ebXML Messaging Services Conformance Checklist

Message Service Specification Version 2.0 rev C

Part I: Core Functionality

All mandatory syntax, features and behavior must be supported.

Section	***Requirement***	***Level***	***GoXML-MS***
2.1.1	The ebXML message conforms to SOAP 1.1 and the SOAP Messages with Attachments specification.	MUST	
2.1.2	All MIME headers of the Message Package are in conformance with the SOAP Messages with Attachments Specification.	MUST	
2.1.2	The Content-Type MIME header in the Message Package contains a *type* attribute of "text/xml"	MUST	
2.1.2	The Content-ID MIME header contains a *start* parameter identifying the first MIME part.	RECOMMENDED	
2.1.2	Non-Multipart SOAP messages can be used in instances where there are no payload sections.	MUST	
2.1.3.1	The MIME *Content-Type* header for the Header Container (SOAP Message) has the value "text/xml"	MUST	
2.1.3.1	The MIME Content-Type header for the Header Container (SOAP Message) specifies the character set used to generate the SOAP Message.	MAY	
2.1.3.2	The UTF-8 character set is used by default when encoding the SOAP Message.	RECOMMENDED	
2.1.4	If there is no application payload then there are zero payload MIME parts.	MUST	
2.1.4	The contents of each payload MIME part are identified in the Manifest element within the SOAP body.	MUST	
2.1.5	Unrecognised MIME headers in any MIME part are ignored.	MUST	
2.1.6	If a MIME error is present in the ebXML Message then it is reported as per the SOAP with Attachments specification.	MUST	
2.2.1	If the SOAP Message includes an XML prolog it contains the version of XML applied.	MUST	
2.2.2	If the SOAP Message includes an XML prolog and it contains an encoding declaration then the encoding declaration matches the *charset* attribute of the *Content-Type* MIME header.	MUST	
2.3	All ebXML extension elements included with the SOAP Envelope, Header and Body are namespace qualified to: "http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd"	MUST	
2.3.2	The SOAP *Envelope* element includes the XMLSchema-instance namespace qualified *schemaLocation* attribute indicating the extended ebXML envelope schema "http://www.oasis-open.org/committees/ebxml-	RECOMMENDED	

Section	Requirement	Level	GoXML-MS
	msg/schema/envelope.xsd"		
2.3.2	The SOAP *Header* and *Body* attributes both include a XMLSchema-instance namespace qualified *schemaLocation* attribute indicating the extended ebXML Msg Header schema "http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd"	RECOMMENDED	
2.3.3	The SOAP *Header* element is namespace qualified as per the SOAP namespace declaration in the SOAP Envelope element.	MUST	
2.3.4	The SOAP *Body* element is namespace qualified as per the SOAP namespace declaration in the SOAP *Envelope* element.	MUST	
2.3.5.1	The SOAP Header element always contains one ebXML MessageHeader element.	REQUIRED	
2.3.6	Any foreign namespace qualified elements present within the ebXML extension elements must be namespace qualified with a namespace that is not "http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-2_0.xsd".	MUST	
2.3.7	An XML ID attribute is supplied for each ebXML element to assist with such tasks as specifying elements included in a digital signature.	MAY	
2.3.8	The ebXML *MessageHeader* element contains a *version* attribute with a value of "2.0". If a message is received with a version that is not recognised by the MSH then the MSH responds with an error.	MUST	
2.3.9	All ebXML extensions of the SOAP *Header* element (*MessageHeader*, *SyncReply*, *MessageOrder*, ...) contain the *mustUnderstand* attribute namespace qualified to the SOAP namespace (http://schemas.xmlsoap.org/soap/envelope).	MUST	
2.3.9	Any SOAP *Header* extension with a *mustUnderstand* attribute set to "1" that is not understood by the MSH will result in the message being rejected.	MUST	
3.1.1	The *From* and *To* elements are always present as the children of the *MessageHeader* element and identify the parties that originated/are-intended-to-receive the message respectively.	MUST	
3.1.1	The *From* and *To* elements each contain one or more *PartyId* elements and zero or one *Role* element.	MUST	
3.1.1	When the *From* or *To* elements contain multiple PartyId elements the party id values are treated as identifying the same organisation.	MUST	
3.1.1	If the *Role* element is present it always follows the last *PartyId* element	MUST	
3.1.1.1	If content of the *PartyId* element is not a URI and the element does not contain a *type* attribute then the MSH responds with an error (Inconsistent/Error).	SHOULD	
3.1.2	If the receiving MSH cannot resolve the value of the CPAId element it responds with an error (NotRecognized/Error)	MUST	

Section	***Requirement***	***Level***	***GoXML-MS***
3.1.2	If the receiving MSH detects an inconsistency between an incoming message and the relevant CPA then it responds with an error (Inconsistent/Error)	MUST	
3.1.3	Generated *ConversationId* element values are strings that are unique within the content of a the specified CPAId.	MUST	
3.1.3	The generated *ConversationId* will be present in all messages pertaining to the given conversation.	MUST	
3.1.4.1	If the value of the *Service* element is not a URI and the element's *type* attribute is not present then the MSH responds with an error (Inconsistent/Error)	MUST	
3.1.5	If the receiving MSH does not recognise both the Service and Action values of an incoming message then it responds with an error (NotRecognised/Error).	MUST	
3.1.6.1	The *MessageId* element within the *MessageData* element is always present and its value is generated as a globally unique identifier.	MUST	
3.1.6.2	The *Timestamp* element within the MessageData element is always present and indicates the time at which the message header was generated. The element value is expressed according to the dateTime XMLSchema datatype in the UTC timezone.	MUST	
3.1.6.3	The *RefToMessageId* element within the *MessageData* element, if present, contains the MessageId value of an earlier ebXML Message to which this message relates.	MUST	
3.1.6.3	If there is no earlier related message then the *RefToMessageId* element is never present.	MUST	
3.1.6.3	For an error message the *RefToMessageId* element is always present with a value indicating the message in error.	MUST	
3.1.6.4	If a TimeToLive element is present within the MessageData element then it represents the time by which the message should be delivered to the To Party MSH. The element value is expressed according to the XML Schema dateTime datatype in the UTC timezone.	MUST	
3.1.6.4	If the MSH receives a message for which it is the To Party MSH and the TimeToLive is greater than the time of the internal clock (adjusted for UTC) then an error message is returned to the To Party MSH (TimeToLiveExpired/Error).	MUST	
3.1.7	If the DuplicateElimination element is present within the MessageHeader then the MSH (receiving) elimates duplicate messages.	MUST	
3.1.7	If the relevant CPA specifies a diplicateElimination of *never* then the DuplicateElimination element must not be present.	MUST	
3.1.8	If the Description element is present then the language of the free form description is specified via the *xml:lang* attribute.	MUST	
3.2	No payload/application data is present in the SOAP Body / ebXML Manifest elements.	RECOMMENDED	
3.2.2	If the *xlink:href* element of a Manifest/Reference element specifies a URI via content id ("cid:") then	MUST	

Section	***Requirement***	***Level***	***GoXML-MS***
	a MIME part with a matching content-id exists in the payload container of the message. If there is no matching payload then an error message is directed to the From Party MSH (MimeProblem/Error).		
3.2.2	If the xlink:href element of a Manifest/Reference element specifies a URI that is not a content id (not "cid:"), and that cannot be resolved, then the receiving MSH reports an error to the from Party MSH (MimeProblem/Error)	MAY	
3.2.2	If a MIME payload part exists that is not referenced by a Manifest/Reference element then it is discarded.	SHOULD	
4.1	Zero or more *Signature* elements belonging to the XML Signature namespace may be included to digitally sign a message.	MUST	
4.1	If there is more than one *Signature* element within the SOAP Header then it is the first signature that represents digital signing of the message by the From Party MSH.	MUST	
4.1.2.1	The MSH offers security utility services with regard to message payload items according to the Document Exchange section of the relevant CPA.	MAY	
4.1.2.1	The MSH applies security measures to an ebXML Message (as a whole) based upon the Transport section of the relevent CPA.	MUST	
4.1.3	Digital signatures are generated and rendered according the XML Signature specification (XMLDSIG). The SignedInfo element has a CanonicalizationMethod, SignatureMethod and one or more Reference elements.	MUST	
4.1.3	The canonicalization method applied to the data to be signed is Algorithm="http://www.w3.org/TR/2001/REC-xml-c14n-20010315"	RECOMMENDED	
4.1.3	The *SignatureMethod* element is present and has an *Algorithm* attribute.	MUST	
4.1.3	The value of the Algorithm attribute is Algorithm="http://www.w3.org/2000/09/xmldsig#dsa-sha1"	RECOMMENDED	
4.1.3	The MSH supports the signature algorithm DSA-SHA1.	MUST	
4.1.3	The XML Signature *Reference* element has a URI attribute value of "" (indicating that the signature is to be applied to the document that contains the Signature element).	SHALL	
4.1.3	The MSH supports the optional addition of the informative *Type* attribute with value "http://www.w3.org/2000/09/xmldsig#Object" on the XML Signature *Reference* element.	SHALL	
4.1.3	The XML Signature *Reference* element includes a child *Transforms* element which in turn includes a first *Transform* element with an Algorithm attribute of value "http://www.w3.org/2000/09/xmldsig#enveloped-signature".	SHALL	
4.1.3	A second *Transform* element exists with the requisite XPath element exluding all elements with SOAP actor attributes targetting the nextMSH or	MUST	

Section	***Requirement***	***Level***	***GoXML-MS***
	next SOAP node.		
4.1.3	The MSH with acting in an intermediary role does not change, format or in any way modify any element not tagetted at that intermediary MSH. The MSH does not add or delete white space.	MUST	
4.1.3	The last *Transfom* element has an Algorithm attribute with a value of "http://www.w3.org/TR/2001/REC-xml-c14n-20010315".	SHOULD	
4.1.3	Any payload data requiring digital signature is identified by an XML Signature *Reference* element that has a URI attribute resolving to the location of that data.	MUST	
4.1.3	The URI attribute of an XML Signature *Reference* element identifying a payload object matches the xlink:href URI value present in the Manifest/Reference element corresponding to that object.	RECOMMENDED	
4.1.3	Signature generation takes place before any transfer encoding (eg base64) is applied to the SOAP Envelope or payload MIME parts.	MUST	
4.1.3.2	A digitally signed messaged may be acknowledged with a digitally signed acknowledgement. Any such acknowledgement message contains an XML Signature *Reference* element list corresponding to the *Reference* elements contained in the original message.	MUST	
4.1.4.3	The communication channel used to transport the ebXML message can be used to provide uni or bi-directional party authentication (eg TLS over TCP/IP).	MAY	
4.1.4.4	The communication channel used to transport the ebXML message can be used to provide data integrity of the message content (eg TLS over TCP/IP).	MAY	
4.1.4.5	If signature and encryption of a message component is requested of the MSH, signing takes place prior to encryption.	MUST	
4.1.4.6	The communication channel used to transport the ebXML message can be used to provide data confidentiality for the message content (eg TLS over TCP/IP).	MUST	
4.1.4.8	The source of an ebXML message can be authorised by using a secure network protocol for bilateral authentication of certificates prior to establishing a session (eg TLS over TCP/IP).	MAY	
4.2	When sending messages the MSH can accept and process SOAP Fault values from a downstream SOAP processor.	MUST	
4.2	If an MSH returns a SOAP Fault message to the sender of a SOAP message then this returned message conforms to the SOAP specification guidelines for SOAP Fault values.	MUST	
4.2	When an ebXML Message is reporting an error with a *highestSeverity* value of 'Warning' it is not reported or returned as a SOAP Fault.	SHALL NOT	
4.2.2	Errors associated with data communications protocols are detected and reported using the standard mechanisms supported by that protocol	SHALL NOT	

Section	***Requirement***	***Level***	***GoXML-MS***
	and do not use ebXML reporting mechanisms.		
4.2.3	The ErrorList extension element of the SOAP Header element is never present if there are no errors to be reported.	MUST NOT	
4.2.3.1	The highestSeverity attribute contains the highest severity of any Error elements present.	MUST	
4.2.3.2.2	The *codeContext* attribute of an *Error* element is a always a URI.	MUST	
4.2.3.2.2	The namespace/scheme specified by *codeContext* for identifying errorCodes is the default value of **urn:oasis:names:tc:ebxml-msg:service:errors**.	RECOMMENDED	
4.2.3.2.3	For each *Error* element the *errorCode* attribute is present and indicates the nature of the error.	REQUIRED	
4.2.3.2.4	For each *Error* element the *severity* attribute has the value of Warning or Error indicating the severity of the error.	REQUIRED	
4.2.3.2.5	If an error exists in an ebXML element and the containing document is well-formed then the *location* attribute of the Error element is an XPointer the erroneous element.	MUST	
4.2.3.2.5	If an error exists in a payload MIME part then the *location* attribute of the Error element contains the content-id (via "cid:") of the erroroneous MIME part.	MUST	
4.2.3.4	The "Short Description" text for each error code provided by the Message Service Specification does not appear in any relevant Error element.	MUST NOT	
4.2.4.1	When an MSH detects an error in a message the error is reported to the MSG that sent the message in error.	RECOMMENDED	
4.2.4.1	If the error reporting location cannot be found or the message in error has an ErrorList element with highestSeverity set to Error the error should be: * Logged; * Resolved by other means; and, * No further action is taken.	RECOMMENDED	
4.2.4.2	If the ErrorURI is implied in the relevant CPA then this is used as the Error Reporting Location.	SHOULD	
4.2.4.2	If the ErrorURI is unavailable in the relevant CPA then a URI specified in the *From Party* of the message is used as the Error Reporting Location.	MAY	
4.2.4.3	If an *ErrorList* is included as part of a message being sent as a result of processing an earlier message and the *Service* and *Action* values are set as specified by the service enacted, then highestSeverity will not be Error.	MUST	
4.2.4.3	If an ErrorList is included as part of an independent message then the values of Service and Action are: Service - urn:oasis:names:tc:ebxml-msg:service Action - MessageError	MUST	
4.3.1	If a *SyncReply* element is present in a message received over a synchronous communications protocol then that connection is to be kept open in expectation of a response message using the same connection.	SHOULD	
4.3.1	The *SyncReply* element may not be used to	MUST NOT	

Section	Requirement	Level	GoXML-MS
	override the syncReplyMode in the CPA.		
4.3.1	If the CPA syncReplyMode is set to none and a SyncReply element is present in an inbound message then the receiving MSH issues an error (Inconsistent/Error).	SHOULD	

Part II: Additional Features

All mandatory syntax, features and behavior must be supported for any modules that the MSH implementation has chosen to implement.

Section	Requirement	Level	GoXML-MS
	Reliable Messaging Module		
6	Failure to receive an Acknowledgment message from a Receiving MSH by the sending MSH results in successive retries until an Acknowledgment is received or a predetermined number of retries is exceeded.	MAY	
6	If the predetermined number of retries is exceeded the *From Party* is notified of the probably delivery failure.	MUST	
6.1	All messages sent or received reliably are kept in persistent storage.	MUST	
6.1	After a system failure the MSH ensures that all messages in persistent storage are processed as if the system failure or interruption had not occured.	MUST	
6.1	The MessageId of any received messaged is recorded in persistent storage.	MUST	
6.1	A received message is recorded in its entirety at least until the information in the message has been passed to the application needing to process it.	RECOMMENDED	
6.1	The time at which a message is received is recorded in persistent storage.	RECOMMENDED	
6.1	Each response message is stored in its entirety in persistent storage.	RECOMMENDED	
6.3.1	An acknowledgment can be requested from a downstream MSH by inserting the AckRequested element in the SOAP Header of an outbound message.	MUST	
6.3.1	If there are two AckRequested elements in a generated message Header they do not specify the same value for their respective SOAP actor attributes.	MUST	
6.3.1	At most one AckRequested element may be targeted at the actor URI for the *Next MSH* and at most one AckRequested element may be targeted at the actor URI for the *To Party MSH* in a given message.	MUST	
6.3.1.1	The AckRequested element can only be targeted at either the *Next MSH* or the *To Party MSH*.	MUST	
6.3.1.2	The From Party explicitly requests that an acknowledgment message be either signed or unsigned by setting the *signed* attribute of the *AckRequested* element appropriately.	MUST	
6.3.1.2	Before setting the value of the signed attribute on	SHOULD	

Section	***Requirement***	***Level***	***GoXML-MS***
	an outgoing message the *Sending MSH* checks if the *Receiving MSH* supports acknowledgment messages of the type requested.		
6.3.1.2	If an acknowledgment message of the type requested can be produced then a message containing an *Acknowlegment* element is returned the to the *Sending MSH*.	MUST	
6.3.1.2	If an acknowledgment message of the type requested cannot be produced then an error is reported to the Sending MSH. The error is Inconsistent/Error if the request in inconsistent with the relevant CPA, or Inconsistent/Warning if the mode is not supported.	MUST	
6.3.1.3	If an acknowledgment message is requested of the MSH node acting in the role of *To Party* then the Acknowledgment element generated is targeted to the MSH node acting in the role of *From Party*.	MUST	
6.3.1.4	A pure acknowledgment message (with no payload) will not include an AckRequested element.	MUST	
6.3.1.4	An error message will not include an AckRequested element.	MUST	
6.3.2.1	The SOAP actor attribute of the Acknowledgment element is not specified or has a value corresponding to the AckRequested element of the message being acknowledged.	SHALL	
6.3.2.2	The *Timestamp* element is present within any Acknowledgment element. The value is in XML Schema dateTime format in the UTC timezone and represents the time at which the message being acknowledged was received by the MSH generating the ackowledgement message.	REQUIRED	
6.3.2.3	The *RefToMessageId* element contains the MessageId of the message who delivery is being acknowledged.	REQUIRED	
6.3.2.4	If present, the From element in Acknowledgment lement identifies the party generating the acknowledgment message.	MUST	
6.3.2.4	If the *From* element is omitted from the *Acknowledgment* element then the value of the *From* element in the *MessageHeader* is used to identify the party sending the acknowledgment.	MUST	