OASIS ebXML Messaging Services

Working Draft 08, 18 January 2006

Document identifier:
wd-ebMS3-08

Location:
http://www.oasis-open.org/committees/documents.php?wg_abbrev=ebxml-msg

Editor:
Matthew MacKenzie, Adobe Systems Incorporated mattm@adobe.com>
Jeff Turpin, Cyclone Commerce <jturpin@cyclonecommerce.com>
Pete Wenzel, Sun Microsystems <pete.wenzel@sun.com>

Contributors:
Doug Bunting, Sun Microsystems <doug.bunting@sun.com>
Jacques Durand, Fujitsu Software <jdurand@us.fujitsu.com>
Ric Emery, Cyclone Commerce <remery@cyclonecommerce.com>
Kazunori Iwasa, Fujitsu Limited <kiwasa@jp.fujitsu.com>
Hamid Ben Malek, Fujitsu Software <hmalek@us.fujitsu.com>
Dale Moberg, Cyclone Commerce <dmoberg@cyclonecommerce.com>
Sacha Schlegel, Cyclone Commerce <sschlegel@cyclonecommerce.com>

Abstract:
This specification focuses on defining a communications-protocol neutral method for exchanging electronic business messages. It defines specific enveloping constructs supporting reliable, secure delivery of business information. Furthermore, the specification defines a flexible enveloping technique, permitting messages to contain payloads of any format type. This versatility ensures legacy electronic business systems employing traditional syntaxes (i.e. UN/EDIFACT, ASC X12, or HL7) can leverage the advantages of the ebXML infrastructure along with users of emerging technologies.

Status:
This draft reflects the TC's consensus on the general feature set expressed herein; however, the technical details are subject to change.
This document was last revised or approved by the TC on the above date. The level of approval is also listed above. Check the current location noted above for possible later revisions of this document. This document is updated periodically on no particular schedule.
Technical Committee members should send comments on this specification to the ebxmlmsg@lists.oasis-open.org list. Others should use the comment form at http://www.oasisopen.org/committees/comments/form.php?wg_abbrev=ebxml-msg.
For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the

Intellectual Property Rights section of the OASIS ebXML Messaging Services TC web page (http://www.oasis-open.org/committees/ebxml-msg/ipr.php).

Appendix A. SOAP Format and Bindings

This appendix specifies the SOAP format (SOAP versions, packaging of attachments and/or binary data) used in ebMS-3, as well as how this SOAP format is transported over HTTP and SMTP.

ebMS-3 does not require the usage of SOAP-1.1 and/or SwA (SOAP-1.1 With Attachments). We consider the attachments specification of SwA as being orthogonal to the SOAP version. In other words, attachments could well be used for SOAP 1.2 in the same way they are used for SOAP 1.1. Similarly, we also consider MTOM being orthogonal to the SOAP version.

A conformant implementation of ebMS-3 may well choose to use SOAP-1.2 instead of SOAP-1.1. When using binary data and/or attachments, two alternatives are available, namely SwA and MTOM. Since SwA and MTOM are orthogonal to the SOAP version, there are four possibilities:

- An implementation of ebMS-3 may choose SOAP-1.1 with Attachments
- An implementation of ebMS-3 may choose SOAP-1.1 with MTOM
- An implementation of ebMS-3 may choose SOAP-1.2 with Attachments
- An implementation of ebMS-3 may choose SOAP-1.2 with MTOM

Both SwA and MTOM use the same attachment/encapsulation mechanism, namely the multipart/related MIME encapsulation. This encapsulation is independent of the version of SOAP being used (in fact it can encapsulate any XML document, not just SOAP), and also independent of the transport protocol (the encapsulation could be transported via HTTP, SMTP, etc…).

Since there are four possibilities, how could an MSH choose which one to use? Each of the above cases has its own merits. The following is merely a suggestion (not even a recommendation) on which SOAP format to use:

- Use SOAP 1.1 with Attachments if your partners do not use SOAP 1.2 yet and web services are not used as the primary endpoints of your deployment.
- Use SOAP 1.1 with MTOM if your partners do not use SOAP 1.2 yet and one of your endpoints is a Web Service. Also, if at least one of the payloads is an XML document (or XML fragment) that needs to contain or point to a binary data, using MTOM is a good choice since the overhead of encoding/decoding to base64 is eliminated, plus the benefit of having a well structured XML infoset with binary data that could be defined in WSDL.
- Use SOAP 1.2 with attachments if your partners can process SOAP 1.2 and the payload being transported in the messages is not intended to be directly consumed by web services as endpoints.
- Use SOAP 1.2 with MTOM if your partners can process SOAP 1.2, web services are deployed as endpoints and/or that your payload consists of XML fragments that need to contain binary data. Also, if large binary data are being exchanged, using MTOM will eliminate the overhead of encoding/decoding to/from base64 since the binary data would be transported as attachments in its raw binary form.

1.1 Using SwA

The following example shows an ebMS-3 message using SOAP 1.1 with attachment. The ebMS-3 message in this example contains two payloads:

1. The first payload is the picture of a car. This picture is in binary form as an attachment with a Content-ID equal to “car-photo”.

2. The second payload is an XML fragment within the SOAP body. This XML fragment has id attribute equal to “carData”

The XML fragment in the SOAP body contains a reference to another binary data, namely the picture of the car owner):

```
Content-Type: Multipart/Related; boundary=MIME_boundary; type=text/xml;
        start="<car-data@toyoya.com>"

--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <car-data@toyoya.com>

<?xml version='1.0' ?>
<S11:Envelope xmlns:S11="http://schemas.xmlsoap.org/soap/envelope/"
   xmlns:eb="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-3_0.xsd">
  <S11:Header>
     <eb:Messaging eb:version="3.0" S11:mustUnderstand="1">
         …
           <eb:PayloadInfo>
              <eb:PartInfo href="cid:car-photo" />
              <eb:PartInfo href="#carData" />
           </eb:PayloadInfo>
     </eb:Messaging>
  </S11:Header>

  <S11:Body>
     <t:Data id="carData" xmlns:t="http://toyota.com">
        <t:Mileage>20000</t:Mileage>
        <t:OwnerPicture href="cid:picture-of-owner"/>
     </t:Data>
  </S11:Body>
</S11:Envelope>

--MIME_boundary
Content-Type: image/tiff
Content-Transfer-Encoding: binary
Content-ID: <car-photo>

...binary TIFF image of the car...

--MIME_boundary—
Content-Type: image/tiff
Content-Transfer-Encoding: binary
Content-ID: <picture-of-owner>

...binary TIFF image of the car’s owner...
```

```
--MIME_boundary–

```

Example 1: SOAP-1.1 with Attachment

The following (Example 2) shows the same message given in example 1, except that SOAP-1.2 is being used instead of SOAP-1.1:

```
Content-Type: Multipart/Related; boundary=MIME_boundary; type=text/xml;
              start="<car-data@toyoya.com>"


--MIME_boundary
Content-Type: text/xml; charset=UTF-8
Content-Transfer-Encoding: 8bit
Content-ID: <car-data@toyoya.com>

<?xml version='1.0' ?>
<S12:Envelope xmlns:S12="http://www.w3.org/2003/05/soap-envelope"
    xmlns:eb="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-3_0.xsd">
  <S12:Header>
     <eb:Messaging eb:version="3.0" S12:mustUnderstand="true">
         …
           <eb:PayloadInfo>
              <eb:PartInfo href="cid:car-photo" />
              <eb:PartInfo href="#carData" />
           </eb:PayloadInfo>
     </eb:Messaging>
  </S12:Header>

  <S12:Body>
    <t:Data id="carData" xmlns:t="http://toyota.com">
        <t:Mileage>20000</t:Mileage>
        <t:OwnerPicture href="cid:picture-of-owner"/>
     </t:Data>
  </S12:Body>
</S12:Envelope>

--MIME_boundary
Content-Type: image/tiff
Content-Transfer-Encoding: binary
Content-ID: <car-photo>

...binary TIFF image of the car...

--MIME_boundary–
Content-Type: image/tiff
Content-Transfer-Encoding: binary
Content-ID: <picture-of-owner>

...binary TIFF image of the car’s owner...
--MIME_boundary–
```

Example 2: SOAP-1.2 with Attachments over HTTP

What were the differences between Example 1 and Example 2 (SOAP 1.1/SOAP 1.2 with attachments)? The differences are the following:

- In SOAP 1.1, the namespace of the SOAP elements (Envelope, Header, and Body) is http://schemas.xmlsoap.org/soap/envelope/ versus the namespace http://www.w3.org/2003/05/soap-envelope for SOAP 1.2
- In SOAP 1.1, the attribute mustUnderstand takes 0 or 1 as values, whereas in SOAP 1.2, the values for the attribute mustUnderstand are true and false.

That’s it. Another difference between SOAP 1.1 and SOAP 1.2 would be in the SOAPAction header. When using HTTP as the transport protocol, there will be an HTTP header called SOAPAction if SOAP 1.1 is being used. If SOAP 1.2 is used, instead of the SOAPAction header there will be an action parameter, as illustrated in the following listings:

```
SOAPAction: leasing
Content-Type: Multipart/Related; boundary=MIME_boundary; ype=text/xml;
               start="<car-data@toyoya.com>"
```

HTTP headers when using SOAP 1.1 with attachments

```
SOAPAction: leasing
Content-Type: Multipart/Related; boundary=MIME_boundary; ype=text/xml;
               start="<car-data@toyoya.com>"; action=leasing
```

HTTP headers when using SOAP 1.2 with attachments

When using SMTP transport, the only additional requirement is that the Mime-Version header must be present (among other SMTP related headers such as To, From, Date, etc…). The following listings show the headers for both SOAP 1.1 and SOAP 1.2 over SMTP:

```
From: hamid@us.fujitsu.com
To: leasing-office@toyota.com
Date: Mon, 23 Jan 2006 17:33:00 CST
Mime-Version: 1.0
SOAPAction: leasing
Content-Type: Multipart/Related; boundary=MIME_boundary; ype=text/xml;
               start="<car-data@toyoya.com>"
```

SMTP headers when using SOAP 1.1 with attachments

```
From: hamid@us.fujitsu.com
To: leasing-office@toyota.com
Date: Mon, 23 Jan 2006 17:33:00 CST
Mime-Version: 1.0
Content-Type: Multipart/Related; boundary=MIME_boundary; ype=text/xml;
               start="<car-data@toyoya.com>"; action=leasing
```

SMTP headers when using SOAP 1.2 with attachments

1.2 Using MTOM

MTOM is not a competitor to SwA – MTOM was designed to fix the issues with SwA, enabling it to work within the composable model of the Advanced Web services specifications. MTOM messages are actually valid SwA messages.

SwA defines a way for binding attachments to a SOAP envelope using the multipart/related MIME type - this is the same attachment/encapsulation mechanism used for e-mail. MIME is inefficient because it uses text strings to delineate boundaries between parts. Consumers must scan the entire message to find the string value used to delineate a boundary. MIME cannot be represented as an XML Infoset – this effectively breaks the web services model since attachments cannot be secured using WS-Security. The DIME specification was created to address performance issues when processing MIME attachments. DIME avoided having to scan the entire message to locate boundaries because the length of the attached files was encoded in the message header, enabling large attachments to be processed in “chunks”. While DIME provided a more efficient processing model it still didn’t provide an infoset model for the message and attachment. MTOM provides a compromise between the MIME model and the Web services model (an infoset representation is available). MTOM messages are valid SwA messages, lowering the cost of supporting MTOM for existing SWA implementations. MTOM attachments are streamed as binary data within a MIME message part, making it fairly easy to pass MTOM attachments to SwA or receive SwA attachments into an MTOM implementation.

More formally speaking, MTOM is actually a collection of three W3C recommendations:

- Resource Representation SOAP Header Block (http://www.w3.org/TR/soap12-rep)
- XML-binary Optimized Packaging (http://www.w3.org/TR/xop10/)
- SOAP Message Transmission Optimization (http://www.w3.org/TR/soap12-mtom)

It is also interesting to notice that MTOM is an abstract layer that is above Mime encapsulation and even above SOAP. The second recommendation (XML-binary Optimized Packaging) is about serialization/deserialization of XOP packages (XML documents mixed with binary data). Mime multipart/related is only one way of encapsulating XOP packages (in other words, the recommendation leaves it open for other possible means of encapsulation). The third recommendation (SOAP Message Transmission Optimization) is simply a concrete XOP encapsulation packages for SOAP over HTTP, using Mime multipart/related mechanism. This is just to say that MTOM is indeed orthogonal to SOAP versions and can be used for SOAP 1.1 too.

The following listing is an example of an ebMS-3 message using SOAP-1.2 with MTOM. The ebMS-3 message in this example contains one payload which is the XML fragment in the SOAP body refered to it by href=“#myPhoto”. The XML fragment in the SOAP body contains references to binary data which is attached in raw form:

```
Content-Type: Multipart/Related;boundary=MIME_boundary;
    type="application/xop+xml";
    start="<mymessage.xml@example.org>";
    startinfo="application/soap+xml; action=\"ProcessData\""

--MIME_boundary
Content-Type: application/xop+xml; charset=UTF-8;
    type="application/soap+xml; action=\"ProcessData\""
Content-Transfer-Encoding: 8bit
Content-ID: <mymessage.xml@example.org>

<S12:Envelope xmlns:S12='http://www.w3.org/2003/05/soap-envelope'
    xmlns:xmlmime='http://www.w3.org/2004/11/xmlmime'
    xmlns:eb="http://www.oasis-open.org/committees/ebxml-msg/schema/msg-header-3_0.xsd"
    xmlns:xop="http://www.w3.org/2004/08/xop/include">

  <S12:Header>
     <eb:Messaging eb:version="3.0" S12:mustUnderstand="true">
         …
           <eb:PayloadInfo>
               <eb:PartInfo href="#myPhoto" />
           </eb:PayloadInfo>
     </eb:Messaging>
  </S12:Header>

  <S12:Body>

    <m:data id="myPhoto" xmlns:m='http://example.org/stuff'>
      <m:photo xmlmime:contentType='image/png'>
          <xop:Include href='cid:http://example.org/me.png'/>
      </m:photo>

      <m:sig xmlmime:contentType='application/pkcs7-signature'>
          <xop:Include href='cid:http://example.org/my.hsh'/>
      </m:sig>
    </m:data>

  </S12:Body>
</S12:Envelope>

--MIME_boundary
Content-Type: image/png
Content-Transfer-Encoding: binary
Content-ID: <http://example.org/me.png>

// binary octets for png

--MIME_boundary
Content-Type: application/pkcs7-signature
Content-Transfer-Encoding: binary
Content-ID: <http://example.org/my.hsh>

// binary octets for signature

--MIME_boundary--
```

From the sample listing above, we can see the differences in the Mime headers between “SOAP-1.1/1.2 with attachments” and “SOAP-1.1/1.2 with MTOM” as the following:

- In SOAP-1.1/1.2 with attachments, the type parameter of the Content-Type header of the package is text/xml, whereas in SOAP-1.1/1.2 with MTOM the type parameter has a value of “application/xop+xml”.
- In SOAP-1.1/1.2 with MTOM, there is a new parameter “start-info” whose value must be the same value for the type parameter of the SOAP part, and this must be “application/xop+xml” instead of “text/xml” in the case of SOAP-1.1/1.2 with attachments.

The above described changes in the Mime headers are highlighted in the sample listing above.

The following short listings illustrate the difference with “SOAP 1.1 with MTOM” and “SOAP 1.2 with MTOM”:

```
SOAPAction: ProcessData

Content-Type: Multipart/Related;boundary=MIME_boundary; type="application/xop+xml";
    start="<mymessage.xml@example.org>"; startinfo="application/soap+xml

--MIME_boundary
Content-Type: application/xop+xml; charset=UTF-8; type="application/soap+xml
Content-Transfer-Encoding: 8bit
Content-ID: mymessage.xml@example.org

<S11:Envelope …
```

SOAP 1.1 with MTOM : action parameter is absent (SOAPAction header used instead)

When using SMTP as a transport protocol for SOAP-1.1/1.2 with MTOM, nothing really changes besides the addition of SMTP related headers (such as From, To, Date, etc…)