:

The purpose of this test is to extend the single payload capability of the participant MSH implementations to multi-payload capabilities.

F.1 Multiple Payload Transfer – two payloads
F.2 Multiple Payload Transfer – five payloads
F.3 Multiple Payload Signed – two payloads
F.4 Multiple Payload Signed with Signed Acknowledgment – five payloads

Test G Encrypted File Transfer – OPTIONAL

Test Description:

Data security is achieved through S/MIME encryption of the header information including the Manifest information. In this case, the entire ebXML message, including headers, will be encrypted and encapsulated as the payload of a secure message. The result of decryption may be an entire ebXML message with headers, which must be re-parsed as a normal ebXML message.

When a message is both signed and encrypted, the message with headers must first be signed and then encrypted. The resulting encrypted object will then be placed as the payload of a simple ebXML message as described above.

Encrypt a file and transfer. This exchange is to test the ability to send to non-HTTP/S systems with data security. A large file (10MB) will be used for this test.

G.1 Encrypted Data Exchange
G.2 Signed & Encrypted Data Exchange
G.3 Signed & Encrypted Data/Signed Ack
G.4 Signed & Encrypted Large File/Signed Ack
G.5 Multiple Payload Encrypted – two payloads

Encrypt a file and transfer. This exchange is to test the ability to send to non-HTTP/S systems with data security. A large file (10MB) will be used for this test. Detailed Process: http://www.drummondgroup.com/pdfs/Encapsulation.pdf.

Test H Message Services – OPTIONAL

Test Description:

This tests the Ping service allowing an MSH to determine if another MSH is responding.

H.1 Ping/Pong
H.2 Message Status

Test I REMOVED: Other Languages

Test Description:

Create a Description element with the appropriate xml:lang attribute.

I.1 Spanish
I.2 Japanese

Test J Single-Hop Reliable Messaging

Test Description:

For this test, Reliable Messaging should be turned ON and there should be an AckRequested element present.

J.1 Retries, RetryInterval
J.2 Acknowledgement
J.3 REMOVED: Delivery Failure Notification
J.4 Duplicate Detection
J.5 REMOVED: SequenceNumber

For test J1, messages will be sent to the DGI test system. The test system will not reply and the sending system should Retry after RetryInterval and, after the retry count exceeds Retries, notify the application of a Delivery Failure (this may be implementation dependent).

For test J2, two participants will send the test message to each participant and then trigger a retry. The receiving system should recognize the second message as a duplicate and return the original acknowledgment. It is not necessary that all participant act as client. Each participant must act in the server role.

For test J4, there will be a DuplicateElimination element present. At least two participants should send messages to the other test participants and then resend the identical message. The receiving system should recognize the second message as a duplicate and should return the original, unsigned Acknowledgment but should not notify the application of the duplicate messages.

Test K Error Handling

Test Description:

EbXML specifies a number of errors which are generated during the parsing or security validation of a received message. This test is designed to simulate a number of errors and validate system handling due to those errors. Some errors will return error messages to the sender while others may be recoverable errors. Still other situations may be created by time-outs

Since these test sends cannot be initiated by correctly operating systems, these tests will be performed by a third system. The error test system will send erroneous files to each participant to generate the required errors.

K.1 SOAP:Fault
K.2 ValueNotRecognized
K.3 NotSupported
K.4 Inconsistent
K.5 REMOVED: OtherXML
K.6 REMOVED: DeliveryFailure (multihop only)
K.7 SecurityFailure
K.8 REMOVED: Unknown
K.9 TimeToLiveExpired

The error test will be created by directly editing properly formatted files and creating specific errors. For instance:

Test K.7. a signature failure can be created by changing any single byte within the signature block.

Test K.9 will be created by setting TimeToLive to an artificially low value so the receiving system, even though the message arrived successfully, will report an error.

About DRUMMOND GROUP, INC.

The Drummond Group Inc. works with software vendors, vertical industry and the standards community to drive adoption for standards by facilitating vertical industry pilots, interoperability conformance testing and building competitive supply chain strategies. Founded in 1990, the vendor-neutral group represents best-of-breed in the industry on linking horizontal infrastructure technologies, standards and interoperability issues with the needs of vertical industry such as retail, grocery, healthcare, transportation, government and automotive.

For further information, please contact Beth Morrow at Beth@drummondgroup.com