

# 1    ebXML Transport, Routing & Packaging 2    Messaging Service Specification

## 3    Working Draft 10-August-2000

### 4    This version:

5       ebXML Messaging Service Specification v0-1.doc

### 6    Latest version:

7       N/A

### 8    Previous version:

9       ebXML Transport, Routing & Packaging Message Envelope Specification 0.6

10      ebXML Message Header Specification v0-63.doc

### 11   Editor:

12      Dick Brooks [dick@8760.com](mailto:dick@8760.com)

13      David Burdett <[david.burdett@commerceone.com](mailto:david.burdett@commerceone.com)>

### 14   Authors:

15      Dick Brooks <[dick@8760.com](mailto:dick@8760.com)>

16      Nick Kassem [nick.kassem@sun.com](mailto:nick.kassem@sun.com)

17      David Burdett <[david.burdett@commerceone.com](mailto:david.burdett@commerceone.com)>

18      John Ibbotson <[john\\_ibbotson@uk.ibm.com](mailto:john_ibbotson@uk.ibm.com)>

19      Christopher Ferris <[chris.ferris@sun.com](mailto:chris.ferris@sun.com)>

### 21   Contributors:

22      The members of the Transport Routing and Packaging Project Team

---

## 23   Abstract

24   This document is a draft proposal whose purpose is to solicit additional input and convey the  
25   current state of the ebXML message structure recommendations.

26   This document defines the envelope and header structure used to encapsulate data for transport  
27   between parties. Every attempt has been made to ensure that ebXML requirements as stated in  
28   the ebXML Transport, Routing and Packaging: Overview and Requirements, Version 0.96, are  
29   addressed. Adherence to industry standards, consideration of existing business-to-business  
30   practices and support for small and medium enterprises were key factors influencing the  
31   direction of this specification.

## 32   Notational Conventions

33   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",  
34   "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be  
35   interpreted as described in Key Words for Use in RFC's to Indicate Requirement Levels (RFC  
36   2119).



- 37 Terms in *Italics* are defined in the ebXML Glossary of Terms [3]. Terms listed in ***Bold Italics*** represent the element and/or attribute content of the ebXML Message Header.
- 38

## 39 **Status of this Document**

- 40 This document represents work in progress upon which no reliance should be made.



## 41 Table of Contents

42	1	Introduction .....	5
43	1.1	Purpose and Scope .....	5
44	1.1.1	Goals .....	5
45	1.2	Related Transport, Routing and Packaging Specifications .....	5
46	1.3	Specification Structure.....	6
47	1.4	General Conventions .....	6
48	2	Packaging Specification .....	7
49	2.1	ebXML Message Structure.....	7
50	2.2	Transport Envelope .....	8
51	2.3	Message Envelope Specifications.....	8
52	2.3.1	Content-Type .....	8
53	2.3.1.1	type Attribute .....	8
54	2.3.1.2	boundary Attribute .....	8
55	2.3.2	Content-Length .....	8
56	2.3.3	ebXML Message Envelope Example .....	9
57	2.4	ebXML Header Container Specifications.....	9
58	2.4.1	Content-ID .....	9
59	2.4.2	Content-Length .....	9
60	2.4.3	Content-Type .....	9
61	2.4.4	Optional Support for Signed Headers .....	10
62	2.4.5	Example of an ebXML Header container .....	10
63	2.5	Payload Container Specifications .....	10
64	2.5.1	Content-ID .....	11
65	2.5.2	Content-Length .....	11
66	2.5.3	Content-Type .....	11
67	2.5.4	Optional Support for Signed and Encrypted Payloads .....	11
68	2.5.5	Example of an ebXML Payload Container .....	11
69	2.6	Message Digest Computation .....	12
70	3	Message Header.....	12
71	3.1	Root Element.....	13
72	3.2	Manifest.....	13
73	3.2.1	DocumentReference .....	13
74	3.3	Header .....	14
75	3.3.1	From and To .....	14
76	3.3.2	TPAInfo.....	15
77	3.3.3	MessageData .....	15
78	3.3.4	ReliableMessagingInfo .....	15
79	4	Security Considerations .....	16
80	5	References .....	16
81	6	Acknowledgments .....	16
82	7	Authors' Address .....	17
83	Appendix A	Schemas and DTD Definitions .....	19
84	A.1	XML Header DTD .....	19
85	A.2	XML Header Schema Definition.....	20



86	Appendix B Examples .....	23
87	B.1 Complete Example of an ebXML Message enveloped using multipart/related	
88	Content-Type sent via HTTP POST .....	23
89	B.2 Complete Example of an ebXML Message enveloped using multipart/related	
90	Content-Type sent via SMTP .....	26
91	Appendix C Candidate Packaging Technologies and Selection Process .....	31
92	C.1 Selection Process .....	31
93	C.2 MIME .....	31
94	C.3 XML .....	31
95	C.4 Conclusion .....	32
96	Appendix D MIME Type discussion .....	33
97		



## 98    1 Introduction

99    This specification defines the ebXML message structure used to encapsulate ebXML message  
100    headers and payloads for transport between parties. No assumption or dependency is made  
101    relative to transport protocol or type of payload. The specifications contained here are both  
102    payload and transport agnostic. The main goal of this specification is to define an enveloping  
103    structure and ebXML message header elements used to encapsulate any digitally encoded  
104    payload for transport over any data communication mechanism.

### 105    1.1 Purpose and Scope

106    This document provides sufficient detail to develop software for the packaging, exchange and  
107    processing of ebXML messages. This document defines the enveloping and ebXML message  
108    header structure used to transfer ebXML messages over a data communication mechanism.

109    Software practitioners MAY use this document in combination with other ebXML specification  
110    documents when creating ebXML compliant software.

111    **NOTE: Message security, extensibility, service interface, reliability, and versioning will be  
112    addressed in future versions of this document.**

#### 113    1.1.1 Goals

114    The goals of this specification are to:

- 115    • Meet the requirements as specified by the ebXML Transport, Routing and Packaging:  
116       Overview and Requirements, Version 0.96 [1]
- 117    • Be compatible with other ebXML specifications
- 118    • Leverage existing industry standards
- 119    • Enable parties to "package" very simple to very complex payloads
- 120    • Be payload neutral
- 121    • Be transport neutral

### 122    1.2 Related Transport, Routing and Packaging Specifications

123    The following set of related specifications will be delivered in phases:

- 124    • **ebXML Messaging Service Specification** (this document) - defines the ebXML message  
125       structure and is a composite of the following documents:
  - ebXML Transport, Routing & Packaging Message Envelope Specification 0.6
  - ebXML Message Header Specification v0-63.doc
- 128    • **ebXML Reliable Messaging Specification** (under development) - defines a method to  
129       achieve robust reliable once-only delivery of ebXML message.
- 130    • **ebXML Messaging Security and Signature Specification** (under development) -  
131       describes the use of IETF/W3C XML Digital Signatures, S/MIME and PGP with ebXML  
132       messages.



- 133     • **ebXML Messaging Audit Trail Specification** (future development) - defines an audit  
134                          trail method.

135    **1.3 Specification Structure**

136    This specification is organized around the following main topics:

- 137     • **Packaging Specification** - A description of how to package an ebXML message and  
138                          associated parts. This section includes specifications for the various structures and  
139                          containers.
- 140     • **Message Headers** - A description of the elements REQUIRED.
- 141     • **ebXML Message Types** - An introduction to the different sequences in which ebXML  
142                          messages of different *Message Types* are exchanged

143    **1.4 General Conventions**

- 144     • All headers, attributes and values defined in this specification are to be handled in a case-  
145                          insensitive fashion.
- 146     • For all messages following the ebXML standard, a single message structure is defined,  
147                          regardless of message type.
- 148     • Values associated with MIME header attributes are valid in both quoted and unquoted form.  
149                          For example, the forms type="ebxml" and type=ebxml are both valid.

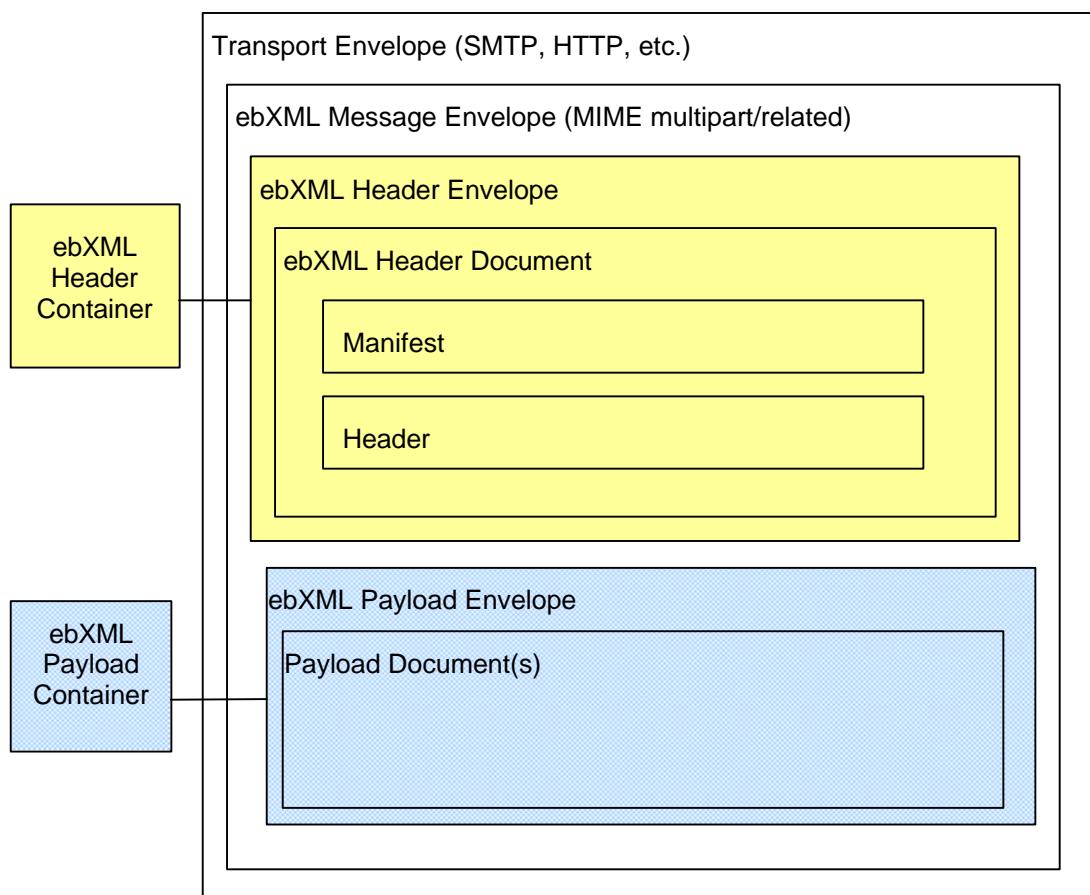


## 150 2 Packaging Specification

### 151 2.1 ebXML Message Structure

152 An ebXML message consists of:

- 153 • a conditional outer Transport Envelope, such as HTTP or SMTP,
- 154 • a transport independent Message Envelope, for example MIME multi-part/related, that  
155 contains the two main parts of the Message:
- 156 – a Header Container that is used to envelope one ebXML header document, and
- 157 – a Payload Container MUST be used to envelope the real payload of the Message when  
158 the payload is present





181 **2.2 Transport Envelope**

182 This document does NOT define any requirements affecting the structure of transport level  
183 envelopes. Existing transport protocols, such as SMTP, HTTP, FTP and others, MAY be used to  
184 send and receive ebXML compliant messages, without modification. The only requirement  
185 ebXML has on the transport envelope is the ability to identify a specific *ebXML handler* to  
186 receive incoming ebXML messages.

187 A transport envelope is only REQUIRED in those cases requiring such structures. In the case of  
188 HTTP or SMTP, transport envelopes are REQUIRED. However, a transport envelope is not  
189 needed for FTP. Implementers of software to process ebXML messages MUST be aware of  
190 transport specific requirements relative to transport envelopes.

191 **2.3 Message Envelope Specifications**

192 The message envelope is used to identify the message as an ebXML compliant structure and  
193 encapsulates the header and payload body parts. A message envelope MUST HAVE two MIME  
194 headers:

- 195     1. Content-Length  
196     2. Content-Type

197 **2.3.1 Content-Type**

198 The multi-part/related Content-Type SHALL be used for all ebXML message envelopes. See  
199 Appendix X for selection rationale.

200 The Content-Type header contains four attributes:

- 201     1. type  
202     4. boundary

203 **2.3.1.1 type Attribute**

204 The type attribute is used to identify the message envelope as an ebXML compliant structure.  
205 There is only one valid value for this attribute: "application/vnd.eb+xml". The following is an  
206 example usage of the type attribute:

```
Content-Type: multipart/related; type="application/vnd.eb+xml"
```

208 **2.3.1.2 boundary Attribute**

209 The boundary attribute is used to identify the body part separator used to identify the start and  
210 end points of each body part contained in the message. The boundary SHOULD be chosen  
211 carefully to insure that it does not occur within the content area of a body part. Example usage of  
212 the boundary attribute:

```
Content-Type: multipart/related; type="application/vnd.eb+xml"; version="0.1";  
boundary="-----8760"
```

215 **2.3.2 Content-Length**

216 The Content-Length header is a decimal value used to identify the total number of OCTETS  
217 contained in all constituent message body parts, including body part boundaries. Example:

```
Content-Length: 9841
```



219 **2.3.3 ebXML Message Envelope Example**

220 An example of a ebXML compliant message envelope appears as follows:

221 Content-Type: multipart/related; type=" application/vnd.eb+xml "boundary="-----8760"  
222 Content-Length: 9841

223 **2.4 ebXML Header Container Specifications**

224 The ebXML Header Container is a MIME body part used to encapsulate an ebXML header  
225 document. The ebXML header document is described in section 3 of this document. There  
226 MUST be one ebXML header document associated with every ebXML Message. The ebXML  
227 Header Container consists of a MIME Header portion (referred to as the ebXML Header  
228 envelope) and a content portion.

229 The ebXML Header envelope, consists of three MIME headers:

- 230     1. Content-ID  
231     2. Content-Length  
232     3. Content-Type

233 The ebXML header document within the content portion of the container MAY be enhanced  
234 during transport, provided it has not been digitally signed. Any change in the size of the ebXML  
235 header document MUST be reflected in Content-Length attribute of the ebXML Message  
236 Envelope and ebXML Header envelope.

237 **2.4.1 Content-ID**

238 The Content-ID MIME header identifies this instance of an ebXML message header body part.  
239 The value for Content-ID SHOULD be a unique identifier, in accordance with RFC 2045. An  
240 example usage follows:

241 Content-ID: ebxmlheader-com-8760-2000-0722-161201-123456789

242 **2.4.2 Content-Length**

243 The Content-Length header is a decimal value used to identify the total number of OCTETS  
244 contained in all constituent message body parts, including body part boundaries. Example:

245 Content-Length: 4208

246 **2.4.3 Content-Type**

247 The Content-Type for an ebXML header is identified with the value "application/vnd.eb+xml".  
248 Content-Type SHALL contain two attributes: **version** and **charset**. The charset attribute shall  
249 agree with the value passed in the **encoded** attribute.

- 250     • **version** – a value indicating the actual version of the ebXML message structures . There  
251       are currently two valid values for version:  
252           1. "0" indicating a version-less message; all ebXML implementations MUST  
253           support version-less messages.  
254           2. "0.1" indicating the current version of ebXML message structures.



- 255     • **charset** - a value used to identify the character set used to create the message. The list  
256       of valid values can be found at <http://www.iana.org/>. The default charset value is "iso-  
257       8895-1".

258     An example of this Content-Type is:

259     Content-Type: application/vnd.eb+xml; version="1.0"; charset = "UTF-8"

#### 260     **2.4.4 Optional Support for Signed Headers**

261     Implementers are free to support digitally signed ebXML header documents. Digitally signed  
262     ebXML headers MUST be identified with the appropriate Content-Type and structure-appropriate  
263     for the cryptographic tool used. In the case of S/MIME, the Content-Type MUST contain the  
264     correct value and attributes as specified in RFC 2633. In the case of *OpenPGP*, the Content-  
265     Type MUST contain the correct values and attributes specified in MIME Security with PGP (RFC  
266     2015).

267     Implementers MUST follow the guidelines specified in RFC 2633 and RFC 2015 for creating and  
268     processing digitally signed objects.

269     The XML Signature Syntax and Processing [4] MUST be followed when implementers use XML  
270     Digital Signatures.

#### 271     **2.4.5 Example of an ebXML Header container**

272     The following represents an example of an ebXML Header envelope and ebXML Header  
273     document:

274	Content-ID: ebxmlheader-123	-----	ebXML Header Envelope	ebXML Header Container
275	Content-Length: 2048	-----		
276	Content-Type: application/vnd.eb+xml	-----		
277	<ebXMLHeaderDocument>	-----	ebXML header Document	
278	<MessageHeader>.....	-----		
279	</MessageHeader>	-----		
280	</ebXMLHeaderDocument>	-----		
281				
282				

283     A complete example of the ebXML Header envelope is presented in Appendix X.

### 284     **2.5 Payload Container Specifications**

285     A single Payload Container of an ebXML message MUST be used to envelop the payload(s)  
286     when one or more payloads is/are present. The ebXML header document contains a *Message*  
287     *Manifest* that identifies whether a Payload Container is present or not. If the *Message Manifest* of  
288     the ebXML header does not contain any entries, the ebXML Payload Container will not be  
289     present in the ebXML Message. However, if the *Message Manifest* of the ebXML header  
290     indicates that a payload is present, it will consist of a MIME header portion (referred to as the  
291     *ebXML Payload envelope*) and a content portion.

292     The ebXML Payload envelope, consists of three MIME headers:

- 293       1. Content-ID
- 294       2. Content-Length
- 295       3. Content-Type

296



297 ebXML makes no provision nor limits in any way the structure or content of payloads. Payloads  
298 MAY contain simple plain text object or complex nested multipart objects. This is the  
299 implementer's decision.

300 **2.5.1 Content-ID**

301 The Content-ID MIME header identifies this instance of an ebXML is used to uniquely identify an  
302 instance of an ebXML message payload body part. The value for Content-ID SHOULD be a  
303 unique identifier, in accordance with Multipurpose Internet Mail Extensions (RFC 2045). An  
304 example usage follows:

305 Content-ID: ebxmlpayload-com-8760-2000-0722-161201-123456789

306 **2.5.2 Content-Length**

307 The Content-Length header is a decimal value used to identify the total number of OCTETS  
308 contained in the content portion of the Payload Container. Example:

309 Content-Length: 5012

310 **2.5.3 Content-Type**

311 The Content-Type for an ebXML header is determined by the implementer and is used to identify  
312 the type of data contained in the content portion of the Payload Container.

313 Content-Type: application/vnd.eb+xml

314 **2.5.4 Optional Support for Signed and Encrypted Payloads**

315 Implementers are free to support digitally signed ebXML payloads. Digitally signed ebXML  
316 payloads MUST be identified with the appropriate Content-Type and structure-appropriate for the  
317 cryptographic tool used. In the case of S/MIME, the Content-Type MUST contain the correct  
318 value and attributes as specified in RFC 2633. In the case of OpenPGP, the Content-Type  
319 MUST contain the correct values and attributes specified in MIME Security with PGP (RFC  
320 2015).

321 Implementers MUST follow the guidelines specified in RFC 2633 and RFC 2015 for creating and  
322 processing digitally signed objects.

323 The XML Signature Syntax and Processing [4] MUST be followed when implementers use XML  
324 Digital Signatures.

325 **2.5.5 Example of an ebXML Payload Container**

326 The following represents an example of an ebXML Payload envelope and ebXML Payload  
327 document:

328 Content-ID: ebxmlpayload-123	-----	ebXML Payload Envelope	ebXML Payload Container
329 Content-Length: 4096	-----		
330 Content-Type: application/xml	-----		
331 <Invoice>	-----	ebXML Payload	
332    <Invoicedata>.....	-----		
333    </Invoicedata>	-----		
334    </Invoice>	-----		
335			

336 A complete example of the ebXML Payload Container is presented in Appendix X.



## 337 2.6 Message Digest Computation

338 Parties wishing to ensure the integrity of data exchanged using ebXML standards MAY wish to  
339 create a *Message Digest* of the information contained in an ebXML message. In order to ensure  
340 consistent usage of a *Message Digest*, it is imperative that all parties agree on the range of data  
341 to include when performing a *Message Digest* calculation. This range of data is referred to as the  
342 *Domain of Computation*. The *Domain of Computation* defines the beginning and ending  
343 boundaries (inclusive) of the information to be processed when calculating a *Message Digest*.

344 The *Domain of Computation* begins with the first "-" octet of the first boundary following the  
345 ebXML Message Envelope. The *Domain of Computation* ends with the last "-" of the final  
346 boundary of an ebXML message. All octets between and including these begin and end points  
347 *Domain of Calculation*. Parties wishing to create a *Message Digest* of an ebXML message  
348 SHOULD apply their hash algorithms over the *Domain of Computation*, as defined.

349 In the following example, the *Domain of Computation* begins with the first "--" of "--8760" and  
350 ends with the last "--" of "--8760--". The hash algorithm used to calculate a *Message Digest*  
351 SHOULD process all octets between and including these points.

```
352 Content-Type: multipart/related; type="ebxml"; version="0.1"; charset="iso-8859-1";
353 boundary="8760"
354 Content-Length: 9841
355 --8760

356 ebXML header and payload body parts removed for brevity

357 --8760--
```

358

## 359 3 Message Header

360 The ebXML Header is a single XML document with a number of principal *Header* elements within  
361 it where each *Header* element is a separate XML element. In general, separate header elements  
362 are used where:

- 363 • different software is likely to be used to generate that header element,
- 364 • the structure of the header element might vary independently of the other header elements,  
365 or
- 366 • the data contained in the header element MAY need to be digitally signed separately from  
367 the other header elements.

368

369 Using this principle, the following header elements have been identified:

- 370 • **Message Manifest** - contains a list of references to the other parts of the Message. This  
371 includes references to the documents, which comprise the *Payload* of the Message.
- 372 • **Message Header** - contains the information REQUIRED by the recipient to process the  
373 message. The message originator creates this information to which additional information  
374 MAY be added.

375 The *Message Header* and *Message Manifest* are REQUIRED elements in every *Message*.

376 NOTE: It MAY prove necessary that a separate document MAY be REQUIRED for header  
377 elements not yet defined. Every effort will be made to preclude this possibility.



378 **3.1 Root Element**

379 The root element of the ebXML Message Header document is **ebXMLHeader**. It is comprised of  
380 three attributes and two subordinate elements.

381 The first attribute is the namespace declaration (**xmlns**) which has a REQUIRED value of  
382 “<http://www.ebxml.org/namespaces/messageHeader>”.

383 The second attribute is the **Version** attribute. This attribute is required. Its purpose is to provide  
384 for future versioning capabilities. It has a default value of '1.0'.

385 The last of the **ebXMLHeader** attributes is the **MessageType** attribute. Its purpose is to enable  
386 ebXML-aware software to distinguish between normal and transport-specific messages, such as  
387 acknowledgment and error messages. This is described in more detail in section 4 below.

388 The **MessageType** is an enumeration consisting of three possible values:

- **Normal** – an application-generated message, distinct from a message generated by the infrastructure software providing the ebXML messaging service.
- **Acknowledgment** – a ebXML Messaging Service-specific acknowledgment message.
- **Error** – an ebXML Messaging Service-specific error message.

393 There are two subordinate elements of the **ebXMLHeader** root element:

- **Manifest** – identification of the payload contents
- **Header** – routing information

396 The following is a sample **ebXMLHeader** document fragment demonstrating the overall  
397 structure:

```
398 <?xml version="1.0"?>
399 <ebXMLMessageHeader xmlns="http://www.ebxml.org/namespaces/messageHeader" 
400     Version="1.0" MessageType="Normal">
401     <Manifest>...</Manifest>
402     <Header>...</Header>
403   </ebXMLMessageHeader>
```

405 **3.2 Manifest**

406 The required **Manifest** element is a composite element consisting of zero or more  
407 **DocumentReference** elements. Each **DocumentReference** element identifies data associated  
408 with the message, whether included as part of the message, or remote resources accessible via  
409 a URL. The **Manifest** SHALL be the first subordinate element in the **ebXMLMessageHeader**. It  
410 identifies the payload document(s) contained in the ebXML message envelope. The purpose of  
411 the **Manifest** is to make it easier to directly extract a particular document associated with the  
412 Message.

413 **3.2.1 DocumentReference**

414 The **DocumentReference** element is a composite element consisting of two required  
415 subordinate elements as follows:

- **DocumentLabel** – a textual description of the document/resource referenced by:
- **DocumentId** – a URL of the Content-ID of a MIME body part, as defined in [RFC2111],  
418 representing payload data, or a remote URL to some external resource.

419 The following fragment demonstrates a typical **Manifest** for a message with a single payload  
420 MIME body part:

421



```
422 <Manifest>
423     <DocumentReference>
424         <DocumentLabel>PurchaseOrder</DocumentLabel>
425         <DocumentId>cid:0987654321</DocumentId>
426     </DocumentReference>
427 </Manifest>
428
```

### 429 3.3 Header

430  
431 The **Header** element immediately follows the **Manifest** element. It is required in all  
432 **ebXMLMessageHeader** documents. The **Header** element is a composite element comprised of  
433 the following required subordinate elements:  
434

- 435 • **From** – the logical address of the sender of the message.
- 436 • **To** – the logical address of the intended recipient of the message.
- 437 • **TPAInfo** – a composite set of information which relates to the *Trading Partner Agreement*  
438 under which the message is governed
- 439 • **MessageData** – a composite set of information which uniquely identifies the *Message*
- 440 • **ReliableMessagingInfo** - information which identifies the degree of reliability with which the  
441 message SHOULD be delivered

442  
443 The following fragment demonstrates the structure of the **Header** element of the  
444 **ebXMLMessageHeader** document:

```
445 <Header>
446     <From>...</From>
447     <To>...</To>
448     <TPAInfo>...</TPAInfo>
449     <MessageData>...</MessageData>
450     <ReliableMessagingInfo>...</ReliableMessagingInfo>
451 </Header>
452
```

#### 454 3.3.1 From and To

455 The **From** element identifies the *Party* which originated the message. It is a logical identifier,  
456 which MAY take the form of a URN. An example of this would be a DUNS number. The **From**  
457 element consists of a **PartyId** element.  
458

459 The **To** element identifies the intended recipient of the message. As with **From**, it is a logical  
460 identifier which is comprised of a **PartyId** element.  
461

462 The **PartyId** element has a single attribute; **context** and a text value. The purpose of the context  
463 attribute is to provide a context for the text value of the **PartyId** element. The following fragment  
464 demonstrates usage of the **From** and **To** elements of the **ebXMLMessageHeader**.  
465

```
466 <From>
467     <PartyId context="DUNS">12345</PartyId>
468 </From>
469 <To>
470     <PartyId context="DUNS">54321</PartyId>
471 </To>
```



473    **3.3.2 TPAInfo**

474    The **TPAInfo** element follows the **From** and **To** elements in the **Header** element structure. The  
475    **TPAInfo** element is a composite set of information which relates to the *Trading Partner*  
476    *Agreement* under which the message is governed. The **TPAInfo** element has four subordinate  
477    elements as follows:

- 478    • **TPAId** – a URI which identifies the *Trading Partner Agreement* which governs the processing  
479    of the message
- 480    • **ConversationId** – a URI which identifies the conversation instance of the *Trading Partner*  
481    *Agreement*.
- 482    • **ServiceInterface** – Identifies the Service Interface that SHOULD act on the payload in the  
483    message. It is unique within the domain of the **Party** to which the message is being sent.  
484    URN's MAY be considered suitable for the element content.
- 485    • **Action** – Identifies a process within a Service Interface, which processes the Message.  
486    **Action** SHALL be unique within the Service Interface in which it is defined.

487    The following example fragment demonstrates the usage of the **TPAInfo** element.

```
488
489 <TPAInfo>
490   <TPAId context = "tpadb">12345678</TPAId>
491   <ConversationId context = "tpadb">987654321</ConversationId>
492   <ServiceInterface>QuoteToCollect</ServiceInterface>
493   <Action>NewPurchaseOrder</Action>
494
495 </TPAInfo>
```

496

497    **3.3.3 MessageData**

498    The required **MessageData** element follows the **TPAInfo** element. The purpose of the  
499    **MessageData** element is to provide a means of uniquely identifying the message. It is a  
500    composite element which contains the following three subordinate elements:

- 501    • **MessageId** – a globally unique identifier for the message conforming to [RFC2111].
- 502    • **TimeStamp** – a value representing the time that the message header was created  
503    conforming to [ISO-8601]. The format of CCYYMMDDTHHMMSS.SSSZ is used. This time  
504    format is Coordinated Universal Time (UTC).
- 505    • **RefToMessageId** – a globally unique identifier which relates the current message to a  
506    previous message. It MUST contain the value of the **MessageId** of the related message or  
507    the value "Not Applicable".

508

509    The following example demonstrates the usage of the **MessageData** element.

```
510
511 <MessageData>
512   <MessageId>UUID</MessageId>
513   <TimeStamp>20000725T121905.000Z</TimeStamp>
514   <RefToMessageId>UUID</RefToMessageId>
515
516 </MessageData>
```

517    **3.3.4 ReliableMessagingInfo**

518    The last element of the **ebXMLMessageHeader** is the **ReliableMessagingInfo** element. This  
519    element identifies the degree of reliability with which the message will be delivered. This element  
520    has a single subordinate attribute, **DeliverySemantics**. This attribute is an enumeration, which  
521    MAY have one of the following values:

- 522    • "AtMostOnce" – reliable messaging semantics, which specifies that a given message will be  
523    received by the Service Interface handler no more than once.
- 524    • "Unspecified" – reliable delivery semantics are not specified.



## 525 4 Security Considerations

526 Implementers SHOULD examine carefully the security features of each transport. In the case of  
527 HTTP, Implementers are encouraged to use Realm Security, using basic authentication for  
528 access controls and SSL to protect sensitive information.

529 Users of E-Mail based solution SHOULD ensure that anti-spamming services are in place and  
530 filtering is used to prevent unauthorized access to E-Commerce Servers.

531

## 532 5 References

533 [1] [Overview] ebXML Transport, Routing and Packaging: Overview and Requirements,  
534 Version 0.96, Published 25 May 2000

535 [2] XMT - Extensible Mail Transport Protocol  
536 <http://www.openhealth.org/documents/xmt.htm>

537 [3] ebXML Glossary, see [ebXML Project Team Home Page](#)

538 [4] XML Signature Syntax and Processing

539 [ISO 8601] International Standards Organization Ref. ISO 8601 Second Edition, Published  
540 1997

541 [RFC 2111] IETF Request For Comments 2111 E. Levinson, Published March 1997

542

543

## 544 6 Acknowledgments

545 The author's wish to acknowledge the support of the members of the Transport, Routing and  
546 Packaging Project Team who contributed ideas to this specification by the group's discussion e-  
547 mail list, on conference calls and during face-to-face meetings.

548 Ralph Berwanger – bTrade.com

549 Jonathan Borden - Author of XMT

550 Jon Bosak - Sun

551 Doug Bunting - Ariba

552 David Burdett - Commerce One

553 Lawrence Ding - WorldSpan

554 Rik Drummond - Drummond Group

555 Christopher Ferris - Sun

556 Ian Jones - BT

557 Henry Lowe - OMG

558 Jim McCarthy - webXI

559 Bob Miller - GEIS

560 Dale Moberg - Sterling Commerce



561 Kathy Spector – Extricity  
562 Martha Warfelt - DaimlerChrysler  
563 Prasad Yendluri - Vitria  
564

## 565 7 Authors' Address

566 Contact information for the Authors of this specification follow.

567 David Burdett  
568 Commerce One Inc  
569 4400 Rosewood Drive 3rd Fl, Bldg 4,  
570 Pleasanton, CA 94588,  
571 USA  
572 Telephone: (925) 520-4422 or (650) 623-2888  
573 E-mail: [david.burdett@commerceone.com](mailto:david.burdett@commerceone.com)

574  
575 John Ibbotson  
576 IBM UK Ltd  
577 Hursley Park  
578 Winchester  
579 SO21 2JN  
580 United Kingdom  
581 Telephone: +44 (1962) 815188  
582 E-mail: [john\\_ibbotson@uk.ibm.com](mailto:john_ibbotson@uk.ibm.com)

583  
584 Christopher Ferris  
585 Sun Microsystems, Inc.  
586 One Network Drive  
587 Burlington, Ma 01803-0903  
588 USA  
589 Telephone: (781) 442-3063  
590 E-mail: [chris.ferris@sun.com](mailto:chris.ferris@sun.com)

591  
592 Dick Brooks  
593 Group 8760  
594 110 12th Street North  
595 Suite F103  
596 Birmingham, Alabama 35203  
597 USA  
598 Telephone: (205) 250-8053  
599 E-mail: [dick@8760.com](mailto:dick@8760.com)

600  
601 Nicholas Kassem  
602 Java Software, Sun Microsystems  
603 901 San Antonio Road, MS CUP02-201  
604 Palo Alto, CA 94303-4900



605      Telephone: (408) 863-3535  
606      E-mail: [Nick.Kassem@eng.sun.com](mailto:Nick.Kassem@eng.sun.com)  
607



## 608 Appendix A Schemas and DTD Definitions

609 The following are definitions for validation of the ebXML message header structure.

### 610 A.1 XML Header DTD

```
<?xml version ="1.0"?>
<schema xmlns = "http://www.w3.org/1999/XMLSchema">
<!ELEMENT ebXMLHeader (Manifest , Header )>
<!ATTLIST ebXMLHeader Version   CDATA #FIXED '1.0'
      MessageType CDATA #FIXED 'Normal' >
<!ELEMENT Manifest (DocumentReference )+>
<!ELEMENT DocumentReference (DocumentLabel , DocumentId )>
<!ELEMENT DocumentLabel (#PCDATA )>
<!ATTLIST DocumentLabel e-dtype NMTOKEN #FIXED 'string' >
<!ELEMENT DocumentId (#PCDATA )>
<!ATTLIST DocumentId e-dtype NMTOKEN #FIXED 'uri' >
<!ELEMENT Header (From , To , TPAInfo , MessageData , ReliableMessagingInfo )>
<!--
<!ELEMENT ServiceInterface (#PCDATA )>
<!ATTLIST ServiceInterface e-dtype NMTOKEN #FIXED 'string' >
<!ELEMENT Action (#PCDATA )>
<!ATTLIST Action e-dtype NMTOKEN #FIXED 'string' >
<!ELEMENT TPAId (#PCDATA )>
<!ATTLIST TPAId context CDATA  'Undefined'
      e-dtype NMTOKEN #FIXED 'uri' >
<!ELEMENT ConversationId (#PCDATA )>
<!ATTLIST ConversationId context CDATA  'Undefined'
      e-dtype NMTOKEN #FIXED 'uri' >
<!ELEMENT MessageData (MessageId , TimeStamp , RefToMessageId )>
<!ELEMENT RefToMessageId (#PCDATA )>
<!ATTLIST RefToMessageId e-dtype NMTOKEN #FIXED 'uuid' >
<!ELEMENT TimeStamp (#PCDATA )>
<!ATTLIST TimeStamp e-dtype NMTOKEN #FIXED 'dateTime' >
<!ELEMENT MessageId (#PCDATA )>
<!ATTLIST MessageId e-dtype NMTOKEN #FIXED 'uuid' >
<!ELEMENT From (PartyId )>
<!ELEMENT To (PartyId )>
<!ELEMENT PartyId (#PCDATA )>
<!ATTLIST PartyId context CDATA  'Undefined'
      e-dtype NMTOKEN #FIXED 'uri' >
<!ELEMENT ReliableMessagingInfo EMPTY>
<!ATTLIST ReliableMessagingInfo DeliverySemantics (AtMostOnce | Unspecified ) #FIXED
'Unspecified' >
<!ELEMENT TPAInfo (TPAId , ConversationId , ServiceInterface , Action )>
```



## 650 A.2 XML Header Schema Definition

```
651 <?xml version ="1.0"?>
652 <schema xmlns = "http://www.w3.org/1999/XMLSchema">
653     <element name = "ebXMLHeader">
654         <complexType content = "elementOnly">
655             <sequence>
656                 <element ref = "Manifest"/>
657                 <element ref = "Header"/>
658             </sequence>
659             <attribute name = "Version" use = "fixed" value = "1.0" type = "string"/>
660             <attribute name = "MessageType" use = "fixed" value = "Normal" type = "string"/>
661         </complexType>
662     </element>
663
664     <element name = "Manifest">
665         <complexType content = "elementOnly">
666             <sequence minOccurs = "1" maxOccurs = "unbounded">
667                 <element ref = "DocumentReference"/>
668             </sequence>
669         </complexType>
670     </element>
671
672     <element name = "DocumentReference">
673         <complexType content = "elementOnly">
674             <sequence minOccurs = "1" maxOccurs = "unbounded">
675                 <element ref = "DocumentLabel"/>
676                 <element ref = "DocumentId"/>
677             </sequence>
678         </complexType>
679     </element>
680
681     <element name = "DocumentLabel" type = "string">
682     </element>
683
684     <element name = "DocumentId" type = "uri">
685     </element>
686
687     <element name = "Header">
688         <complexType content = "elementOnly">
689             <sequence>
690                 <element ref = "From"/>
691                 <element ref = "To"/>
692                 <element ref = "TPA"/>
693                 <element ref = "MessageData"/>
694                 <element ref = "ReliableMessagingInfo"/>
695             </sequence>
696         </complexType>
697     </element>
698
699     <element name = "BusinessServiceInterface" type = "string">
700     </element>
```



```
701 <element name = "Action" type = "string"/>
702 <element name = "TPAId">
703     <complexType base = "uri" content = "textOnly">
704         <attribute name = "context" use = "default" value = "Undefined" type = "string"/>
705     </complexType>
706 </element>
707
708 <element name = "ConversationId">
709     <complexType base = "uri" content = "textOnly">
710         <attribute name = "context" use = "default" value = "Undefined" type = "string"/>
711     </complexType>
712 </element>
713
714 <element name = "MessageData">
715     <complexType content = "elementOnly">
716         <sequence>
717             <element ref = "MessageId"/>
718             <element ref = "TimeStamp"/>
719             <element ref = "RefToMessageId"/>
720         </sequence>
721     </complexType>
722 </element>
723
724 <element name = "RefToMessageId" type = "uuid">
725 </element>
726
727 <element name = "TimeStamp" type = "dateTime">
728 </element>
729
730 <element name = "MessageId" type = "uuid">
731 </element>
732
733 <element name = "From">
734     <complexType content = "elementOnly">
735         <sequence>
736             <element ref = "PartyId"/>
737         </sequence>
738     </complexType>
739 </element>
740
741 <element name = "To">
742     <complexType content = "elementOnly">
743         <sequence>
744             <element ref = "PartyId"/>
745         </sequence>
746     </complexType>
747 </element>
748
749 <element name = "PartyId">
750     <complexType base = "uri" content = "textOnly">
751         <attribute name = "context" use = "default" value = "Undefined" type = "string"/>
752     </complexType>
753 </element>
754
```



```
755 <element name = "ReliableMessagingInfo">
756     <complexType content = "empty">
757         <attribute name = "DeliverySemantics" use = "fixed" value = "Unspecified">
758             <simpleType base = "ENUMERATION">
759                 <enumeration value = "AtMostOnce"/>
760                 <enumeration value = "Unspecified"/>
761             </simpleType>
762         </attribute>
763     </complexType>
764 </element>
765
766 <element name = "TPAInfo">
767     <complexType content = "elementOnly">
768         <sequence>
769             <element ref = "TPAId"/>
770             <element ref = "ConversationId"/>
771             <element ref = "BusinessServiceInterface"/>
772             <element ref = "Action"/>
773         </sequence>
774     </complexType>
775 </element>
776
777 </schema>
778
779
```



780

## 781 Appendix B Examples

782 The following are complete examples of ebXML messages showing the structure as defined in  
783 this specification.

### 784 B.1 Complete Example of an ebXML Message enveloped using 785 multipart/related Content-Type sent via HTTP POST

```
786 POST /ebxmlhandler HTTP/1.1
787 Accept: multipart/related
788 Accept-Language: en-us
789 Accept-Encoding: gzip, deflate
790 User-Agent: Group 8760 InsideAgent
791 Host: localhost:9090
792 Connection: Keep-Alive
793 Content-Type: multipart/related; type=application/vnd.eb+xml; version=0.1;
794 boundary=-----7d02a82e5f8
795 Content-Length: 9293
796
797 -----7d02a82e5f8
798 Content-ID: ebxmlheader-9981
799 Content-Length: 211
800 Content-Type: application/vnd.eb+xml; charset="UTF-8";
801
802 <?xml version="1.0" encoding="UTF-8"?>
803 <ebXMLMessageHeader xmlns='http://www.xml.org/ebXMLStds/ebXMLMessageHeaderv1'>
804     <Version>1.0</Version>
805     <MessageType>Request</MessageType>
806     <ServiceType>Payroll</ServiceType>
807     <Intent>RecordCommission</Intent>
808 </ebXMLMessageHeader>
809
810 -----7d02a82e5f8
811 Content-ID: ebxmlpayload-9981
812 Content-Length: 7517
813 Content-Type: text/xml
814
815 <?xml version="1.0" encoding="UTF-8"?>
816 <!-- edited with XML Spy v2.5 - http://www.xmlspy.com -->
817 <HITISMessage xmlns="" Version="1.0">
818     <Header OriginalBodyRequested="false" ImmediateResponseRequired="true">
819         <FromURI>http://www.pms.com/HITISInterfaces</FromURI>
820         <ToURI>http://www.crs.com/HITISInterface</ToURI>
821         <ReplyToURI>http://www.pms.com/HITISInterface</ReplyToURI>
822         <MessageID>1234567890</MessageID>
823         <OriginalMessageID>1234567890</OriginalMessageID>
824         <TimeStamp>1999-11-10T10:23:44</TimeStamp>
825         <Token>1234-567-8901</Token>
826         <!--Token to be assigned in response to HITISRegister-->
827     </Header>
828     <Body>
829         <HITISOperation OperationName="CommissionEventsUpdate">
830             <CommissionEvents>
831                 <CommissionEvent>
832                     <ConfirmationID>18097YZ</ConfirmationID>
833                     <ConfirmationOriginatorCode>DBZ223</ConfirmationOriginatorCode>
834                     <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
835                     <ReservationID>098787818097YZ</ReservationID>
836                     <HotelReference>
837                         <ChainCode>HI234</ChainCode>
838                         <HotelCode>1234STL</HotelCode>
839                     </HotelReference>
840                     <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
841                     <StayDateRange>
```



```
841             <StartInstant>20000122</StartInstant>
842             <Duration>00000003T000000</Duration>
843         </StayDateRange>
844         <GuestNames>
845             <NameInfo>
846                 <NamePrefix>Mr.</NamePrefix>
847                 <NameFirst>John</NameFirst>
848                 <NameMiddle>Q.</NameMiddle>
849                 <NameSur>jones</NameSur>
850                 <NameSuffix>Jr.</NameSuffix>
851                 <NameTitle>Professor</NameTitle>
852                 <NameOrdered>JohnJones</NameOrdered>
853             </NameInfo>
854             <NameInfo>
855                 <NamePrefix>Mrs.</NamePrefix>
856                 <NameFirst>Sally</NameFirst>
857                 <NameMiddle>T.</NameMiddle>
858                 <NameSur>Jones</NameSur>
859                 <NameSuffix/>
860                 <NameTitle/>
861                 <NameOrdered>SallyJones</NameOrdered>
862             </NameInfo>
863         </GuestNames>
864         <ProfileCertification CertificationType="ARC">
865             <CertificationID>67TR901-AZ</CertificationID>
866         </ProfileCertification>
867         <ProfileReference>
868             <!--Profile to be inserted as a reusable component-->
869             <Profile/>
870         </ProfileReference>
871         <Commissions>
872             <Commission CommissionStatusType="Full">
873                 <CommissionableAmount>
874                     <Currency>
875                         <CurrencyCode>USD</CurrencyCode>
876                         <Amount>185.00</Amount>
877                     </Currency>
878                 </CommissionableAmount>
879                 <PrepaidAmount>
880                     <Currency>
881                         <CurrencyCode>USD</CurrencyCode>
882                         <Amount>12.00</Amount>
883                     </Currency>
884                 </PrepaidAmount>
885                 <CommissionPercent>0.0525</CommissionPercent>
886                 <FlatCommission>not applicable<Currency>
887                     <CurrencyCode>USD</CurrencyCode>
888                     <Amount>00.00</Amount>
889                 </Currency>
890             </FlatCommission>
891             <Comment>Default percentage commission agreement</Comment>
892             <CommissionReasonCode>7930</CommissionReasonCode>
893             <BillToID>HOTEL7890</BillToID>
894             <HotelReference>
895                 <ChainCode>HI234</ChainCode>
896                 <HotelCode>1234STL</HotelCode>
897             </HotelReference>
898         </Commission>
899         <Commission CommissionStatusType="Partial">
900             <CommissionableAmount>
901                 <Currency>
902                     <CurrencyCode>USD</CurrencyCode>
903                     <Amount>185.00</Amount>
904                 </Currency>
905             </CommissionableAmount>
906             <PrepaidAmount>
907                 <Currency>
908                     <CurrencyCode>USD</CurrencyCode>
909                     <Amount>00.00</Amount>
910                 </Currency>
911             </PrepaidAmount>
912             <Comment>This commission per agreement with Travel Agents,
913             Inc.</Comment>
914             <CommissionPercent>00.00</CommissionPercent>
915             <FlatCommission>
```



```
916             <Currency>
917                 <CurrencyCode>USD</CurrencyCode>
918                 <Amount>10.00</Amount>
919             </Currency>
920         </FlatCommission>
921         <CommissionReasonCode>7930</CommissionReasonCode>
922         <BillToID>HOTEL7890</BillToID>
923         <HotelReference>
924             <ChainCode>HI234</ChainCode>
925             <HotelCode>1234STL</HotelCode>
926         </HotelReference>
927     </Commission>
928 </Commissions>
929 </CommissionEvent>
930 <CommissionEvent>
931     <ConfirmationID/>
932     <ConfirmationOriginatorCode/>
933     <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
934     <ReservationID>09878783276XY</ReservationID>
935     <HotelReference>
936         <ChainCode>BASS123</ChainCode>
937         <HotelCode>1234STL</HotelCode>
938     </HotelReference>
939     <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
940     <StayDateRange>
941         <StartInstant>20000122</StartInstant>
942         <Duration>00000003T000000</Duration>
943     </StayDateRange>
944     <GuestNames>
945         <NameInfo>
946             <NamePrefix>Mr.</NamePrefix>
947             <NameFirst>Kevin</NameFirst>
948             <NameMiddle>R.</NameMiddle>
949             <NameSur>Smithson</NameSur>
950             <NameSuffix>Jr.</NameSuffix>
951             <NameTitle>Professor</NameTitle>
952             <NameOrdered> Kevin Smithson</NameOrdered>
953         </NameInfo>
954         <NameInfo>
955             <NamePrefix>Miss</NamePrefix>
956             <NameFirst>Mary</NameFirst>
957             <NameMiddle>T.</NameMiddle>
958             <NameSur>Smithson</NameSur>
959             <NameSuffix>esq.</NameSuffix>
960             <NameTitle>Professor</NameTitle>
961             <NameOrdered> MarySmithson</NameOrdered>
962         </NameInfo>
963     </GuestNames>
964     <ProfileCertification CertificationType="ARC">
965         <CertificationID>67TR901-AZ</CertificationID>
966     </ProfileCertification>
967     <ProfileReference>
968         <Profile/>
969     </ProfileReference>
970     <Commissions>
971         <Commission CommissionStatusType="Full">
972             <CommissionableAmount>
973                 <Currency>
974                     <CurrencyCode>USD</CurrencyCode>
975                     <Amount>185.00</Amount>
976                 </Currency>
977             </CommissionableAmount>
978             <PrepaidAmount>
979                 <Currency>
980                     <CurrencyCode>USD</CurrencyCode>
981                     <Amount>12.00</Amount>
982                 </Currency>
983             </PrepaidAmount>
984             <CommissionPercent>0.0525</CommissionPercent>
985             <FlatCommission>not applicable<Currency>
986                 <CurrencyCode>USD</CurrencyCode>
987                 <Amount>00.00</Amount>
988             </Currency>
989         </FlatCommission>
990     <Comment>Default percentage commission agreement</Comment>
```



```
991 <CommissionReasonCode>7930</CommissionReasonCode>
992 <BillToID>HOTEL7890</BillToID>
993 <HotelReference>
994 <ChainCode>HI234</ChainCode>
995 <HotelCode>1234STL</HotelCode>
996 </HotelReference>
997 </Commission>
998 <Commission CommissionStatusType="Partial">
999 <CommissionableAmount>
1000 <Currency>
1001 <CurrencyCode>USD</CurrencyCode>
1002 <Amount>185.00</Amount>
1003 </Currency>
1004 </CommissionableAmount>
1005 <PrepaidAmount>
1006 <Currency>
1007 <CurrencyCode>USD</CurrencyCode>
1008 <Amount>0.00</Amount>
1009 </Currency>
1010 </PrepaidAmount>
1011 <Comment>Flat commission per agreement with TA</Comment>
1012 <CommissionPercent>00.00</CommissionPercent>
1013 <FlatCommission>
1014 <Currency>
1015 <CurrencyCode>USD</CurrencyCode>
1016 <Amount>10.00</Amount>
1017 </Currency>
1018 </FlatCommission>
1019 <CommissionReasonCode>7930</CommissionReasonCode>
1020 <BillToID>HOTEL7890</BillToID>
1021 <HotelReference>
1022 <ChainCode>HI234</ChainCode>
1023 <HotelCode>1234STL</HotelCode>
1024 </HotelReference>
1025 </Commission>
1026 </Commissions>
1027 </CommissionEvent>
1028 </CommissionEvents>
1029 </HITISOperation>
1030 </Body>
1031 </HITISMessage>
1032 -----
1033 -----7d02a82e5f8--
```

## 1034 B.2 Complete Example of an ebXML Message enveloped using 1035 multipart/related Content-Type sent via SMTP

1036 The default Content-transfer-encoding type of 7BIT is being used in this message.

```
1037 From dick@8760.com Sun May 7 17:01:14 2000
1038 Received: from granger.mail.mindspring.net by alpha2000.tech-comm.com;
1039 (8.8.5/1.1.8.2/05Jun95-1217PM)
1040 id RAA32702; Sun, 7 May 2000 17:01:13 -0500 (CDT)
1041 Received: from gamma (user-33qt101.dialup.mindspring.com [199.174.132.21])
1042 by granger.mail.mindspring.net (8.9.3/8.8.5) with SMTP id SAA11942
1043 for <ebxmlhandler@8760.com>; Sun, 7 May 2000 18:11:14 -0400 (EDT)
1044 From: "Dick Brooks (E)" <dick@8760.com>
1045 To: <ebxmlhandler@8760.com>
1046 Subject: OTA Commission Event
1047 Date: Sun, 7 May 2000 17:07:38 -0500
1048 Message-ID: <NDDBIOBLMLCDOHCHIKMGKEEIDAAA.dick@8760.com>
1049 MIME-Version: 1.0
1050 X-Priority: 3 (Normal)
1051 X-MSMail-Priority: Normal
1052 X-Mailer: Microsoft Outlook IMO, Build 9.0.2416 (9.0.2910.0)
1053 Importance: Normal
1054 X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2314.1300
1055 Content-Length: 8081
1056 Content-Type: multipart/related; type="application/vnd.eb+xml"; version="0.1";
1057 charset="iso-8859-1"; boundary="-----_NextPart_0005_01BFB846.BF7FABA0"
```



```
1059
1060 -----_NextPart_000_0005_01BFB846.BF7FABA0
1061 Content-Type: application/vnd.eb+xml
1062 Content-ID: ebxmlheader-9000
1063 Content-Length: 272
1064
1065 <?xml version="1.0" encoding="UTF-8"?>
1066 <ebXMLMessageHeader xmlns='http://www.xml.org/ebXMLStds/ebXMLMessageHeaderv1'>
1067 <Version>1.0</Version>
1068 <MessageType>Request</MessageType>
1069 <ServiceType>Payroll</ServiceType>
1070 <Intent>RecordCommission</Intent>
1071 </ebXMLMessageHeader>
1072 -----_NextPart_000_0005_01BFB846.BF7FABA0
1073 Content-Type: text/xml
1074 Content-ID: ebxmlpayload-9000
1075 Content-Length: 7515
1076
1077 <?xml version="1.0" encoding="UTF-8"?>
1078 <!-- edited with XML Spy v2.5 - http://www.xmlspy.com -->
1079 <HITISMessage xmlns="" Version="1.0">
1080     <Header OriginalBodyRequested="false" ImmediateResponseRequired="true">
1081         <FromURI>http://www.pms.com/HITISInterface</FromURI>
1082         <ToURI>http://www.crs.com/HITISInterface</ToURI>
1083         <ReplyToURI>http://www.pms.com/HITISInterface</ReplyToURI>
1084         <MessageID>1234567890</MessageID>
1085         <OriginalMessageID>1234567890</OriginalMessageID>
1086         <TimeStamp>1999-11-10T10:23:44</TimeStamp>
1087         <Token>1234-567-8901</Token>
1088         <!--Token to be assigned in response to HITISRegister-->
1089     </Header>
1090     <Body>
1091         <HITISOperation OperationName="CommissionEventsUpdate">
1092             <CommissionEvents>
1093                 <CommissionEvent>
1094                     <ConfirmationID>18097YZ</ConfirmationID>
1095                     <ConfirmationOriginatorCode>DBZ223</ConfirmationOriginatorCode>
1096                     <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
1097                     <ReservationID>098787818097YZ</ReservationID>
1098                     <HotelReference>
1099                         <ChainCode>HI234</ChainCode>
1100                         <HotelCode>1234STL</HotelCode>
1101                     </HotelReference>
1102                     <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
1103                     <StayDateRange>
1104                         <StartInstant>20000122</StartInstant>
1105                         <Duration>00000003T000000</Duration>
1106                     </StayDateRange>
1107                     <GuestNames>
1108                         <NameInfo>
1109                             <NamePrefix>Mr.</NamePrefix>
1110                             <NameFirst>John</NameFirst>
1111                             <NameMiddle>Q.</NameMiddle>
1112                             <NameSur>jones</NameSur>
1113                             <NameSuffix>Jr.</NameSuffix>
1114                             <NameTitle>Professor</NameTitle>
1115                             <NameOrdered>JohnJones</NameOrdered>
1116                         </NameInfo>
1117                         <NameInfo>
1118                             <NamePrefix>Mrs.</NamePrefix>
1119                             <NameFirst>Sally</NameFirst>
1120                             <NameMiddle>T.</NameMiddle>
1121                             <NameSur>Jones</NameSur>
1122                             <NameSuffix/>
1123                             <NameTitle/>
1124                             <NameOrdered>SallyJones</NameOrdered>
1125                         </NameInfo>
1126                     </GuestNames>
1127                     <ProfileCertification CertificationType="ARC">
1128                         <CertificationID>67TR901-AZ</CertificationID>
1129                     </ProfileCertification>
1130                     <ProfileReference>
1131                         <!--Profile to be inserted as a reusable component-->
1132                         <Profile/>
1133                     </ProfileReference>
```



```
1134 <Commissions>
1135     <Commission CommissionStatusType="Full">
1136         <CommissionableAmount>
1137             <Currency>
1138                 <CurrencyCode>USD</CurrencyCode>
1139                 <Amount>185.00</Amount>
1140             </Currency>
1141         </CommissionableAmount>
1142         <PrepaidAmount>
1143             <Currency>
1144                 <CurrencyCode>USD</CurrencyCode>
1145                 <Amount>12.00</Amount>
1146             </Currency>
1147         </PrepaidAmount>
1148         <CommissionPercent>0.0525</CommissionPercent>
1149         <FlatCommission>not applicable<Currency>
1150             <CurrencyCode>USD</CurrencyCode>
1151             <Amount>00.00</Amount>
1152         </Currency>
1153     </FlatCommission>
1154     <Comment>Default percentage commission agreement</Comment>
1155     <CommissionReasonCode>7930</CommissionReasonCode>
1156     <BillToID>HOTEL7890</BillToID>
1157     <HotelReference>
1158         <ChainCode>HI234</ChainCode>
1159         <HotelCode>1234STL</HotelCode>
1160     </HotelReference>
1161 </Commission>
1162 <Commission CommissionStatusType="Partial">
1163     <CommissionableAmount>
1164         <Currency>
1165             <CurrencyCode>USD</CurrencyCode>
1166             <Amount>185.00</Amount>
1167         </Currency>
1168     </CommissionableAmount>
1169     <PrepaidAmount>
1170         <Currency>
1171             <CurrencyCode>USD</CurrencyCode>
1172             <Amount>00.00</Amount>
1173         </Currency>
1174     </PrepaidAmount>
1175     <Comment>This commission per agreement with Travel Agents,
1176 Inc.</Comment>
1177     <CommissionPercent>00.00</CommissionPercent>
1178     <FlatCommission>
1179         <Currency>
1180             <CurrencyCode>USD</CurrencyCode>
1181             <Amount>10.00</Amount>
1182         </Currency>
1183     </FlatCommission>
1184     <CommissionReasonCode>7930</CommissionReasonCode>
1185     <BillToID>HOTEL7890</BillToID>
1186     <HotelReference>
1187         <ChainCode>HI234</ChainCode>
1188         <HotelCode>1234STL</HotelCode>
1189     </HotelReference>
1190 </Commissions>
1191 </Commissions>
1192 </CommissionEvent>
1193 <CommissionEvent>
1194     <ConfirmationID/>
1195     <ConfirmationOriginatorCode/>
1196     <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
1197     <ReservationID>09878783276XY</ReservationID>
1198     <HotelReference>
1199         <ChainCode>BASS123</ChainCode>
1200         <HotelCode>1234STL</HotelCode>
1201     </HotelReference>
1202     <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
1203     <StayDateRange>
1204         <StartInstant>20000122</StartInstant>
1205         <Duration>0000003T000000</Duration>
1206     </StayDateRange>
1207     <GuestNames>
1208         <NameInfo>
```



```
1209 <NamePrefix>Mr.</NamePrefix>
1210 <NameFirst>Kevin</NameFirst>
1211 <NameMiddle>R.</NameMiddle>
1212 <NameSur>Smithson</NameSur>
1213 <NameSuffix>Jr.</NameSuffix>
1214 <NameTitle>Professor</NameTitle>
1215 <NameOrdered> Kevin Smithson</NameOrdered>
1216 </NameInfo>
1217 <NameInfo>
1218 <NamePrefix>Miss</NamePrefix>
1219 <NameFirst>Mary</NameFirst>
1220 <NameMiddle>T.</NameMiddle>
1221 <NameSur>Smithson</NameSur>
1222 <NameSuffix>esq.</NameSuffix>
1223 <NameTitle>Professor</NameTitle>
1224 <NameOrdered> MarySmithson</NameOrdered>
1225 </NameInfo>
1226 </GuestNames>
1227 <ProfileCertification CertificationType="ARC">
1228 <CertificationID>67TR901-AZ</CertificationID>
1229 </ProfileCertification>
1230 <ProfileReference>
1231 <Profile/>
1232 </ProfileReference>
1233 <Commissions>
1234 <Commission CommissionStatusType="Full">
1235 <CommissionableAmount>
1236 <Currency>
1237 <CurrencyCode>USD</CurrencyCode>
1238 <Amount>185.00</Amount>
1239 </Currency>
1240 </CommissionableAmount>
1241 <PrepaidAmount>
1242 <Currency>
1243 <CurrencyCode>USD</CurrencyCode>
1244 <Amount>12.00</Amount>
1245 </Currency>
1246 </PrepaidAmount>
1247 <CommissionPercent>0.0525</CommissionPercent>
1248 <FlatCommission>not applicable<Currency>
1249 <CurrencyCode>USD</CurrencyCode>
1250 <Amount>00.00</Amount>
1251 </Currency>
1252 </FlatCommission>
1253 <Comment>Default percentage commission agreement</Comment>
1254 <CommissionReasonCode>7930</CommissionReasonCode>
1255 <BillToID>HOTEL7890</BillToID>
1256 <HotelReference>
1257 <ChainCode>HI234</ChainCode>
1258 <HotelCode>1234STL</HotelCode>
1259 </HotelReference>
1260 </Commission>
1261 <Commission CommissionStatusType="Partial">
1262 <CommissionableAmount>
1263 <Currency>
1264 <CurrencyCode>USD</CurrencyCode>
1265 <Amount>185.00</Amount>
1266 </Currency>
1267 </CommissionableAmount>
1268 <PrepaidAmount>
1269 <Currency>
1270 <CurrencyCode>USD</CurrencyCode>
1271 <Amount>00.00</Amount>
1272 </Currency>
1273 </PrepaidAmount>
1274 <Comment>Flat commission per agreement with TA</Comment>
1275 <CommissionPercent>00.00</CommissionPercent>
1276 <FlatCommission>
1277 <Currency>
1278 <CurrencyCode>USD</CurrencyCode>
1279 <Amount>10.00</Amount>
1280 </Currency>
1281 </FlatCommission>
1282 <CommissionReasonCode>7930</CommissionReasonCode>
1283 <BillToID>HOTEL7890</BillToID>
```



```
1284             <HotelReference>
1285                 <ChainCode>HI234</ChainCode>
1286                 <HotelCode>1234STL</HotelCode>
1287             </HotelReference>
1288         </Commission>
1289     </Commissions>
1290     </CommissionEvent>
1291   </CommissionEvents>
1292 </HITISOperation>
1293 </Body>
1294 </HITISMMessage>
1295 -----=_NextPart_000_0005_01BFB846.BF7FABA0--
```

1296



## 1297   **Appendix C Candidate Packaging Technologies and** 1298   **Selection Process**

1299

1300   The packaging sub-group began its investigation of packaging technologies by identifying the  
1301   technologies currently used for business-to-business message exchange or were being  
1302   developed for this purpose. The following packaging technologies were identified:

- 1303   •   MIME - currently in use by companies exchanging business transactions using E-mail and  
1304   HTTP
- 1305   •   XML - currently used by RosettaNet and Microsoft (BizTalk and SOAP) and others

### 1306   **C.1 Selection Process**

1307   Each candidate technology was evaluated based on its ability to meet the requirements listed in  
1308   the section titled "Packaging and other Requirements" in this document. When necessary,  
1309   specific parties were contacted to provide details describing how a technology was being used to  
1310   meet specific requirements. The following parties were contacted to provide expert insight:

- 1311   •   Microsoft - David Turner, regarding use of XML packaging in BizTalk
- 1312   •   Develop Mentor - Don Box, regarding use of XML packaging in SOAP
- 1313   •   Vitria - Prasad Yendluri, regarding use of XML packaging in RosettaNet
- 1314   •   Jonathan Borden - author of XMTP [3], an XML to MIME transformation tool
- 1315   The packaging sub-group considered the inputs of people from the ebXML Transport mailing list  
1316   as well as the parties listed above, before making a selection.

### 1317   **C.2 MIME**

1318   Multipurpose Internet Mail Extensions (MIME) is an international standard created by the Internet  
1319   Engineering Task Force. It has been implemented by numerous software vendors across the  
1320   globe and has been used to exchange mixed type payloads, including XML, for several years.  
1321   MIME was designed purely as a packaging (enveloping) solution to allow the transport of mixed  
1322   payloads using Internet E-mail (SMTP). MIME is also being used by other transport technologies  
1323   as a packaging technology, most notably HTTP.

### 1324   **C.3 XML**

1325   eXtensible Markup Language (XML) version 1.0 is a technical specification holding a  
1326   RECOMMENDED status created by the World Wide Web Consortium. It has been implemented  
1327   by numerous software vendors across the globe and has been used to describe a broad  
1328   spectrum of document structures from very simple to very complex. XML is a very flexible  
1329   markup language that can be used to represent virtually any type of document. XML can be used  
1330   solely for packaging (enveloping) documents of any type, providing the data can be  
1331   "transformed" into "legal" XML.



1332 In some cases, XML documents MUST be placed into transport specific "envelopes" before  
1333 being transported. For example, XML data MUST be placed in a MIME envelope when being  
1334 transported via SMTP or HTTP.

#### 1335 C.4 Conclusion

1336 The packaging sub-group examined the capabilities of both XML and MIME relative to the list of  
1337 packaging requirements above. It's important to note that neither technology met all of the  
1338 ebXML requirements and in the end it was the packaging sub-groups assessment of which  
1339 technology came closest to meeting ALL of the ebXML requirements that determined which  
1340 technology SHOULD be used.

1341 MIME was chosen to serve as the ebXML packaging technology, over XML, based on the  
1342 information contained in following table:

1343

Reason	Requirement(s) Satisfied
There is no formal packaging recommendation within IETF or W3C, based on XML. If ebXML were to choose XML as a packaging technology it would be required to define an XML packaging specification and submit this to IETF or W3C for adoption as a formal standard.	to not reinvent the wheel - re-use where possible [2]
XML requires that binary and other types of payload data including XML documents be base64 encoded in order to be encapsulated within a XML root document. Base64 encoding ensures that no illegal XML characters exist within a document and recursive XML documents are "hidden". Base64 encoding imposes a significant processing overhead and results in larger messages, which affect both transmission and processing times. Base64 encoding of binary data is required of MIME content when being transported by SMTP, but this is a transport level requirement, not a requirement imposed by MIME. Binary data can be packaged and transported without alteration when using MIME over HTTP	Minimize intrusion to payload (special encoding or alteration) Low processing overhead
At the time of defining this specification there is no industry standard way to package an encrypted message, or portion of a message, using XML.	All or part of the documents in a message MAY be encrypted prior to sending [2]
MIME could be used in conformance within existing IETF recommendations, no additions or changes are initially required to produce a functional envelope.	to not reinvent the wheel - re-use where possible [2]

1344

1345



## 1346 Appendix D MIME Type discussion

1347

1348 Three MIME media types were considered to serve as Content-Type for the ebXML Message  
1349 Envelope:

1350 • Multipart/related

1351 • Multipart/Mixed

1352 • Multipart/form-data

1353 **The group selected the multipart/related media type to serve as the preferred message  
1354 envelope Content-Type.**

1355 Note:

1356 *There was some discussion over the similarities of multipart/related and multipart/mixed, both of  
1357 which appear to offer similar capabilities and both could meet stated requirements. However, the  
1358 group converged on multipart/related, believing it to be more semantically appropriate for ebXML.*

1359 *There was significant discussion over whether to support multipart/form-data as an alternate  
1360 Content-Type for message-envelope, due to the large installed base of web browsers that  
1361 support this Content-Type.*

1362 *It was determined that multipart/related was a more generic Content-Type than multipart/form-  
1363 data and the multipart/related Content-Type is the preferred Content-Type for ebXML message  
1364 envelopes. Multipart/form-data Content-Type is typically associated with HTTP/HTML web forms,  
1365 whereas multipart/related can be associated with any type of data.*

1366 Additionally, due to limitations in their handling of multipart ebXML payloads it was determined  
1367 that existing web browsers are unable to support the full breadth of functions needed to package  
1368 complex ebXML messages containing multipart payloads. Therefore browser vendors are  
1369 encouraged to add support for the ebXML enveloping standard as specified in this document.