

Final Report

ebMS Interoperability Test

Fourth Quarter 2007 (4Q07)



December 14, 2007

Prepared & Administered By:
DRUMMOND GROUP INC.
www.drummondgroup.com

Table of Contents

Cover Letter.....	4
Disclaimer.....	5
Test Participants.....	6
Definitions.....	7
Interoperability Test Summary.....	8
Interoperability Test History.....	8
Test Summary: Basic Profile.....	9
Test Summary: XML Encryption with SSL Client Authentication Profile.....	11
Test Summary: Automotive Retail Profile - GZIP Based Compression.....	12
Test Summary: Digital Signatures using DSAwithSHA1.....	12
Interoperability Issues.....	14
Interoperability Issues Resolved or Confirmed in this Round (4Q07).....	14
Use of SOAPAction header with the Apache Web Server.....	14
Use of ISO10126Padding algorithm for XML Encryption.....	16
Interoperability Issues Resolved or Confirmed – ebMS 4Q06.....	16
Empty responses must be HTTP 204 replies with an empty body (Updated 4Q06).....	16
SyncReply element may appear in a synchronous reply (Updated 4Q06).....	17
Digital Signature Algorithm RSA .vs. DSA for Signing.....	18
DSA Signature Certificate not fully support SSL Cipher Suites.....	18
Multiple inbound signed messages caused Apache XML Security Jar to miscalculate the hash.....	18
Common Name Must support the correct PrintableString Characters.....	18
Interoperability Issues Resolved or Confirmed – ebMS 4Q05	19
Use of KeyInfo elements for XML Encryption	19
XML Encryption Key sizes.....	19
Duplicate Elimination in an Acknowledgment message	19
X509 Extensions.....	19
Additional Discussion.....	19
Distinguished Names within Digital Certificates.....	19
Timing of Asynchronous replies versus http responses.....	20
Use of CPA (Collaboration Protocol Agreement).....	20
HTML formatted errors.....	20
Differing interpretations on the use of ConversationID	21
SyncReply without expected Response	21
Ping/Pong Synchronicity	21
MessageID and Content-ID	21
Interoperability Issues Resolved or Confirmed – ebMS 4Q04.....	22
Empty responses must be HTTP 204 replies with an empty body.....	22
SOAP Action HTTP Header in Sync Reply responses.....	22
The SOAPAction value must be "ebXML" with the quotation marks.....	22
Format for Messages without payloads.....	22
SOAP Faults.....	23
Role Element value.....	23
cid based start parameter in the MIME Content-Type header.....	23
MIME Multipart/Related header, case insensitivity.....	23
XML DSIG KeyInfo.....	23
XML DSIG Namespace Attribute.....	24
Payloads are not canonicalized during the digital signing process.....	24
Payloads are not canonicalized for Digital Signature / XMLEncryption	24
Sign first, Encrypt second.....	25

Compress first, Sign second.....	25
Compressed data will not be base64 encoded.....	25
Reference to original message MUST be included -signed Acknowledgment.....	26
MessageID MUST conform to MIME and include @.....	26
Content-ID SHOULD conform to MIME and include @.....	26
Error Messages should not be signed.....	26
Content-type for XML Encryption.....	26
XMLDSIG Namespace declaration must be at Signature element level.....	27
KeyName must be X509 Distinguished Name format.....	27
XMLSchema Instance declaration must be present.....	27
Timestamps may include fractions of seconds.....	28
Test Requirements.....	29
Trading Partner Requirements.....	29
Technical Requirements – Basic Profile.....	29
Message Packaging.....	30
Digital Signature.....	30
Error Handling.....	30
Synchronous and Asynchronous messaging.....	30
Synchronous and Asynchronous Acknowledgments of Receipt.....	30
Transfer Protocols.....	31
Payloads.....	31
Large Messages.....	31
Reliable Messaging.....	32
Message Status.....	32
Ping/Pong.....	32
Error Handling.....	32
Technical Requirements - XML Encryption & SSL Client Authentication Profile.....	33
Client Authentication.....	33
XML Encryption.....	33
Technical Requirements – Automotive Retail Profile for GZIP based Compression.....	34
GZIP Based Compression.....	34
Industry recommended common header field values.....	34
Debug Phase Basic Profile Test Suite.....	35
Overview of the Interoperability Compliance Process®.....	36
DGI In-the-Queue Test Round.....	36
DGI Interoperability Test Round.....	37
About Drummond Group Inc.....	38

Cover Letter

DRUMMOND GROUP INC. is pleased to announce that the following participants in the Drummond Certified™ ebXML Message Service Interoperability Test 4Q07 (ebMS-4Q07) have completed all requirements and passed all required tests (see Final Test Results) between each product demonstrating interoperability and conformance to a Basic Profile subset of the ebMS version 2.0 specification. The Certification Run was performed on Dec. 3-4, 2007.

Several participants engaged in and successfully completed additional optional tests of Technical Profiles which comprise compliant supersets of the ebMS v2.0 standard based on recommendations from industry specific sources:

- Profile for XML Encryption and SSL Client Authentication as described by the Centers for Disease Control and Prevention (CDC)
- Automotive Retail Profile for GZIP based Compression as described by Standards for Technology in Automotive Retail (STAR)

Furthermore, several participants engaged in and successfully completed additional optional tests using the DSAwithSHA1 digital signature algorithm for signing ebMS messages over both HTTP and HTTPS. While these were optional tests for the 2007 certification, these tests will be required for the 2008 certification.

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


Sincerely,

Rik Drummond
CEO,
Drummond Group Inc.

Disclaimer

Drummond Group Inc. (DGI) conducts interoperability and conformance testing in a neutral test environment for various companies and organizations ("Participant"). At the end of the testing process, DGI may list the name of the Participant in the final test report along with an indication that the Participant passed the test. The fact that the name of the Participant appears in the final report is not an endorsement of the Participant or its products or services, and DGI therefore makes no warranties, either express or implied, regarding any facet of the business conducted by the Participant or their product.

Test Participants

 <p>Axway</p> <p>http://www.axway.com</p> <p>Product Name: Synchrony Gateway Interchange v5.5 / Synchrony EndPoint Activator v5.5</p>	 <p>Cleo Communications</p> <p>http://www.cleo.com</p> <p>Product Name: VersaLex™ v3.5 tested in VLTrader™ v3.5</p>
 <p>Generix Group</p> <p>http://www.generixgroup.com</p> <p>Product Name: ebMS for TradeXpress version 2.1</p>	 <p>IBM</p> <p>http://www.ibm.com</p> <p>Product Name: Websphere Partner Gateway v6.1</p>
 <p>Inovis</p> <p>http://www.inovis.com/</p> <p>Product Name: BizManager v3.1</p>	 <p>TIBCO Software Inc.</p> <p>http://www.tibco.com/</p> <p>Product Name: TIBCO BusinessConnect ebXML Protocol 5.0.0</p>

Definitions

Interoperability -- A product is deemed interoperable with all other products in the Interoperability Test Round if and only if it demonstrates in a full-matrix manner the pair wise exchange of data covering the *Test Criteria* between all products in the Interoperability Test Round. A product is either totally interoperable or it is not interoperable. Waivers or exceptions are not given in demonstrating interoperability for the *Test Criteria* unless the entire *Product Test Group* and DGI agree.

Interoperable products – is that group of products, from the *Product Test Group*, which successfully completed the *Test Criteria*, in a full duplex manner with every other *Product Test Group* participant in an Interoperability Test Round without any errors in the final test Phase. Interoperable products receive a Drummond Certified™ Seal.

Product Test Group – A group of products involved in an interoperability or conformant Test Round.

Product, product-with-version, or product-with-version-with-release – are interchangeable and are defined for the purpose of a Test Round as a product name, followed by a product version, followed by a single digit release. The assumption is that version and release syntax is as: “VV.Rx...x,” where VV is the version numeral designator, R is the single digit release numeral designator and x is the sub-release multiple digit numeral designator. DGI assumes that any digits of less significance than the R place do not indicate code changes on the product-with-version-with-release tested in the Test Round. A vendor must list a product as product name, followed by version digits followed by a decimal point followed by a single release designator digit before the Test Round is complete.

Test case – The test criteria is a set of individual test cases, often 10 to 50 which the product test group exchange among themselves to verify conformance and interoperability.

Test Criteria – A set of individual tests, based on one or more standard specifications, that is used to verify that a product is conformant to the specification(s) or that a set of Product-with-version’s are interoperable under the *Test Criteria*.

Interoperability Test Summary

ebMS v2.0 is a Message Service protocol for reliable Business-to-Business data interchange. ebMS v2.0 adds quality of service features on top of transfer protocols such as HTTP and SMTP. Key qualities of service features include guaranteed delivery and non-repudiation of receipt. ebMS v2.0 can reliably transfer any data type including XML, X12, EDIFACT, or binary data between two parties over the Internet. The purpose of this test is to provide software vendors a neutral venue to test interoperability of ebMS v2.0 products in a non-competitive environment with the goal to accelerate adoption of high quality ebMS v2.0 deployments.

This is the seventh round of DGI Interoperability testing of the OASIS ebXML Message Service specification version 2.0 (ebMS v2.0). The test was performed from October to December 2007.

During the seventh round of ebMS testing (4Q07), DGI maintained the “In the Queue” process to ebMS testing rounds where new participants tested for compliance against reference platforms before joining the full interoperability test round. Six participants successfully tested over the Basic Profile, and five participants executed the additional Industry/Technology profiles for:

- XML Encryption and SSL Client Authentication
- Automotive Retail with GZIP based data compression of payloads

For the seventh round of ebMS testing, a couple of new optional tests were defined in order to test interoperability using the DSAwithSHA1 digital signature algorithm over both an HTTP and HTTPS connection. Five participants successfully executed the DSA tests. Beginning with the 2008 ebMS Test Round, the DSA tests will be part of the required Basic Profile series of tests.

Interoperability Test History

ebMS 4Q06 Interoperability Test October - December 2006

During the sixth round of ebMS testing (4Q06), DGI maintained the “In the Queue” process to ebMS testing rounds where new participants tested for compliance against reference platforms before joining the full interoperability test round. Six participants successfully tested over the Basic Profile, and five participants executed the additional Industry -Technology profiles for:

- XML Encryption and SSL Client Authentication
- Automotive Retail with GZIP based data compression of payloads

ebMS 3Q05 Interoperability Test September - December 2005

During the fifth round of ebMS testing (3Q05), DGI introduced the “In the Queue” process to ebMS testing rounds where new participants tested for compliance against reference platforms before joining the full interoperability test round.

ebMS 3Q04 Interoperability Test September - December 2004

During the fourth round of ebMS testing (3Q04), all participants successfully tested over the Basic Profile and several participants tested additional Industry/Technology profiles for:

- XML Encryption and SSL Client Authentication
- Automotive Retail with GZIP based data compression of payloads
- HL7 version 2 and version 3 payload support

ebMS 3Q03 Interoperability Test September - December 2003

During the third round (3Q03), DGI defined an approach to accelerate multiple industry interoperability. A Basic Profile was defined to include all required ebMS v2.0 features plus several optional features including reliable messaging, ping/pong and message status and industry requested features were tested as separate profiles. Six participants successfully tested over a profile for XML Encryption & SSL Client Authentication where details prescribed the CDC.

ebMS 3Q02 Interoperability Test August - December 2002

In the second round (3Q02) error testing was expanded and all participants tested directly with the Centers for Disease Control’s implementation of ebMS. Two participants executed optional testing of XML Encryption & SSL Client Authentication.

ebMS 4Q01 Interoperability Test September - December 2001

The first round (4Q01) covered all ebMS v2.0 required features including message packaging with additional testing of reliable messaging features, multiple attachments, SSL, SMIME encryption, informal testing of error scenarios. Test Case Summary

Test Summary: Basic Profile

The following tests were identified as representative of the overall Basic Profile test suite and are composed of the most complex features. These tests comprise

the ebMS v2.0 Dry Run/Final Run Test Suite and were executed as the Final Test.

Test	Description	Transfer	Sync/Async	Payload
C1	Large Message	http	async	Very Large X12
E1	UnSigned with Ack	http	async	Small XML
E3	Signed Data/UnSigned Ack	http	async	Small XML
E4	Signed Data/ Signed Ack Sync	http	sync	Small XML
E5	Signed Data/Signed Ack SSL	https	async	Small XML
F3	Two Payloads Signed Data	http	sync	Small XML Medium binary jpeg
F4	Five Payloads Signed Data/Ack SSL	https	async	Medium X12 HCCO Small EDIFACT Small XML Large XML Medium binary jpeg
G1	Ping Pong	http	sync	none
H1	Once and only once	https	async	Small XML
H2	Duplicate Detection	https	async	Small XML
I2	Value Not Recognized	http	sync	Small XML
I3	Not Supported	http	sync	Small XML
I4	Inconsistent sync	http	async	Small XML
I5	Security Failure	http	sync	Small XML
I6	Time to Live expired	http	sync	Small XML
I7	Message header format	http	sync	Small XML
I8	Missing Payload	http	sync	Small XML
I9	Delivery Failure	http	async	Small XML

Interoperability is determined by each product-with-version successfully sending and receiving each test case with the others. A test case is successful when the expected result is achieved according to the message specifications.

On Dec. 3-4, 2007, all products-with-version listed on this test report successfully sent and received test cases E1, E3, E4, E5, F3, F4, G1, H1 and H2 with each and every other participant. Test case C1 (Large Message) was successfully sent and received between each participant over the course of the Debug, Dry Run, and Certification Run. The I2-I9 tests were successfully executed between each participant and DGI hosted test server.

It should also be noted that no warranty of product interoperability is implied over and above the publishing of the results of the Test Round as completed by all vendors during the specified time period of testing.

Large Messages

The C1 Large Message Test is included in the Basic Profile as a straightforward test of a product-with-version's ability to send, receive and process large messages (50 megabyte). The test is not intended as a stress test or as a performance test. DGI does not require a full matrix test for Large Messages to avoid performance problems related to memory issues, as participant test

servers are typically medium sized servers. However, during the entire Interoperability Test Event, each participant exchanges a large message with every other participant. Participants were grouped with two partners during the Debug Phase to successfully test Large Messages. During the Dry Run Phase, participants were partnered with two different partners. Finally, during the Certification Run, participants were partnered with the last remaining partner with whom they had yet to exchange large messages. In effect each participant tested large message sends each of the other participants during the test round.

Error Testing

Error tests defined in the Basic Profile are not tested in a round-robin fashion. Messages-in-error are from a DGI hosted test server and sent to participants.

During the Debug Phase, the Dry Run and the Certification Run, a full range of tests designed to test the error handling of each participant's Product-Under-Test were repeated with each participant. The executed Error Tests are listed in Debug Phase Basic Profile Test Suite.

Test Summary: XML Encryption with SSL Client Authentication Profile

The following participants participated in the XML Encryption with SSL Client Authentication Industry Optional Profile Tests identified below.

Company	Product, Version
Axway	Synchrony Gateway Interchange v5.5 / Synchrony EndPoint Activator v5.5
Cleo Communications	VersaLex™ v3.5 tested in VLTrader™ v3.5
Inovis	BizManager v3.1
TIBCO Software Inc.	TIBCO BusinessConnect ebXML Protocol 5.0.0
IBM	WebSphere Partner Gateway v6.1

As the ebMS 2 specification does not provide detailed requirements or recommendations for the use of XML Encryption, this profile makes use of CDC experience and recommendations for the use of XML Encryption with ebMS.

The following tests were executed by each participant noted above. Each participant successfully executed each test against the other participants involved in this optional test.

Test	Description	Transfer	Sync/Async	Payload
J1	Client Authentication	https	sync	Small XML
J2	Client Authentication & XML Encryption	https	sync	Small XML
J3	Client Authentication, Digital Signature & XML Encryption	https	sync	Small XML

For the Dry Run and Certification Run, only the J2 and J3 tests were executed by the participants that opted in for the XML Encryption optional profile testing.

Test Summary: Automotive Retail Profile - GZIP Based Compression

The following participants participated in the Automotive Retail with GZIP based Compression Industry Optional Profile Tests identified below.

Company	Product, Version
Axway	Synchrony Gateway Interchange v5.5 / Synchrony EndPoint Activator v5.5
Cleo Communications	VersaLex™ v3.5 tested in VLTrader™ v3.5
Inovis	BizManager v3.1
Generix Group	ebMS for TradeXpress v2.1
IBM	WebSphere Partner Gateway v6.1

As ebMS 2 specification does not provide detailed requirements or recommendations for the use of compression, this profile makes use of the STAR (Standards for Technology in Automotive Retail) Profile experience with ebMS.

The following tests were executed by each participant noted above. Each participant successfully executed each test against the other participants involved in this optional test.

Test	Description	Transfer	Sync / Async	Payload
K1	XML Payload Synchronous	https	sync	Small XML, PartsOrder BOD
K2	XML Payload Asynchronous, Compressed	https	async	Large XML, PartsInvoice BOD
K3	XML Payload Asynchronous, Compressed and Signed	https	async	Large XML, PartsInvoice BOD

For the Dry Run and Certification Run, only the K2 and K3 tests were executed by the participants that opted in for the Automotive Retail optional profile testing.

Test Summary: Digital Signatures using DSAwithSHA1

The following participants participated in the DSA Digital Signatures Optional Profile Tests identified below.

Company	Product, Version
Axway	Synchrony Gateway Interchange v5.5 / Synchrony EndPoint Activator v5.5
Cleo Communications	VersaLex™ v3.5 tested in VLTrader™ v3.5
Inovis	BizManager v3.1
TIBCO Software Inc.	TIBCO BusinessConnect ebXML Protocol 5.0.0
IBM	WebSphere Partner Gateway v6.1

The ebMS 2 specification recommends the use of the DSAwithSHA1 algorithm for digitally signing ebMS messages. Historically this ebMS certification event has used the RSAwithSHA1 algorithm because of its widespread use in the marketplace. However, since the ebMS specification does recommend the use of DSA, this certification event offered an optional test to certify the interoperability of the use of DSAwithSHA1 digital signatures over both HTTP and HTTPS.

The following tests were executed by each participant noted above. Each participant successfully executed each test against the other participants involved in this optional test.

Test	Description	Transfer	Sync/Async	Payload
E6	DSA Signed Data with Unsigned Ack	http	sync	Small XML
E7	DSA Signed Data with Signed Ack	https	async	Small XML

For the Dry Run and Certification Run, only the E6 test was executed by the participants that opted in for the DSA Signatures optional profile testing.

Interoperability Issues

During the first six ebMS interoperability rounds, several issues arose that required consensus to achieve interoperability. Some of these items are outside the scope of the ebMS v2.0 and are related to underlying technical specifications such as MIME, and some of these issues address ebMS v2.0 features which have been interpreted differently by different readers.

The consensus items are separated into the timeframe of the interop.

Interoperability Issues Resolved or Confirmed in this Round (4Q07)

Use of SOAPAction header with the Apache Web Server

The presence of multiple SOAPAction headers in an ebMS message in combination with certain HTTP proxies can inadvertently lead to a non-conformant SOAPAction header format. The solution would be for ebMS message senders to ensure that messages contain one and only one SOAPAction header.

An ebMS v2.0 message should contain one (1) and only (1) SOAPAction header with the value and format as specified by a consensus item from the 4Q04 test round. This clarifying constraint is to ensure that the ABNF specification for the SOAPAction header (as defined in the SOAP 1.1 specification) is not violated by certain HTTP proxies (such as using the Apache Web Server in proxy mode).

During the 4Q07 Interoperability Test Event, it was discovered that if an ebMS message contained multiple SOAPAction headers

```
POST /ebxmlhandler HTTP/1.1
Connection: Keep-Alive
Host: 127.0.0.1
SOAPAction: "ebXML"
MIME-Version: 1.0
Content-Length: 2478
SOAPAction: "ebXML"
Content-Type: multipart/related; type="text/xml";boundary="-----MyMimeBoundary"
```

and an Apache Web Server was being used by the receiver as an HTTP proxy, the Apache Web Server was combining the multiple SOAPAction headers into a single SOAPAction header with a comma-separated list of values according to Section 4.2 of RFC2616. This resulted in the ebMS receiving handler seeing the Apache-modified SOAPAction headers as:

```
SOAPAction: "ebXML", "ebXML"
```

The relevant text in Section 4.2 of RFC2616 says:

Multiple message-header fields with the same field-name MAY be present in a message if and only if the entire field-value for that header field is defined as a comma-separated list [i.e., #(values)]. It MUST be possible to combine the multiple header fields into one "field-name: field-value" pair, without changing the semantics of the message, by appending each subsequent field-value to the first, each separated by a comma. The order in which header fields with the same field-name are received is therefore significant to the interpretation of the combined field value, and thus a proxy MUST NOT change the order of these field values when a message is forwarded.

The first two sentences in the text above are important for interpretation. The HTTP specification does allow for multiple instances of a header to occur in an HTTP message, but only under the stipulation that the header values can be described as a list of comma-separated values and if that combination of values doesn't change the semantics of the message.

The SOAP 1.1 specification and more specifically to Section 6.1.1 for clarification of SOAPAction header field in SOAP messages, the specification provides the following ABNF syntax for SOAPAction:

```
soapaction      = "SOAPAction" ":" [ <"> URI-reference <"> ]  
URI-reference   = <as defined in RFC 2396>
```

The ABNF syntax does not include the variable repetition operator for the URI-reference element that would specify that list of comma-separated values are permissible for this field. And since the URI-reference element is enclosed in square brackets, this equivalent to

```
*1(<"> URI-reference <">)
```

According to ABNF syntax, the *1 syntax means there should be exactly 0 or 1 field-values in the form of a URI-reference for the SOAPAction header.

It is clear that the presence of HTTP proxies can legally (per HTTP 2616) transform multiple SOAPAction headers into a single header with a list of comma-separated values. But, the HTTP specification only allows for senders to send multiple header if the value can be placed into a comma-separated list of header values. From the ABNF syntax specified by the authors of the SOAP 1.1 specification, it appears that the specification intends for their to be *1 (0 or 1) values for the SOAPAction header, which is further constrained by ebMS v2 and DGI Interop Test Consensus to a fixed value of "ebXML". As such, the presence of multiple SOAPAction headers in an ebMS message in combination with certain HTTP proxies can inadvertently lead to a non-conformant SOAPAction header format. The solution would be for ebMS message senders to ensure that messages contain one and only one SOAPAction header.

Use of ISO10126Padding algorithm for XML Encryption

During the 4Q07 Interoperability Test Event, one participant was using the DESede/CBC/ISO101026Padding algorithm in the process of constructing and sending encrypted ebMS messages during the XML Encryption Optional Profile tests. The use of this algorithm caused decryption problems with a couple of participants that either did not support this algorithm or that their decrypt cipher was initialized with only the DESede/CBC/PKCS5Padding algorithm.

Historically, it is believed that participants in previous Interoperability Test Events had used the more common DESede/CBC/PKCS5Padding algorithm. In order to ensure interoperability, the participant that was using ISO101026Padding switched to PKCS5Padding.

However, for future Interoperability Test Events, participants should be prepared to accept and successfully decrypt ebMS messages that used the ISO101026Padding algorithm during the encryption process.

Interoperability Issues Resolved or Confirmed – ebMS 4Q06

Empty responses must be HTTP 204 replies with an empty body (Updated 4Q06)

The ebMS v2.0 specification states under section B.2.3 that “A 2xx code MUST be returned when the HTTP Posted message is successfully received by the receiving HTTP entity.” and under section B.2.4 in the case of asynchronous messaging “a HTTP response code in the range 2xx MUST be returned when the message is received successfully”.

During previous test rounds interoperability issues have been found when participants did not return a simple empty HTTP reply in these situations.

The consensus has been developed that in these situations an HTTP reply with code 204 must be returned and the body of the HTTP must be empty.

There has been strong agreement and proof of interoperability over this consensus including agreement that it is in compliance with the ebMS v2.0 specification and in compliance with RFC2616 the HTTP 1.1 specification.

Addendum-4Q06

The argument of 204 .vs. 2xx left some unresolved questions around this consensus. Another conceptual view was necessary to qualify a revelation of this consensus.

DGI Internal review revealed this issue became known due to how HTTP 1.1 persistent connections could cause resources to become unknown or defunct. Consensus items “Timing of Asynchronous replies versus http responses” and “SyncReply without expected Response” also revealed how responses could cause stale resources.

In the absence of 204 the MSH (Message Service Handler) allows the connection to remain open to receive a response/message at any point in time. Response codes other than 204 may cause ebMS to expect an incoming response with a body (message), but it may or may not arrive leaving the connection open for a prolong period of time. This issue caused implementers to question how stagnant resources should be released or removed in a non-harmful manner.

The only viable solution was to issue a 204 response code. The response means the server fulfilled the request. The 204 response channeled the Initiator (request) to abolish unnecessary resources and close the connection (transaction) safely.

This consensus corrected the issue of persistent connections and resource management over an extended period.

SyncReply element may appear in a synchronous reply (Updated 4Q06)

ebMS v2.0 does not forbid the SyncReply element from appearing in a synchronous reply. In general discussion, some participants noted that there may be some use case scenarios where this is useful and a receiving Message Handler (the MSH receiving the response) should allow for SyncReply element in an HTTP reply.

Addendum-4Q06

According to CPP/A V 2.0 lines:1760-1768 says, “the SyncReply element may appear in ebMS replies except synchronous Acknowledgements when the mode is ‘mshSignalsOnly’ “

The SyncReply element must not appear in synchronous Acknowledgements. Asynchronous acknowledgements are checked by the CPA. The CPAs should not request SyncReply on an asynchronous Acknowledgement – no need to ack an ack.

Digital Signature Algorithm RSA .vs. DSA for Signing

DGI recommended Signature Algorithm RSA for ebMS Interops, but DSA is recommended in the ebMS V2.0 Spec. DSA signed messages failed signature validation when received by IBM's XML Security Suite (XSS4J).

XMLDsig says DSA is required but accepts RSA Signature Algorithm.

The XSS4J Toolkit is no longer offered as a product by IBM. It has been rolled into the JSR 105/106 Security Services.

To ensure interoperability among the Test Group, every one used RSA. The Test Group believes DSA and RSA are valid in the marketplace, therefore both should be supported.

Future Interops will include Test Cases for both DSA and RSA Signature Algorithms.

DSA Signature Certificate not fully support SSL Cipher Suites

Some toolkits have difficulties validating Cipher Suites (during the SSL Handshake) when the certificate has sha1dsa as the Signature Algorithm.

The solution is to use sha1RSA Signature Algorithm where most suites are supported.

Multiple inbound signed messages caused Apache XML Security Jar to miscalculate the hash

Sender (using JSR 105: XML Digital Signature APIs) sent multiple signed message to the Recipient (using Apache XML Security Jar 1.0.5). The first message was verified successfully by the Recipient, but subsequent signed messages were rejected (signature value verification failures).

To resolve this issue, the Recipient upgraded to Apache XML Security Jar 1.3 – all subsequent signed messages were verified successfully.

Common Name Must support the correct PrintableString Characters

Common Name with an underscore character is an invalid character within this field. The common name field must have valid 'PrintableString' characters. The common name (CN) field in a digital certificate must contain valid characters if it is created as a 'printable string'. Reference ASN.1 for valid PrintableString characters.

Interoperability Issues Resolved or Confirmed – ebMS 4Q05

Use of KeyInfo elements for XML Encryption

Several issues arose during the current test round directly related to Distinguished Names and the use of the XML Digital Signature / XML Encryption KeyInfo element.

As above, DGI recommends that Distinguished Names in digital certificates should be dns names or ip addresses. This enables an expected value for SSL host authentication. Regarding the KeyInfo element, the XML Encryption specification does not recommend any specific type of values. DGI recommends that the value of KeyInfo (if present) be a string representation of the certificate's Distinguished Name per RFC 2253.

XML Encryption Key sizes

During the current test round several participants resolved issues between Java and non Java cryptographic libraries, where the Java based libraries default to 192 bit keys and a Microsoft based product defaulted to 128 bit keys. The issued was resolved by both parties utilizing 192 bit keys.

Duplicate Elimination in an Acknowledgment message

During the current test round, a participant had a problem handling an acknowledgment message that had an unexpected DuplicateElimination element present. After codes changes, the participant chose to allow partner to include DuplicateElimination element in responses. The ebMS v2.0 specification does not explicitly prohibit the presence of DuplicateElimination in Acknowledgments.

X509 Extensions

During the current test round two participants resolved an issue with X509 version 3 extensions. A participant was instantiating an extension that was not recognized by the second participant and was marking it as critical. As a resolution, the participant regenerated a new certificate without the extension.

Additional Discussion

Distinguished Names within Digital Certificates

Many products recommend or require that the Distinguished Name of a digital certificate used for a B2B / Web server be the DNS name or IP Address of the

server. To improve interoperability, DGI recommends that participants follow this convention where possible.

Timing of Asynchronous replies versus http responses

ebMS was originally intended as a transfer protocol neutral standard and provides bindings for SMTP and HTTP. Occasionally issues come up related to HTTP that are not addressed by the ebMS binding discussion, for example ebMS does not attempt to assign meaning to an empty HTTP response. During early test rounds an issue was discovered with the timing of synchronous HTTP responses versus asynchronous ebMS replies. In other words, due to design or network issues it is theoretically possible to receive an asynchronous ebMS level reply before receiving a synchronous HTTP response. When designing for asynchronous messaging, DGI suggests that ebMS implementations do not depend on the timing of HTTP synchronous responses versus ebMS asynchronous replies.

Use of CPA (Collaboration Protocol Agreement)

The DGI ebMS v2.0 Interoperability tests do not require the use of ebXML CPA but does provide testing between implementations where:

- Both partners employ CPA
- Neither partner employs CPA
- One of two partners employs CPA

Although these tests are not intended to formally test CPA features, by default the tests exercise the above configurations and some of the tests exercise behavior that must be implemented by CPA or by a CPA like architecture. For example, Test I3 exercises the ability to return an ebMS error declaring a requested feature is “Not Supported,” the ability to recognize this error may or may not be implemented via a CPA based system. Participants may or may not physically use a CPA based system to support these types of tests.

HTML formatted errors

Per ebMS v2.0, message handlers may return SOAP Faults or ebMS error list messages when errors are encountered. Some implementations return html format errors in specific situations. This is probably due to the fact that message handlers are often implemented in web servlet containers, which by default return html formatted errors. While ebMS v2.0 does allow for error handling at lower level protocols such as HTTP, it is important that end users and implementers be aware that this is a very commonly encountered situation and needs to be taken into account during system design.

Differing interpretations on the use of ConversationID

During previous test rounds, discussions revealed a difference of interpretation on the meaning and use of the ConversationID element.

The ebMS v2.0 specification requires that ConversationID be present in all messages, and requires that if you implement the optional MessageOrdering feature (not tested by DGI) that ConversationID must stay the same over all ordered messages.

End users and implementers should be aware that there are different interpretations of the meaning and use of ConversationIDs and if the use of ConversationID is important its usage should be clearly explained.

SyncReply without expected Response

During the current round a new test, A1, was added with the intention of being the most simple test. This test is an exchange of a synchronous message that did not request an acknowledgment. In practice this test caused runtime problems with some participants, due to the fact that ebMS v2.0 describes that the receiving MSH should keep the connection open in anticipation of returning another ebMS message response. After some discussion, the new test was dropped from the test plan.

Implementers should be aware that some implementations are not designed to expect the presence of SyncReply in cases where an ebMS level response is not expected.

Ping/Pong Synchronicity

During the current round there was a discussion on whether or not the ebMS Ping/Pong services can be implemented asynchronously. If SyncReply is not present on a Ping request does that mean that the Pong response is expected to be received asynchronously? After discussion and code changes, the test that requires/expects asynchronous Ping/Pong behavior was left in the test plan.

MessageID and Content-ID

During the current test round, a participant questioned the consensus on requiring the presence of @ in both the ebms MessageID Element and in the MIME Content-Id header value. On review, DGI has modified the consensus item related to this issue. ebMS v2.0 clearly requires that @ be present in ebMS MessageIDs but it is less clear that @ is required in MIME Content-ID header values and there does not appear to be a best practice to require it.

Interoperability Issues Resolved or Confirmed – ebMS 4Q04

Empty responses must be HTTP 204 replies with an empty body

The ebMS v2.0 specification states under section B.2.3 that “A 2xx code MUST be returned when the HTTP Posted message is successfully received by the receiving HTTP entity.” and under section B.2.4 in the case of asynchronous messaging “a HTTP response code in the range 2xx MUST be returned when the message is received successfully”.

During previous test rounds interoperability issues have been found when participants did not return a simple empty HTTP reply in these situations.

The consensus has been developed that in these situations an HTTP reply with code 204 must be returned and the body of the HTTP must be empty.

There has been strong agreement and proof of interoperability over this consensus including agreement that it is in compliance with the ebMS v2.0 specification and in compliance with RFC2616 the HTTP 1.1 specification.

SOAP Action HTTP Header in Sync Reply responses

Message Handlers should tolerate synchronous replies that contain a SOAPAction header and synchronous replies that do not contain a SOAPAction header. During discussions on this issue the key points made were that some products on the market do format replies with this information, ebMS v2.0 does not take a stand on this issue, while the intention of the SOAP 1.1 HTTP Binding may be that SOAPAction is present only in HTTP requests and not in HTTP replies, the SOAPAction header is not specifically forbidden from being present in a reply.

The SOAPAction value must be "ebXML" with the quotation marks

ebMS v2.0 recommends the value of the HTTP header SOAPAction be “ebXML”, but the specification does not explicitly require that the value must be surrounded by double quotes. In practice, the absence of the double quotes has been shown to cause interoperability problems.

Format for Messages without payloads

Message Handlers should tolerate messages which have no payload sent as multipart/related, and tolerate messages which have no payload sent as plain SOAP format (text/xml). The ebMS v2.0 specification provides examples of both.

SOAP Faults

Message Handlers should tolerate SOAP Faults sent as HTTP 500 responses or sent as HTTP 200 responses, or sent as separate posts. It has been found that many products in the field generate SOAP Faults in the form of HTTP 200 replies and some generate SOAP Faults in the form of HTTP 500 responses. Also, it has been found that the use of HTTP 500 as specified by SOAP 1.1 and ebMS v2.0 can cause some application server or web server products to ignore data in the HTTP body, resulting in a special case that needs to be understood and taken into account.

Role Element value

If the Role element is present, as contained in either ToParty or FromParty elements, its value must be agreed upon by the two testing parties. The ebMS v2.0 specification is unclear on how to set this value. An addendum to ebXML CPA version 2.0 specifies that the value for the Role element should be the same as the Role/@name value in the CPA.

cid based start parameter in the MIME Content-Type header

The start parameter of the MIME Content-Type header may be a cid (content-id) style reference. It may contain a prefix of "cid:" which should be stripped to obtain the Content-ID value. For example,

```
SOAPAction: "ebXML"  
Content-Type: Multipart/Related; type="text/xml";  
            boundary="MIMEBoundary"; start="cid:myContentIDHeader"  
--MIMEBoundary  
Content-Type: text/xml  
Content-ID: <myContentIDHeader>
```

MIME Multipart/Related header, case insensitivity

The ebMS v2.0 specification requires a MIME Content-Type: Multipart/Related header to appear as an HTTP header. Consensus is that Message Handlers must tolerate mixed case in the phrase Multipart/Related. For example these two phrases are both valid "Multipart/Related" and "multipart/related."

ebMS v2.0 does not directly address this issue, but underlying MIME specifications require case insensitivity for this header.

XML DSIG KeyInfo

The KeyInfo element related to Digital Signatures is optional, if it present it may be ignored. In other words, the assumption is that Digital Certificates used to

sign messages will be exchanged out of band and will be known beforehand by all participants.

During previous test rounds, at least one security toolkit in use forced the use of the KeyInfo element if it was present. Because of this, message senders should either not format the KeyInfo element, or make sure that the Digital Certificate passed within the KeyInfo element is the correct Digital Certificate to be used for Digital Signature by the receiving partner.

XML DSIG Namespace Attribute

The attribute that declares the XML Digital Signature namespace must be at the Signature element level, for example:

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmldsig#">  
</ds:Signature>
```

Interoperability issues have been encountered when the attribute is not present at this level.

Payloads are not canonicalized during the digital signing process.

ebMS v2.0 states that transformations applied to payloads are “implementation dependent.” This consensus allows for interoperable validation of signature digests. Some past participants have pointed out that C14 canonicalization removes comments, which could result in security or integrity issues.

Payloads are not canonicalized for Digital Signature / XMLEncryption

When combining XMLEncryption with Digital Signature, payloads will be treated at all times during processing as simple bytestreams, even if the payload is XML being encrypted with XMLEncryption. This follows the above consensus that Payloads are not canonicalized for Signature. This consensus has implications on the way security toolkits are configured and used. Some security toolkits will attempt to canonicalize XML data by default, and must be configured to treat an XML Payload as a simple byte stream.

During previous rounds there have been interoperability problems found in the implementation of this consensus. When processing Encryption and Signature, some security toolkits will by default attempt to canonicalize or de-canonicalize data that it recognizes as XML with XMLEncryption applied. To overcome these issues, some participants had to configure their security toolkits (both sender and receiver implementations) to treat the payloads as simple binary byte streams. In other words, the data was not encoded via DOM or other similar mechanisms, but was treated at all times as a simple byte array.

One specific problem that was found was that if on the receiving side the payload is considered to be XML, the security toolkit will pass the XML data to the end application with the XML prelude declaration

(i.e., "<?xml version="1.0" encoding="UTF-8"?>")

missing, causing the application to view the payload as invalid XML.

Sign first, Encrypt second

For the purposes of the XML Encryption and SSL Client Authentication profile, participants are required to apply Digital Signature first and XML Encryption of the XML Payload second.

There has been discussion during previous rounds that as a general principle there may be situations where it makes sense to encrypt first and sign second. This consensus is based on the original CDC requirements.

Compress first, Sign second

For the purposes of the Automotive Retail Profile for GZIP Based Compression, participants are required to apply compression first and Digital Signature second.

During previous test rounds, there have been discussions that there are cases where signing first and compressing second make sense.

This consensus is based on discussions with STAR members who worked directly on the STAR Transport guidelines. The original intent of recommending this order of operations (compress first sign second) was to allow for implementations where a backend system may be responsible for the compression but the message handler will be responsible for the digital signature.

Compressed data will not be base64 encoded

During the fourth round, the testing group developed a consensus that compressed data will be represented as simple binary data and would not be encoded as base64. The reasoning behind this consensus 1) base64 encoding is not needed as http is considered to be a binary safe protocol 2) base64 encoding and decoding would add unnecessary processing overhead.

This consensus was vetted with the group that developed the STAR Transport guidelines.

Reference to original message MUST be included -signed Acknowledgment

This consensus was gained by a majority of past participants and is supported by ebMS v2.0 specification which states “if you support signed acknowledgments, it is required that you include references to the original message digests.” A further consensus was reached that signed acknowledgments should include these References, even if the original message itself was not signed.

MessageID MUST conform to MIME and include @

Content-ID SHOULD conform to MIME and include @

ebMS v2.0 does not address this issue directly, but the consensus is that the underlying MIME specifications do. The exact consensus is that these ID elements should be formatted in this fashion to comply with MIME specifications, but that receivers should act liberally, and not reject a message solely based on Content-ID or MessageID not containing an @.

In practice, it is generally seen that MessageID is almost always formatted in this fashion, but that MIME Content-IDs are formatted in this fashion only in a minority of cases. The ebMS v2.0 specification shows examples of both.

Error Messages should not be signed

Error Messages should be sent in the clear, to avoid the possibility that the Error is related to signing processes. Specifically, when a signed acknowledgment is requested, and an Error message is generated in reply, that Error message should not be signed.

Content-type for XML Encryption

When implementing XML Encryption, the Content-Type MIME header value for XML Attachments can be either text/xml or application/xml. In other words, the Content-Type should have these values, even if the payload is XML encrypted with the XML Encryption standard. Products should gracefully handle a message that uses one of these two values. This consensus is partially based on recommendations from the CDC.

```
--MIMEBoundary
Content-Type: application/xml
or
--MIMEBoundary
Content-type: text/xml
```

During the 3Q03 ebMS Test round, this topic was discussed. An alternative suggestion was to use the value application/xenc+xml. Investigation into this MIME

type revealed that it has been registered, but not yet accepted formally as a valid MIME type.

XMLDSIG Namespace declaration must be at Signature element level

The attribute that declares the XML Digital Signature namespace must be at the Signature element level, for example

```
<ds:Signature xmlns:ds="http://www.w3.org/2000/09/xmlsig#">
...
</ds:Signature>
```

It is not unreasonable to include this namespace declaration at the SOAP envelope level, but there appears to be an overwhelming best practice to specify it at the Signature element level. During the 3Q03 ebMS test round at least one security toolkit was found not to process XML Digital Signature without its presence.

The XML Digital Signature specification lists a non-normative DTD that describes the declaration at the Signature level. The XML Digital Signature normative XML Schema however, does not describe this format.

All Digital Signature examples from ebMS version 2 specification have the declaration at signature level, an informal survey of examples from the web revealed every available example placing the declaration at the signature level.

KeyName must be X509 Distinguished Name format

For the purposes of the XML Encryption and SSL Client Authentication Profile the value of the KeyName element must be a valid X509 Distinguished name. This is simply a best practice convention used during the test to enable the lookup of the related key, and is the format prescribed by the CDC PHIN architecture. In practice, in the field systems often use "alias" values other than Distinguished Names that are defined and agreed upon by the two parties.

XMLSchema Instance declaration must be present

In past rounds, interoperability problems were encountered if a participant's implementation did not include an XMLSchema instance declaration as below:

```
<SOAP:Envelope
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
&
```

Under section 2.2.2 of ebMS version 2, it is strongly recommended that the declaration be present. It has been found that the absence of the declaration will cause interoperability problems with some implementations, and for the purposes

of the test round the declaration is required to be present, even though in the normative specification this is only a strong recommendation.

Timestamps may include fractions of seconds

In the fourth round, at least one implementation reported problems processing fractions of a second (milliseconds). Discussion among the testing group led to the consensus that XML Schema Datetime format allows for fractions of seconds and that ebMS v2.0 implementations should support its use.

Test Requirements

In order to be part of the certified interoperable products-with-versions, each participant must both successfully send and receive all tests cases in the Basic Profile with each and every other participant.

Trading Partner Requirements

All participants were required to establish trading partner relationships with each other. All participants were remote from each other, and all test messages were exchanged over the public Internet. Participants were responsible for distributing their network information and configuring their firewalls to allow all other participants access to their product-with-version.

Each participant provided their security certificates (including SSL server and client certificates) to the other participants for storage in their trusted store. Each certificate conformed to the X.509 standards but varied with respect to the fields used in the certificates. All participants generated their own self-signed certificates. Most participants chose to use a single certificate for all purposes, including SSL Server Authentication, SSL Client Authentication, Digital Signature and XML Encryption.

For participants that opted in for the optional DSA Signatures Profile tests, these participants generated a second set of DSA certificates for use with the E6 and E7 tests.

DGI provided test payloads and user identification aliases.

Technical Requirements – Basic Profile

Each participant successfully sent and received all tests cases in the Basic Profile with each and every other participant, with the exception of Test C1 (Large Message) and the Error Tests which are executed between a participant and a DGI hosted test server.

The Basic Profile test cases cover the core requirements of ebMS v2.0 and include some optional features of ebMS v2.0 that are widely implemented and or desired by end users. These requirements are described directly below.

The effect is that all the products-with-version are proven interoperable over a feature-rich, industry horizontal profile and demonstrates that the products-with-version can cover the technical requirements listed below. For additional technical information regarding ebMS v2.0 requirements please see the Message Service Specification version 2.0 located at:

http://www.oasis-open.org/committees/ebxml-msg/documents/ebMS_v2_0.pdf

Message Packaging

ebMS leverages SOAP with Attachments (SwA) to define an extensible message package that prescribes message headers for routing, partner identification, message identification, timestamping, digital signature and other quality of service features. The message package is also capable of encapsulating one or more business documents or other binary data as payloads. Participant products-with-version must be capable of formatting SwA messages in the manner described by the specification.

Digital Signature

ebMS v2.0 leverages XMLDigitalSignature to provide proof of content-integrity, authentication of senders and receivers and NonRepudiation. An ebMS v2.0 signature is a signature over the entire message which may include one or more payloads. For the Basic Profile, the RSAwithSHA1 digital signing algorithm is used. There are optional tests for using the DSAwithSHA1 digital signing algorithm.

Error Handling

ebMS v2.0 leverages SOAP Fault semantics for low level SOAP-related errors, and specifies higher level “ebMS error lists” that can be comprised of a list of warnings and or errors that occur at the ebMS transport level. For example, a SOAP syntax error will generally result in a SOAP Fault error reply, while a message where TimeToLive has expired will result in an ebMS defined error list reply stating that the message has expired.

Synchronous and Asynchronous messaging

ebMS supports both synchronous and asynchronous message patterns. The type of message pattern is defined per message. This allows ebMS to be highly transfer protocol neutral and to be used in business scenarios where immediate reply is required and in business scenarios where delayed replies are common due to queuing operations, load balancing, system outages or other technical or business reasons.

Synchronous and Asynchronous Acknowledgments of Receipt

Acknowledgments validate the receipt and persistent storage of a message. Synchronous acknowledgments provide a confirmation of receipt in a message returned over the same session and the same transfer protocol as the original message. Asynchronous acknowledgments are sent back to the originator over a separate session.

Acknowledgments are tested in both synchronous and asynchronous styles, both signed and unsigned. A signed Acknowledgment includes hash digests of the original message allowing for true Non Repudiation of Receipt.

Transfer Protocols

Both HTTP & HTTP/s transports were tested. SMTP was not tested.

Payloads

The ebMS v2.0 message package provides for multiple payloads. Effectively, more than one business document can be sent in a single message. In some cases, the secondary documents may be binary files such as pictures and are often referred to as attachments; conceptually similar to email attachments.

Tests of single and multiple (up to five) payloads were executed, and these tests included Digital Signature and HTTP/S transport.

These payloads were used throughout the testing:

- Medium sized HIPAA compliant X12 document appx. 18k provided by HCCO
- Small EDIFACT EDI document appx. 2k
- Small XML document appx. 600 bytes
- Large XML document appx. 41k
- Medium sized XML automotive PartsOrder BOD appx. 4k
- Large XML automotive PartsInvoice BOD appx. 1meg
- Very large X12 EDI file 50 megabytes
- Medium sized binary jpeg file apx. 11k

Large Messages

ebMS v2.0 provides the ability to transport any data type including large files. As a message service standard gains wider deployment in the market, invariably end users demand the ability to send very large messages. One test was run with a 50 megabyte EDI payload. This test is intended to prove the ability to send and receive large messages, and is not intended as a performance or stress test. Due to the memory requirements of large messaging processing, these tests are not executed in a full matrix or round-robin manner. Specifically, during the Debug Phase, each participant interchanged large messages with two (2) other partners. During the Dry Run Phase, each participant interchanged large messages with two (2) different partners. Finally, during the Certification Run,

each participant interchanged a large message with the final remaining large message partner.

Reliable Messaging

ebMS v2.0 defines features to enable once-and-only-once delivery of messages. This is often referred to as Guaranteed Delivery; a message is received and persisted to storage successfully or the sender is notified of failure.

Tests are executed that exercise the features needed for once-and-only-once:

- Acknowledgment of receipt
- Senders ability to retry failed messages

Message Status

This ebMS v2.0 specific service is used to query the status of a previously sent message. An ebMS v2.0 specific reply is generated listing the previous message as Unauthorized, NotRecognized, Received, Processed or Forwarded.

Ping/Pong

The Ping/Pong feature of ebMS v2.0 can be used as a “keep alive” status message, allowing parties to query the state of a partner’s message handler for management and troubleshooting purposes. Ping/Pong can also be useful as a simple connectivity test when engaging with new partners.

Error Handling

Each participant was sent messages-in-error from a DGI hosted test system. The replies from the products-with-version were analyzed to determine if the participant system recognized the error and responded with an appropriate error response.

Technical Requirements - XML Encryption & SSL Client Authentication Profile

Participants successfully executed an optional suite of tests designed to prove interoperability of XML Encryption and SSL Client Authentication implementations. These tests were executed in a matrix, all participants choosing to opt-in tested as both sender and receiver with the other participant. Five of the six participants opted-in on the XML Encryption Optional Profile Test. The specific details for applying XML Encryption to ebMS v2.0 payloads originated from the CDC. For more information regarding the relationship between CDC PHIN and ebMS v2.0 see:

<http://www.cdc.gov/phinf/messaging/index.htm>

<http://www.cdc.gov/phinf/components>

Client Authentication

Client Authentication is an option of SSL/TLS that allows a Server to authenticate a Client via the Client's possession of a recognizable Digital Certificate. Marketplace deployment of Client Authentication is growing, as organizations realize its potential as a useful part of a business to business security strategy. Participants proved interoperability over SSL Client Authentication and used it for all tests within this Profile.

XML Encryption

ebMS v2.0 allows for the use of a persistent encryption mechanism that can be applied to payloads within a message. Persistent encryption can be leveraged as an additional layer of security for Internet based messaging; essentially part or all of a message payload may be encrypted in a manner that allows only the intended Receiver to decrypt the message. At the time the ebMS v2.0 standard was approved, XML Encryption was still a draft standard, as a result, ebMS v2.0 states that XML Encryption is the preferred encryption method but ebMS v2.0 does not provide detailed methods for applying XML Encryption to ebMS messages.

This profile requires encryption of whole XML payloads using XML Encryption, Participants who executed this Profile successfully interoperated with XML Encryption and also a combination of XML Encryption with DigitalSignature.

Technical Requirements – Automotive Retail Profile for GZIP based Compression

Five of the six participants chose to opt-in on the optional Automotive Retail Profile and successfully executed a suite of tests designed to prove interoperability of key features recommended by the Standards for Technology in Automotive Retail (STAR) consortium's ebMS Implementation Guidelines.

For additional technical information regarding the relationship between STAR and ebMS refer to the STAR documents named Transport Guidelines and ebMS Implementation Guidelines which can be obtained from the Special Interest Groups / Infrastructure section of the public STAR website after completing a free registration. See <http://www.starstandards.org>.

GZIP Based Compression

The use of large messages (multiple megabytes) is common in many industries. There are several available methods for compressing HTTP based messages, the STAR guidelines recommend the use of gzip based compression where the payload itself is composed of compressed data using the MIME type application/gzip.

Participants proved interoperability over gzip based compression including the ability to combine compression with digital signature. For tests that required signature the order of operations were implemented as compress-then-sign for message senders and validate-signature-then -decompress for receivers.

Industry recommended common header field values

To assist in interoperability, the STAR guidelines require common methods for populating some of the key ebMS message header fields. Participants proved interoperability over requirements to populate the Service, Action and Timestamp header fields in accordance with STAR guidelines. Service and Action field values were based on the type of STAR OAG BOD payload in the test message.

Debug Phase Basic Profile Test Suite

A1	Connectivity			Individual Setup Time
B1	Simple Transfer	http	async	Small XML
B2	Simple Transfer SSL	https	async	Small XML
C1	Large Message	http	async	Very Large X12
D1	Signed Data	http	async	Small XML
E1	UnSigned with Ack	http	async	Small XML
E2	UnSigned with ACK sync	http	sync	Small XML
E3	Signed Data/UnSigned Ack	http	async	Small XML
E4	Signed Data/ Signed Ack Sync	http	sync	Small XML
E5	Signed Data/Signed Ack SSL	https	async	Small XML
F1	2 Payloads	http	async	Small XML Medium binary jpeg
F2	5 Payloads	http	async	Medium X12 HCCO HIPPA Small EDIFACT Small XML Large XML Medium binary jpeg
F3	2 Payloads Signed Data	http	sync	Small EDIFACT Medium binary jpeg
F4	5 Payloads Signed Data/Ack SSL	https	async	Medium X12 HCCO HIPPA Small EDIFACT Small XML Large XML Medium binary jpeg
G1	Ping Pong	http	sync	None
G2	Ping Pong SSL	https	async	None
G3	Message Status SSL	https	Async	None
H1	Once and only once	https	Async	Small XML
H2	Duplicate Detection	https	Async	Small XML Medium binary jpeg
I1	SOAP Fault	http	sync	Small XML
I2	Value not recognized	http	sync	Small XML
I3	Not Supported	http	sync	Small XML
I4	Inconsistent sync	http	async	Small XML
I5	Security Failure	http	sync	Small XML
I6	Time to Live expired	http	sync	Small XML
I7	Message Header format	http	sync	Small XML
I8	Missing Payload	http	sync	none
I9	Delivery Failure	http	async	Small XML

During the Debug Phase, every participant executed each Basic Profile test in against each and every other participant, acting as both receiver and sender.

The only exceptions to this matrix style testing were for test C1, which participants are required to only execute with a subset of partners during each Test Phase and the Error Tests where Error Test I9 was executed from the participant to a DGI hosted server and Error Tests I1 through I8 were executed from a DGI hosted server to the participants server.

Overview of the Interoperability Compliance Process®

Interoperability of B2B products for the Internet is essential for the long-term acceptance and growth of electronic commerce. To foster interoperability, DGI facilitates interoperability and conformance tests. This section contains a description of the test process involved with creating and listing interoperable products.

DGI In-the-Queue Test Round

In-the-Queue Test Rounds are designed to allow participants—with products new to DGI interoperability testing, or previously certified products that have made significant product changes or undergone version changes, or missed the most recent test round—to both test and debug their products with the DGI Test Server.

The DGI Test Server is a collection of products-with-version from the previous Interoperability Test Round. These products were provided by the vendors on a voluntary basis. The DGI Test Server allows products new to the interoperability process to be debugged in a quicker manner by testing with proven products-with-version.

Through the In-the-Queue Test Rounds, participants will see their products-with-version become conformant to the ebMS v2.0 standard and interoperable with the DGI Test Server products. Products which successfully complete In the Queue Test Rounds are considered compliant to the respective standard and will be listed on the www.drummondgroup.com website as "In the Queue," but they will not be given product Interoperability Status on the www.drummondgroup.com website.

Successful test completion also qualifies that particular product to participate in the next DGI Interoperability Test round, but does NOT guarantee successful completion of the full Interoperability Test Round. DGI makes no warrants or guarantees that products passing In the Queue Test Rounds will pass the Interoperability Tests.

DGI Interoperability Test Round

Products-with-version from the previous ebMS v2.0 Interoperability Test Round and products-with-version from the In-the-Queue tests come together in a vendor-neutral and non-competitive environment to test with each other in order to become interoperable with each other. In an Interoperability Test Round, each product-with-version must successfully test with each other in order to be certified as interoperable.

The DGI Interoperability Test Round verifies conformance to a standard and then verifies that members of the Product Test Group are interoperable among themselves. Interoperability is an all or nothing within the Product Test Group over the Test Criteria. A product is either interoperable with all other products in the Test Group or not.

Products-with-version which demonstrate complete interoperability among the passing members of the Product Test Group are given a Drummond Certified™ Seal and are listed with Interoperability Status on the www.drummondgroup.com website. Interoperability Test Rounds are periodically repeated to verify that as product names, versions or releases change, the products remain interoperable.

About Drummond Group Inc.

Drummond Group Inc. (DGI) is an independent, privately held company that works with software vendors, vertical industries and the standards community to drive adoption for standards by conducting interoperability and conformance testing, publishing related strategic research and developing vertical industry strategies. Founded in 1999, DGI represents best-of-breed in the industry on linking horizontal infrastructure technologies, standards and interoperability issues with the needs of vertical industries such as retail, grocery, health care, transportation, government and automotive. For more information, please visit www.drummondgroup.com or email: info@drummondgroup.com.