



Creating A Single Global Electronic Market

# ebXML Transport, Routing & Packaging Message Envelope Specification

## Working Draft 26-May-2000

**This version:**

ebXML Message Envelope Specification v0-5.doc

**Latest version:**

N/A

**Previous version:**

ebXML Transport, Routing & Packaging Message Envelope Specification (STRAWMAN 0.4)

**Editor:**

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

**Authors:**

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

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

**Contributors:**

See Acknowledgments

### 1 Abstract

2 This document is a draft proposal whose purpose is to solicit additional input and convey the  
3 current state of the ebXML packaging recommendations.

4 This document defines the structure (or envelope) used to encapsulate data for transport  
5 between parties, following the specifications defined by ebXML. Every attempt has been made to  
6 ensure that ebXML requirements, related to transport, routing and packaging are addressed  
7 within this specification. Adherence to industry standards, consideration of existing business-to-  
8 business practices and support for Small and Medium Enterprises were key factors influencing  
9 the direction of this specification.

### 10 Status of this Document

11 This document is a draft for Public Comment. The document represents work in progress and no  
12 reliance should be made.

13 The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD",  
14 "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be  
15 interpreted as described in IETF RFC 2119.

### Table of Contents

1 Introduction.....3



1.1	Purpose and Scope .....	3
1.1.1	Goals .....	3
1.2	Relationship to other specifications.....	3
1.3	Specification Structure.....	4
2	Packaging and Other Requirements.....	4
3	Candidate Packaging Technologies and Selection Process.....	6
3.1	Selection Process .....	6
3.2	MIME.....	6
3.3	XML.....	6
3.4	Conclusion.....	7
4	Packaging Specification.....	8
4.1	General Conventions.....	8
4.2	Message Structure.....	8
4.3	Transport Envelope .....	9
	OPEN ISSUE 1.0 .....	9
4.4	Message Envelope Specifications.....	10
4.4.1	Content-type .....	10
	OPEN ISSUE 2.0 .....	10
4.4.2	Content-Length .....	11
4.4.3	Complete ebXML Message Envelope Example .....	11
4.5	ebXML Header Container Specifications.....	11
4.5.1	Content-ID .....	12
	OPEN ISSUE 3.0 .....	12
4.5.2	Content-Length .....	12
4.5.3	Content-Type .....	12
	4.5.3.1 Optional Support for Signed Headers .....	12
4.5.4	Complete Example of an ebXML Header container .....	13
4.6	Payload Container Specifications .....	13
4.6.1	Content-ID .....	13
	OPEN ISSUE 4.0 .....	13
4.6.2	Content-Length .....	14
4.6.3	Content-Type .....	14
	4.6.3.1 Optional Support for Signed and Encrypted Payloads.....	14
4.6.4	Complete Example of an ebXML Payload container .....	14
4.7	Complete Example of an ebXML Message enveloped using multipart/related content-type sent via HTTP POST .....	14
4.8	Complete Example of an ebXML Message enveloped using multipart/related content-type sent via SMTP .....	18
5	Security Considerations .....	22
6	References .....	22
7	Acknowledgments.....	22
9	Authors' Address .....	22



# 1 Introduction

This specification defines the message structure used to encapsulate ebXML message headers and payloads for transport between parties. No assumption or dependency is made relative to transport protocol or type of payload; the specifications contained here are both payload and transport agnostic. This main goal of this specification is to define an enveloping structure to encapsulate any digitally encoded payload for transport over any data communication mechanism. No limitation is implied relative to processing mode, the structures defined in this specification can be used in one-way, broadcast, request/response (RPC) or full messaging mode communications between parties.

## 1.1 Purpose and Scope

This document provides software practitioners with sufficient detail to develop software used in the packaging, exchange and processing of information following ebXML Transport, Routing and Packaging specifications. This document defines the enveloping specifications used to represent ebXML messages and encapsulate ebXML message headers and digital payloads for transport over a data communication mechanism. There are other aspects of ebXML messaging that are not addressed in this document, for example: Content and Semantics of Message Headers, payload structure, business processes, choreography of message exchanges and error handling. These are addressed in other ebXML specification documents.

Software practitioners are expected to use this document in combination with other ebXML specification documents when creating ebXML compliant software.

### 1.1.1 Goals

The goals of this specification are:

- Meet the requirements specified by the ebXML Transport, Routing and Packaging Overview and Requirements Document -version 0.92 [2]
- Meet the requirements identified by the packaging sub-group (the people responsible for creating this specification)
- Compatible with other ebXML specifications
- Leverage existing industry standards
- Implementable in a prototype by the May meeting of ebXML group
- Enable parties to "package" very simple to very complex combinations of headers and payloads
- Payload Neutral
- Transport Neutral
- Support corporate security policies and business practices

## 1.2 Relationship to other specifications

This specification is one of a set of related specifications, for details see ebXML Transport, Routing & Packaging Document Control and Index [1]



### 38 **1.3 Specification Structure**

39 This specification is organized around four main topics:

- 40 • Packaging and Other Requirements
- 41 • Candidate Packaging Technologies and Selection Process
- 42 • Packaging Specifications
- 43 • Security Considerations

## 44 **2 Packaging and Other Requirements**

45 The packaging sub-group began development of the ebXML envelope specification (this  
46 document) by first identifying requirements from the Transport Routing and Packaging Overview  
47 and Requirements [2] that directly affect enveloping. Secondly, the group identified  
48 requirements, specific to enveloping that were not included in [2]. This combined list of  
49 requirements was used by the group to evaluate candidate packaging technologies and would  
50 ultimately serve as the "checklist" for choosing a solution. The combined list of requirements  
51 considered by the packaging sub-group includes:

- 52 • Able to handle large documents
- 53 • Able to envelope any document type
- 54 • Minimize intrusion to payload (special encoding or alterations)
- 55 • Minimize potential for abnormal termination caused by envelopes
- 56 • Facilitate a migration path for existing installed base and technologies
- 57 • Low processing overhead
- 58 • Support for recursive documents
- 59 • Able to preserve digital signatures
- 60 • Able to unambiguously identify signed data
- 61 • Documents, expressed either in XML or other electronic formats, must be able to be wrapped  
62 inside a *message envelope* for transporting between the *parties* involved in that want to  
63 execute an eCommerce - *Service* or *Transaction* [2]
- 64 • Multiple *documents*, whether related or not, may be transportable within a single *message*  
65 *envelope* [2]
- 66 • *Messages* can be transported over many network protocols (e.g. HTTP, SMTP, CORBA,  
67 JMQ, MQSeries, MSMQ, etc.) [2]
- 68 • *Messages* can be sent using a variety of methods: [2]
  - 69 – to a single *party*, e.g. by specifying a URL
  - 70 – to multiple *parties*, e.g. by specifying a list of URLs
  - 71 – to an agent or intermediary for forwarding to the next *party*
- 72 • Individual *messages* must be capable of routing serially or in parallel with other related  
73 *messages* [2]
- 74 • Publish and Subscribe[2]



- 75 – Messages may be distributed to the members of a list of *parties* using a "Publish and
- 76 Subscribe"
- 77 – mechanism
- 78 – the anonymity of the subscriber may optionally be maintained
- 79 • Documents and/or message headers may be digitally signed [2]
- 80 • The signature over the *documents* or *message headers* should be independent of the
- 81 transport protocol used [2]
- 82 • A single digital signature may be used to bind together *documents* either: [2]
- 83 – within the same *message*
- 84 – in another *message*
- 85 – somewhere else (for example the content at a URL)
- 86 • Signatures on digitally signed *documents* can be used to: [2]
- 87 – verify the authenticity of the *party* that is the sender,
- 88 – provide non-repudiation of origin or receipt, and
- 89 – ensure that the content of the message has not changed
- 90 • All or part of the *documents* in a *message* may be encrypted prior to sending [2]
- 91 • *messages* may be encrypted during transportation using a transport protocol [2]
- 92 • documents may be time stamped securely with a digital signature [2]
- 93 • Platform Independent Interoperability [2]
- 94 – Servers/systems that support the exchange of documents can be treated as "black
- 95 boxes"
- 96 – The method used to transport documents is completely independent of:
- 97 – the hardware used by the server/services at each end
- 98 – the software or systems architecture of the server/services at each
- 99 – the language used for implementation of systems and *applications*.
- 100 – Support for a *service* can be expressed solely in terms of the type and sequence in
- 101 which *documents* (and their *message envelopes*) can be exchanged
- 102 – The approach must be suitable for implementation on hardware that varies from a very
- 103 simple device to a large multi-processor/system complex
- 104 • The protocol must be extensible to support: [2]
- 105 – additional types of data in message headers and message routing information
- 106 – new values for codes
- 107 – new ways and methods of exchanging data
- 108 • enable any party to carry out integrated eCommerce transactions with any other party
- 109 anywhere in the world using their hardware and software vendor of choice [2]
- 110 • attract a wide variety of vendors to implement the approach [2]
- 111 • to not reinvent the wheel - re-use where possible [2]
- 112 • to enable existing "messaging" solutions to "bridge" to the ebXML solution [2]
- 113 • to scale from SMEs to large companies [2]



- 114
- to scale from low power to high end solutions [2]

## 115 **3 Candidate Packaging Technologies and Selection**

### 116 **Process**

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

- 120
- MIME - currently in use by companies exchanging business transactions using E-mail and  
121 HTTP
  - XML - currently used by RosettaNet and Microsoft (BizTalk and SOAP) and others
- 122

### 123 **3.1 Selection Process**

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

- 128
- Microsoft - David Turner, regarding use of XML packaging in BizTalk
  - 129 • Develop Mentor - Don Box, regarding use of XML packaging in SOAP
  - 130 • Vitria - Prasad Yendluri, regarding use of XML packaging in RosettaNet
  - 131 • Jonathan Borden - author of XMTP [3], an XML to MIME transformation tool

132 The packaging sub-group considered the inputs of people from the ebXML Transport mailing list  
133 as well as the parties listed above, before making a selection.

### 134 **3.2 MIME**

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

### 141 **3.3 XML**

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

149 In some cases, XML documents must be placed into transport specific "envelopes" before being  
150 transported. For example, XML data must be placed in a MIME envelope when being transported  
151 via SMTP or HTTP.



152 **3.4 Conclusion**

153 The packaging sub-group examined the capabilities of both XML and MIME relative to the list of  
 154 packaging requirements above. It's important to note that neither technology met all of the  
 155 ebXML requirements and in the end it was the packaging sub-groups assessment of which  
 156 technology came closest to meeting ALL of the ebXML requirements that determined which  
 157 technology should be used.

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

160

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]

161 **The packaging sub-group did find that the deficiencies listed above that caused XML to**  
 162 **be excluded were directly related to XML's immaturity relative to MIME. It was the sub-**  
 163 **groups opinion that XML is a powerful technology, with great potential and the ebXML**  
 164 **group should continue to monitor XML's progress in these areas. It is expected that XML**  
 165 **will overcome these issues and may one day provide a future packaging solution suitable**  
 166 **for ebXML.**

167 The ebXML executive committee should consider sending this document to the W3C for  
 168 consideration as a set of requirements to be used by a W3C workgroup in the creation of an XML  
 169 based packaging solution.



## 170 4 Packaging Specification

### 171 4.1 General Conventions

- 172 • All headers, attributes and values defined in this specification are to be handled in a case  
173 insensitive fashion, regardless of the way information is presented in this document.
- 174 • All messages following the ebXML standard must follow the specifications for packaging  
175 defined in this document, regardless of message type (request, response, error, et al).
- 176 • Values associated with MIME header attributes are valid in both "quoted" and unquoted  
177 form, for example both of the following forms are valid: (type="ebxml" or type=ebxml)

178

### 179 4.2 Message Structure

180 A Message Consists of:

- 181 • a conditional outer **Transport Envelope**, such as HTTP or SMTP,
- 182 • a transport independent **Message Envelope**, for example MIME multipart/related, that  
183 contains the two main parts of the Message:
  - 184 – a **Header container** that is used to envelope one ebXML header document, and
  - 185 – an optional **Payload container** that is used to envelope the real payload of the Message

186

187

188

189

190

191

192

193

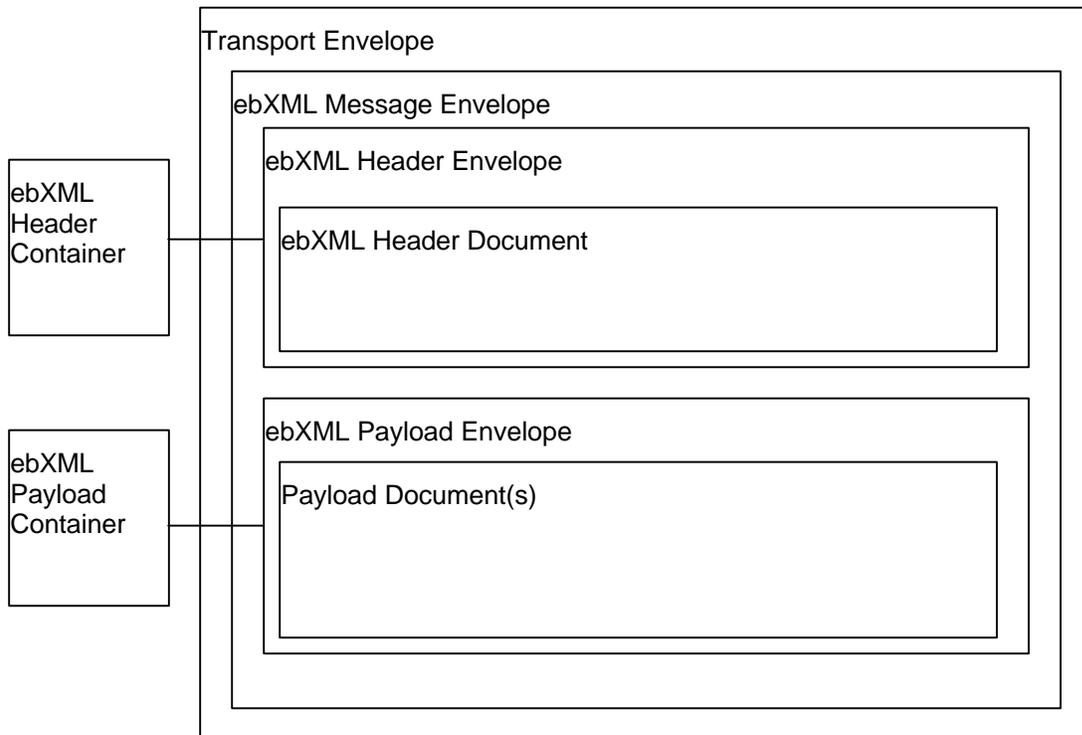
194

195

196

197

198





### 199 4.3 Transport Envelope

200 This document does NOT define any requirements affecting the structure of transport level  
201 envelopes. It is expected that existing transport systems, such as SMTP, HTTP, FTP and others  
202 can be used to send/receive ebXML compliant messages, without modification. The only  
203 requirement ebXML has on the transport envelope is the ability to identify a specific "handler" to  
204 receive incoming ebXML messages. All transports known at the time of creating this specification  
205 support this requirement.

206 A transport envelope is only required in those cases requiring such structures. In the case of  
207 HTTP or SMTP transport envelopes are REQUIRED, however in the case of FTP no transport  
208 envelope is needed.

209 In summary, an implementer of software to process ebXML messages must be aware of  
210 transport specific requirements relative to transport envelopes.

211 Implementers are expected to provide an *ebXML handler* to process all incoming ebXML  
212 requests for service contained within an ebXML message. This handler may be a dispatch  
213 process or an actual application with the capability to process the message. There will be at least  
214 one *ebXML handler* for each supported transport protocol. In the SMTP case this ebXML handler  
215 could be associated with a particular mailbox (e.g. ebxmlhandler@mycompany.com). In the  
216 HTTP case the ebXML handler is contained in the Request-URI on a POST operation, for  
217 example:

218 Request URI

219  
220 POST /ebxmlhandler HTTP/1.1

221 Implementers must provide a means to communicate the name of an ebXML handler for all  
222 trading partners to use when sending ebXML messages. Implementers should consider using a  
223 common identifier, such as "ebxmlhandler", or, alternatively provide a "discovery mechanism" to  
224 be used by a sender to determine the ebXML handler to receive service request.

#### 225 OPEN ISSUE 1.0

226 **When ebXML messages are transported using MIME aware transports, such as SMTP and**  
227 **HTTP, the ebXML message envelope MIME headers are sent as part of the transport layer**  
228 **MIME headers. Concerns over a receivers ability to separate ebXML message envelope**  
229 **MIME headers from the transport layer MIME headers have been raised as this could**  
230 **affect a receivers ability to create a digitally signed receipt of the "received" ebXML**  
231 **message, including the ebXML message envelope MIME headers. The issue to be**  
232 **resolved is:**

233 **"What portion of an ebXML message should be considered within the scope of a**  
234 **signature block when creating signed receipts?"**

235 **If the ebXML message envelope MIME headers are within the "signature block" used to**  
236 **create a signed receipt then software implementers may not be able to reconstruct the**  
237 **original ebXML message, complete with ebXML message MIME headers, using the CGI or**  
238 **Servlet interface used within many commercial web server products. If ebXML message**  
239 **envelope MIME headers are NOT within the signature block used to create signed**  
240 **receipts then there is no issue. Alternatively, ebXML implementers could develop their**  
241 **own HTTP processor, which could provide access to the original stream of data sent by**  
242 **the sender.**



## 243 4.4 Message Envelope Specifications

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

- 247 1. Content-Length
- 248 2. Content-type

### 249 4.4.1 Content-type

250 Three MIME media types were considered to serve as content-type for the ebXML Message  
251 Envelope:

- 252 • Multipart/related
- 253 • Multipart/Mixed
- 254 • Multipart/form-data

255 **The group selected the multipart/related media type to serve as the preferred message**  
256 **envelope content-type.**

257 *Note:*

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

261 *There was significant discussion over whether to support multipart/form-data as an alternate*  
262 *content-type for message-envelope, due to the large installed base of web browsers that support*  
263 *this content-type.*

264 *It was determined that multipart/related was a more generic content-type than multipart/form-data*  
265 *and the multipart/related content-type is the preferred content-type for ebXML message*  
266 *envelopes. Multipart/form-data content-type is typically associated with HTTP/HTML web forms,*  
267 *whereas multipart/related can be associated with any type of data.*

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

272 The Content-type header also contains three attributes:

- 273 1. type
- 274 2. version
- 275 3. boundary

276 The type attribute is used to identify the message envelope as an ebXML compliant structure.  
277 There is only one valid value for this attribute: "ebxml". The following is an example usage of the  
278 type attribute:

```
279 Content-type: multipart/related; type="ebxml"
```

### 280 OPEN ISSUE 2.0

281 **ebXML's use of the type parameter, as described above, is in conflict with the usage of**  
282 **this parameter as defined in RFC 2387, which states "The type parameter must be**  
283 **specified and its value is the MIME media type of the "root" body part."**



284 **Several alternatives have been discussed:**

285 **Option 1 - Add a new parameter to the multipart/related media type (possibly "subtype" or**  
286 **another name) which would be used to identify the message as an "ebxml" compliant**  
287 **message.**

288 **Option 2 - Register a new MIME media type "application/vnd.ebxml" which would be used**  
289 **to identify the Content-type of the ebXML header body part (root body part). The type**  
290 **parameter within the multipart/related message header would contain**  
291 **"application/vnd.ebxml", which conforms with RFC 2387.**

292 The version attribute is used to identify the particular version of ebxml message envelope being  
293 used. There are currently two valid values for version:

- 294 1. "0" indicating a version-less message; ALL ebXML implementations must support  
295 version-less messages
- 296 2. "0.1" indicating the current version of ebXML.

297 Currently, there are no version-less message envelopes defined, therefore all message headers  
298 SHOULD USE "0.1". The following is an example usage of version:

```
299 Content-type: multipart/related; type="ebxml"; version="0.1"
```

300 The boundary attribute is used to identify the body part separator used to identify the start and  
301 end points of each body part contained in the message. The boundary should be chosen  
302 carefully to insure that it does not occur within the content area of a body part. Example usage of  
303 the boundary attribute:

```
304 Content-type: multipart/related; type="ebxml"; version="0.1";  
305 boundary="-----8760"
```

#### 306 **4.4.2 Content-Length**

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

```
309 Content-Length: 9841
```

#### 310 **4.4.3 Complete ebXML Message Envelope Example**

311 An example of a complete ebXML compliant Message Envelope appears as follows:

```
312 Content-type: multipart/related; type="ebxml"; version="0.1";  
313 boundary="-----8760"  
314 Content-Length: 9841
```

### 315 **4.5 ebXML Header Container Specifications**

316 The ebXML Header container is a MIME body part used to encapsulate an ebXML header  
317 document. The ebXML header document is described in ebXML Message Header Specification  
318 [3]. There MUST BE one ebXML header document associated with every ebXML Message. The  
319 ebXML Header container **consists of a MIME Header portion, referred to as the ebXML**  
320 **Header envelope and a content portion.**

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

- 322 1. Content-ID



323 2. Content-Length

324 3. Content-Type

325 The content portion contains a ebXML header document as defined by [3]. The ebXML header  
326 document within the content portion of the container MAY BE enhanced during transport,  
327 provided it has not been digitally signed. Any change in the size of the ebXML header document  
328 must be reflected in Content-Length header of the ebXML Message Envelope and ebXML  
329 Header envelope.

#### 330 4.5.1 Content-ID

331 The Content-ID MIME header is used to uniquely identify this container as the ebXML header  
332 envelope. There is only one possible value to associate with this header "ebxmlheader". An  
333 example usage follows:

```
334 Content-ID: ebxmlheader
```

#### 335 OPEN ISSUE 3.0

336 **ebXML's use of the Content-ID header field, as described above, is in conflict with the**  
337 **usage of this header field as defined in RFC 2045, which states, "Like the Message-ID**  
338 **values, Content-ID values must be generated to be world-unique."**  
339

340 **One possible solution is to replace the Content-ID header field with the Content-**  
341 **Description header field and use as specified above.**

#### 342 4.5.2 Content-Length

343 The Content-Length header contains a decimal value used to identify the total number of  
344 OCTETS contained in the ebXML header document residing in the content portion of the  
345 container. Example:

```
346 Content-Length: 4208
```

#### 347 4.5.3 Content-Type

348 The Content-type for an ebXML header is identified with the value "application/xml", as defined  
349 in RFC2376. An example of this content-type is:

```
350 Content-type: application/xml
```

#### 351 4.5.3.1 Optional Support for Signed Headers

352 Implementers are free to support digitally signed ebXML header documents. Digitally signed  
353 ebXML headers must be identified with the appropriate Content-Type and structure appropriate  
354 for the cryptographic tool used. In the case of S/MIME, the content-type must contain the correct  
355 value and attributes as specified in RFC 2633; in the case of OpenPGP, the content-type must  
356 contain the correct values and attributes specified in RFC 2015.

357 Implementers must follow the guidelines specified in RFC 2633 and RFC 2015 for creating and  
358 processing digitally signed objects.

359 If XML Dsig is used then implementers are expected to follow the specifications contained in the  
360 W3C Recommendation for XML Dsig.



#### 361 4.5.4 Complete Example of an ebXML Header container

362 The following represents an example of an ebXML header envelope and ebXML header  
363 document:

364	Content-ID: ebxmlheader	-----		
365				
366	Content-Length: 2048	-----	ebXML Header Envelope	
367	Content-Type: application/xml	-----		ebXML Header Container
368				
369	<ebXMLHeaderDocument>	-----		
370	<MessageHeader>.....		ebXML header Document	
371	</MessageHeader>			
372	</ebXMLHeaderDocument>	-----		

373 **NOTE: A REAL ebXML Header example here prior to final release as an approved**  
374 **specification.**

#### 375 4.6 Payload Container Specifications

376 The payload container of Message is optional. The ebXML header document contains a Message  
377 Manifest that identifies whether a payload container is present or not. If the Message Manifest of  
378 the ebXML header contains no entries then the ebXML payload container will not be present in  
379 the ebXML Message.

380 However, if the Message Manifest of the ebXML header indicates that a payload is present it **will**  
381 **consist of a MIME header portion, referred to as the ebXML payload envelope and a**  
382 **content portion.**

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

- 384 4. Content-ID
- 385 5. Content-Length
- 386 6. Content-Type

387

388 The content portion contains whatever structure and content two consenting parties agree to  
389 exchange. ebXML makes no provision nor limits in any way the structure or content of payloads.  
390 Payloads may contain simple plain text object or complex nested multipart objects. This is the  
391 implementers decision.

#### 392 4.6.1 Content-ID

393 The Content-ID MIME header is used to uniquely identify this container as the ebXML payload  
394 envelope. There is only one possible value to associate with this header "ebxmlpayload". An  
395 example usage follows:

```
396 Content-ID: ebxmlpayload
```

#### 397 **OPEN ISSUE 4.0**

398 **ebXML's use of the Content-ID header field, as described above, is in conflict with the**  
399 **usage of this header field as defined in RFC 2045, which states "Like the Message-ID**  
400 **values, Content-ID values must be generated to be world-unique."**

401

402 **One possible solution is to replace the Content-ID header field with the Content-**  
403 **Description header field and use as specified above.**



404 **4.6.2 Content-Length**

405 The Content-Length header contains a decimal value used to identify the total number of  
406 OCTETS contained in the content portion of the payload container. Example:

```
407 Content-Length: 5012
```

408 **4.6.3 Content-Type**

409 The Content-type for an ebXML header is determined by the implementer and is used to identify  
410 with the type of data contained in the content portion of the payload container.

```
411 Content-Type: application/xml
```

412 **4.6.3.1 Optional Support for Signed and Encrypted Payloads**

413 Implementers are free to support encrypted and digitally signed payloads. Digitally signed and/or  
414 encrypted payloads must be identified with the appropriate Content-Type and structure  
415 appropriate for the cryptographic tool used. In the case of S/MIME, the content-type must contain  
416 the correct value and attributes as specified in RFC 2633; in the case of OpenPGP, the content-  
417 type must contain the correct values and attributes specified in RFC 2015.

418 Implementers must follow the guidelines specified in RFC 2633 and RFC 2015 for creating and  
419 processing encrypted and digitally signed objects.

420 If XML Dsig is used then implementers are expected to follow the specifications contained in the  
421 W3C Recommendation for XML Dsig.

422 **4.6.4 Complete Example of an ebXML Payload container**

423 The following represents an example of an ebXML payload envelope and ebXML payload  
424 document:

425	Content-ID: ebxmlpayload	-----	ebXML Payload Envelope	ebXML Payload Container
426	Content-Length: 4096	-----		
427	Content-Type: application/xml	-----		
428				
429	<Invoice>	-----	ebXML Payload	
430	<Invoicedata>.....	-----		
431	</Invoicedata>	-----		
432	</Invoice>	-----		

433 **4.7 Complete Example of an ebXML Message enveloped using**  
434 **multipart/related content-type sent via HTTP POST**

435 **NOTE: The following example is representative of the ebXML packaging structure ONLY.**  
436 **ebXML headers used in this example are not to be construed as ACTUAL ebXML header**  
437 **structures. The "official" format and content of ebXML headers is defined in ebXML**  
438 **Transport, Routing and Packaging: Message Header Specification version x.x. Published**  
439 **dd mmmm 2000 [3].**

440 Following is a complete example of an ebXML Message sent via HTTP POST method:

```
441 POST /ebxmlhandler HTTP/1.1
442 Accept: multipart/related
443 Accept-Language: en-us
444 Content-Type: multipart/related; type=ebxml; version=0.1; boundary=-----
445 -----7d02a82e5f8
446 Accept-Encoding: gzip, deflate
```



```
447 User-Agent: Group 8760 InsideAgent
448 Host: localhost:9090
449 Content-Length: 9293
450 Connection: Keep-Alive
451
452 -----7d02a82e5f8
453 Content-ID: ebxmlheader
454 Content-Length: 211
455 Content-Type: application/xml
456
457 <?xml version="1.0" encoding="UTF-8"?>
458 <ebXMLMessageHeader xmlns='http://www.xml.org/ebXMLStds/ebXMLMessageHeaderv1'>
459   <Version>1.0</Version>
460   <MessageType>Request</MessageType>
461   <ServiceType>Payroll</ServiceType>
462   <Intent>RecordCommission</Intent>
463 </ebXMLMessageHeader>
464 -----7d02a82e5f8
465 Content-ID: ebxmlpayload
466 Content-Length: 7517
467 Content-Type: text/xml
468
469 <?xml version="1.0" encoding="UTF-8"?>
470 <!-- edited with XML Spy v2.5 - http://www.xmlspy.com -->
471 <HITISMessage xmlns="" Version="1.0">
472   <Header OriginalBodyRequested="false" ImmediateResponseRequired="true">
473     <FromURI>http://www.pms.com/HITISInterface</FromURI>
474     <ToURI>http://www.crs.com/HITISInterface</ToURI>
475     <ReplyToURI>http://www.pms.com/HITISInterface</ReplyToURI>
476     <MessageID>1234567890</MessageID>
477     <OriginalMessageID>1234567890</OriginalMessageID>
478     <TimeStamp>1999-11-10T10:23:44</TimeStamp>
479     <Token>1234-567-8901</Token>
480     <!--Token to be assigned in response to HITISRegister-->
481   </Header>
482   <Body>
483     <HITISOperation OperationName="CommissionEventsUpdate">
484       <CommissionEvents>
485         <CommissionEvent>
486           <ConfirmationID>18097YZ</ConfirmationID>
487           <ConfirmationOriginatorCode>DBZ223</ConfirmationOriginatorCode>
488           <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
489           <ReservationID>098787818097YZ</ReservationID>
490           <HotelReference>
491             <ChainCode>HI234</ChainCode>
492             <HotelCode>1234STL</HotelCode>
493           </HotelReference>
494           <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
495           <StayDateRange>
496             <StartInstant>20000122</StartInstant>
497             <Duration>00000003T000000</Duration>
498           </StayDateRange>
499           <GuestNames>
500             <NameInfo>
501               <NamePrefix>Mr.</NamePrefix>
502               <NameFirst>John</NameFirst>
503               <NameMiddle>Q.</NameMiddle>
504               <NameSur>jones</NameSur>
505               <NameSuffix>Jr.</NameSuffix>
506               <NameTitle>Professor</NameTitle>
507               <NameOrdered>JohnJones</NameOrdered>
508             </NameInfo>
509             <NameInfo>
510               <NamePrefix>Mrs.</NamePrefix>
511               <NameFirst>Sally</NameFirst>
512               <NameMiddle>T.</NameMiddle>
513               <NameSur>Jones</NameSur>
514               <NameSuffix/>
515               <NameTitle/>
516               <NameOrdered>SallyJones</NameOrdered>
517             </NameInfo>
518           </GuestNames>
519           <ProfileCertification CertificationType="ARC">
520             <CertificationID>67TR901-AZ</CertificationID>
521           </ProfileCertification>

```



```
522 <ProfileReference>
523 <!--Profile to be inserted as a reusable component-->
524 <Profile/>
525 </ProfileReference>
526 <Commissions>
527 <Commission CommissionStatusType="Full">
528 <CommissionableAmount>
529 <Currency>
530 <CurrencyCode>USD</CurrencyCode>
531 <Amount>185.00</Amount>
532 </Currency>
533 </CommissionableAmount>
534 <PrepaidAmount>
535 <Currency>
536 <CurrencyCode>USD</CurrencyCode>
537 <Amount>12.00</Amount>
538 </Currency>
539 </PrepaidAmount>
540 <CommissionPercent>0.0525</CommissionPercent>
541 <FlatCommission>not applicable<Currency>
542 <CurrencyCode>USD</CurrencyCode>
543 <Amount>00.00</Amount>
544 </Currency>
545 </FlatCommission>
546 <Comment>Default percentage commission agreement</Comment>
547 <CommissionReasonCode>7930</CommissionReasonCode>
548 <BillToID>HOTEL7890</BillToID>
549 <HotelReference>
550 <ChainCode>HI234</ChainCode>
551 <HotelCode>1234STL</HotelCode>
552 </HotelReference>
553 </Commission>
554 <Commission CommissionStatusType="Partial">
555 <CommissionableAmount>
556 <Currency>
557 <CurrencyCode>USD</CurrencyCode>
558 <Amount>185.00</Amount>
559 </Currency>
560 </CommissionableAmount>
561 <PrepaidAmount>
562 <Currency>
563 <CurrencyCode>USD</CurrencyCode>
564 <Amount>00.00</Amount>
565 </Currency>
566 </PrepaidAmount>
567 <Comment>This commission per agreement with Travel Agents,
568 Inc.</Comment>
569 <CommissionPercent>00.00</CommissionPercent>
570 <FlatCommission>
571 <Currency>
572 <CurrencyCode>USD</CurrencyCode>
573 <Amount>10.00</Amount>
574 </Currency>
575 </FlatCommission>
576 <CommissionReasonCode>7930</CommissionReasonCode>
577 <BillToID>HOTEL7890</BillToID>
578 <HotelReference>
579 <ChainCode>HI234</ChainCode>
580 <HotelCode>1234STL</HotelCode>
581 </HotelReference>
582 </Commission>
583 </Commissions>
584 </CommissionEvent>
585 <CommissionEvent>
586 <ConfirmationID/>
587 <ConfirmationOriginatorCode/>
588 <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
589 <ReservationID>09878783276XY</ReservationID>
590 <HotelReference>
591 <ChainCode>BASS123</ChainCode>
592 <HotelCode>1234STL</HotelCode>
593 </HotelReference>
594 <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
595 <StayDateRange>
596 <StartInstant>20000122</StartInstant>
```



597  
598  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
670  
671

```
<Duration>00000003T000000</Duration>
</StayDateRange>
<GuestNames>
  <NameInfo>
    <NamePrefix>Mr.</NamePrefix>
    <NameFirst>Kevin</NameFirst>
    <NameMiddle>R.</NameMiddle>
    <NameSur>Smithson</NameSur>
    <NameSuffix>Jr.</NameSuffix>
    <NameTitle>Professor</NameTitle>
    <NameOrdered> Kevin Smithson</NameOrdered>
  </NameInfo>
  <NameInfo>
    <NamePrefix>Miss</NamePrefix>
    <NameFirst>Mary</NameFirst>
    <NameMiddle>T.</NameMiddle>
    <NameSur>Smithson</NameSur>
    <NameSuffix>esq.</NameSuffix>
    <NameTitle>Professor</NameTitle>
    <NameOrdered> MarySmithson</NameOrdered>
  </NameInfo>
</GuestNames>
<ProfileCertification CertificationType="ARC">
  <CertificationID>67TR901-AZ</CertificationID>
</ProfileCertification>
<ProfileReference>
  <Profile/>
</ProfileReference>
<Commissions>
  <Commission CommissionStatusType="Full">
    <CommissionableAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>185.00</Amount>
      </Currency>
    </CommissionableAmount>
    <PrepaidAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>12.00</Amount>
      </Currency>
    </PrepaidAmount>
    <CommissionPercent>0.0525</CommissionPercent>
    <FlatCommission>not applicable<Currency>
      <CurrencyCode>USD</CurrencyCode>
      <Amount>00.00</Amount>
    </Currency>
    </FlatCommission>
    <Comment>Default percentage commission agreement</Comment>
    <CommissionReasonCode>7930</CommissionReasonCode>
    <BillToID>HOTEL7890</BillToID>
    <HotelReference>
      <ChainCode>HI234</ChainCode>
      <HotelCode>1234STL</HotelCode>
    </HotelReference>
  </Commission>
  <Commission CommissionStatusType="Partial">
    <CommissionableAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>185.00</Amount>
      </Currency>
    </CommissionableAmount>
    <PrepaidAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>00.00</Amount>
      </Currency>
    </PrepaidAmount>
    <Comment>Flat commission per agreement with TA</Comment>
    <CommissionPercent>00.00</CommissionPercent>
    <FlatCommission>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>10.00</Amount>
      </Currency>
    </FlatCommission>
  </Commission>
</Commissions>
</ProfileReference>
</ProfileCertification>
</GuestNames>
</StayDateRange>
</Duration>
```



672  
673  
674  
675  
676  
677  
678  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688

```
        </Currency>  
        </FlatCommission>  
        <CommissionReasonCode>7930</CommissionReasonCode>  
        <BillToID>HOTEL7890</BillToID>  
        <HotelReference>  
            <ChainCode>HI234</ChainCode>  
            <HotelCode>1234STL</HotelCode>  
        </HotelReference>  
    </Commission>  
  </Commissions>  
</CommissionEvent>  
</CommissionEvents>  
</HITISOperation>  
</Body>  
</HITISMessage>  
-----7d02a82e5f8--
```

689  
690

#### 4.8 Complete Example of an ebXML Message enveloped using multipart/related content-type sent via SMTP

691  
692  
693  
694  
695

**NOTE: The following example is representative of the ebXML packaging structure ONLY. ebXML headers used in this example are not to be construed as ACTUAL ebXML header structures. The "official" format and content of ebXML headers is defined in ebXML Transport, Routing and Packaging: Message Header Specification version x.x. Published dd mmmm 2000 [3].**

696

**Also, the default Content-transfer-encoding type of 7BIT is being used in this message.**

697

Following is a complete example of an ebXML Message sent via SMTP:

698  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
730  
731  
732  
733  
734  
735

```
From dick@8760.com Sun May 7 17:01:14 2000  
Received: from granger.mail.mindspring.net by alpha2000.tech-comm.com;  
(8.8.5/1.1.8.2/05Jun95-1217PM)  
 id RAA32702; Sun, 7 May 2000 17:01:13 -0500 (CDT)  
Received: from gamma (user-33qt101.dialup.mindspring.com [199.174.132.21])  
 by granger.mail.mindspring.net (8.9.3/8.8.5) with SMTP id SAA11942  
 for <ebxmlhandler@8760.com>; Sun, 7 May 2000 18:11:14 -0400 (EDT)  
From: "Dick Brooks (E)" <dick@8760.com>  
To: <ebxmlhandler@8760.com>  
Subject: OTA Commission Event  
Date: Sun, 7 May 2000 17:07:38 -0500  
Message-ID: <NDBBIOBLMLCDOHCHIKMGKEEIDAAA.dick@8760.com>  
MIME-Version: 1.0  
X-Priority: 3 (Normal)  
X-MSMail-Priority: Normal  
X-Mailer: Microsoft Outlook IMO, Build 9.0.2416 (9.0.2910.0)  
Importance: Normal  
X-MimeOLE: Produced By Microsoft MimeOLE V5.00.2314.1300  
Content-Length: 8081  
Content-Type: multipart/related; type="ebxml"; version="0.1";  
 boundary="-----_NextPart_000_0005_01BFB846.BF7FABA0"  
  
-----_NextPart_000_0005_01BFB846.BF7FABA0  
Content-Type: application/xml  
Content-ID: ebxmlheader  
Content-Length: 272  
  
<?xml version="1.0" encoding="UTF-8"?>  
<ebXMLMessageHeader xmlns='http://www.xml.org/ebXMLStds/ebXMLMessageHeaderv1'>  
<Version>1.0</Version>  
<MessageType>Request</MessageType>  
<ServiceType>Payroll</ServiceType>  
<Intent>RecordCommission</Intent>  
</ebXMLMessageHeader>  
-----_NextPart_000_0005_01BFB846.BF7FABA0  
Content-Type: text/xml  
Content-ID: ebxmlpayload
```



736  
737  
738  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
810

```
Content-Length: 7515

<?xml version="1.0" encoding="UTF-8"?>
<!-- edited with XML Spy v2.5 - http://www.xmlspy.com -->
<HITISMessage xmlns="" Version="1.0">
  <Header OriginalBodyRequested="false" ImmediateResponseRequired="true">
    <FromURI>http://www.pms.com/HITISInterface</FromURI>
    <ToURI>http://www.crs.com/HITISInterface</ToURI>
    <ReplyToURI>http://www.pms.com/HITISInterface</ReplyToURI>
    <MessageID>1234567890</MessageID>
    <OriginalMessageID>1234567890</OriginalMessageID>
    <TimeStamp>1999-11-10T10:23:44</TimeStamp>
    <Token>1234-567-8901</Token>
    <!--Token to be assigned in response to HITISRegister-->
  </Header>
  <Body>
    <HITISOperation OperationName="CommissionEventsUpdate">
      <CommissionEvents>
        <CommissionEvent>
          <ConfirmationID>18097YZ</ConfirmationID>
          <ConfirmationOriginatorCode>DBZ223</ConfirmationOriginatorCode>
          <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
          <ReservationID>098787818097YZ</ReservationID>
          <HotelReference>
            <ChainCode>HI234</ChainCode>
            <HotelCode>1234STL</HotelCode>
          </HotelReference>
          <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
          <StayDateRange>
            <StartInstant>20000122</StartInstant>
            <Duration>00000003T000000</Duration>
          </StayDateRange>
          <GuestNames>
            <NameInfo>
              <NamePrefix>Mr.</NamePrefix>
              <NameFirst>John</NameFirst>
              <NameMiddle>Q.</NameMiddle>
              <NameSur>jones</NameSur>
              <NameSuffix>Jr.</NameSuffix>
              <NameTitle>Professor</NameTitle>
              <NameOrdered>JohnJones</NameOrdered>
            </NameInfo>
            <NameInfo>
              <NamePrefix>Mrs.</NamePrefix>
              <NameFirst>Sally</NameFirst>
              <NameMiddle>T.</NameMiddle>
              <NameSur>Jones</NameSur>
              <NameSuffix/>
              <NameTitle/>
              <NameOrdered>SallyJones</NameOrdered>
            </NameInfo>
          </GuestNames>
          <ProfileCertification CertificationType="ARC">
            <CertificationID>67TR901-AZ</CertificationID>
          </ProfileCertification>
          <ProfileReference>
            <!--Profile to be inserted as a reusable component-->
            <Profile/>
          </ProfileReference>
          <Commissions>
            <Commission CommissionStatusType="Full">
              <CommissionableAmount>
                <Currency>
                  <CurrencyCode>USD</CurrencyCode>
                  <Amount>185.00</Amount>
                </Currency>
              </CommissionableAmount>
              <PrepaidAmount>
                <Currency>
                  <CurrencyCode>USD</CurrencyCode>
                  <Amount>12.00</Amount>
                </Currency>
              </PrepaidAmount>
              <CommissionPercent>0.0525</CommissionPercent>
              <FlatCommission>not applicable<Currency>
```

```

811         <CurrencyCode>USD</CurrencyCode>
812         <Amount>00.00</Amount>
813     </Currency>
814 </FlatCommission>
815 <Comment>Default percentage commission agreement</Comment>
816 <CommissionReasonCode>7930</CommissionReasonCode>
817 <BillToID>HOTEL7890</BillToID>
818 <HotelReference>
819     <ChainCode>HI234</ChainCode>
820     <HotelCode>1234STL</HotelCode>
821 </HotelReference>
822 </Commission>
823 <Commission CommissionStatusType="Partial">
824     <CommissionableAmount>
825         <Currency>
826             <CurrencyCode>USD</CurrencyCode>
827             <Amount>185.00</Amount>
828         </Currency>
829     </CommissionableAmount>
830 <PrepaidAmount>
831     <Currency>
832         <CurrencyCode>USD</CurrencyCode>
833         <Amount>00.00</Amount>
834     </Currency>
835 </PrepaidAmount>
836 <Comment>This commission per agreement with Travel Agents,
837 Inc.</Comment>
838     <CommissionPercent>00.00</CommissionPercent>
839 <FlatCommission>
840     <Currency>
841         <CurrencyCode>USD</CurrencyCode>
842         <Amount>10.00</Amount>
843     </Currency>
844 </FlatCommission>
845 <CommissionReasonCode>7930</CommissionReasonCode>
846 <BillToID>HOTEL7890</BillToID>
847 <HotelReference>
848     <ChainCode>HI234</ChainCode>
849     <HotelCode>1234STL</HotelCode>
850 </HotelReference>
851 </Commission>
852 </Commissions>
853 </CommissionEvent>
854 <CommissionEvent>
855     <ConfirmationID/>
856     <ConfirmationOriginatorCode/>
857     <CommissionOriginatorCode>3457YTXV</CommissionOriginatorCode>
858     <ReservationID>09878783276XY</ReservationID>
859     <HotelReference>
860         <ChainCode>BASS123</ChainCode>
861         <HotelCode>1234STL</HotelCode>
862     </HotelReference>
863     <OriginalBookingDate>19991223T17:53:22</OriginalBookingDate>
864     <StayDateRange>
865         <StartInstant>20000122</StartInstant>
866         <Duration>00000003T000000</Duration>
867     </StayDateRange>
868     <GuestNames>
869         <NameInfo>
870             <NamePrefix>Mr.</NamePrefix>
871             <NameFirst>Kevin</NameFirst>
872             <NameMiddle>R.</NameMiddle>
873             <NameSur>Smithson</NameSur>
874             <NameSuffix>Jr.</NameSuffix>
875             <NameTitle>Professor</NameTitle>
876             <NameOrdered> Kevin Smithson</NameOrdered>
877         </NameInfo>
878         <NameInfo>
879             <NamePrefix>Miss</NamePrefix>
880             <NameFirst>Mary</NameFirst>
881             <NameMiddle>T.</NameMiddle>
882             <NameSur>Smithson</NameSur>
883             <NameSuffix>esq.</NameSuffix>
884             <NameTitle>Professor</NameTitle>
885             <NameOrdered> MarySmithson</NameOrdered>

```



886  
887  
888  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
950  
951  
952  
953  
954  
955  
956

```
</NameInfo>
</GuestNames>
<ProfileCertification CertificationType="ARC">
  <CertificationID>67TR901-AZ</CertificationID>
</ProfileCertification>
<ProfileReference>
  <Profile/>
</ProfileReference>
<Commissions>
  <Commission CommissionStatusType="Full">
    <CommissionableAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>185.00</Amount>
      </Currency>
    </CommissionableAmount>
    <PrepaidAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>12.00</Amount>
      </Currency>
    </PrepaidAmount>
    <CommissionPercent>0.0525</CommissionPercent>
    <FlatCommission>not applicable<Currency>
      <CurrencyCode>USD</CurrencyCode>
      <Amount>00.00</Amount>
    </Currency>
    </FlatCommission>
    <Comment>Default percentage commission agreement</Comment>
    <CommissionReasonCode>7930</CommissionReasonCode>
    <BillToID>HOTEL7890</BillToID>
    <HotelReference>
      <ChainCode>HI234</ChainCode>
      <HotelCode>1234STL</HotelCode>
    </HotelReference>
  </Commission>
  <Commission CommissionStatusType="Partial">
    <CommissionableAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>185.00</Amount>
      </Currency>
    </CommissionableAmount>
    <PrepaidAmount>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>00.00</Amount>
      </Currency>
    </PrepaidAmount>
    <Comment>Flat commission per agreement with TA</Comment>
    <CommissionPercent>00.00</CommissionPercent>
    <FlatCommission>
      <Currency>
        <CurrencyCode>USD</CurrencyCode>
        <Amount>10.00</Amount>
      </Currency>
    </FlatCommission>
    <CommissionReasonCode>7930</CommissionReasonCode>
    <BillToID>HOTEL7890</BillToID>
    <HotelReference>
      <ChainCode>HI234</ChainCode>
      <HotelCode>1234STL</HotelCode>
    </HotelReference>
  </Commission>
</Commissions>
</CommissionEvent>
</CommissionEvents>
</HITISOperation>
</Body>
</HITISMessage>
-----_NextPart_000_0005_01BFB846.BF7FABA0--
```



## 957 **5 Security Considerations**

958 Implementers should examine carefully the security features of each transport. In the case of  
959 HTTP, Implementers are encouraged to use Realm Security, using basic authentication for  
960 access controls and SSL to protect sensitive information.

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

## 963 **6 References**

- 964 [1] ebXML Transport, Routing & Packaging Document Control and Index  
965 [2] ebXML Transport, Routing and Packaging: Overview and Requirements  
966 [3] ebXML Transport, Routing and Packaging: Message Header Specification  
967 [4] XMTP - Extensible Mail Transport Protocol  
968 <http://www.openhealth.org/documents/xmtp.htm>  
969

## 970 **7 Acknowledgments**

- 971 Jonathan Borden - Author of XMTP  
972 Jon Bosak - Sun  
973 David Burdett - Commerce One  
974 Rik Drummond - Drummond Group  
975 Christopher Ferris - Sun  
976 Ian Jones - British Telecom  
977 Henry Lowe - OMG  
978 Jim McCarthy - webXI  
979 Bob Miller - GEIS  
980 Dale Moberg - Sterling Commerce  
981 Prasad Yendluri - Vitria

## 982 **9 Authors' Address**

- 983 Dick Brooks  
984 Group 8760  
985 110 12th Street North  
986 Suite F103  
987 Birmingham, Alabama 35203  
988 Telephone: 205-250-8053  
989 E-mail: dick@8760.com  
990



991 Nicholas Kassem  
992 Java Software, Sun Microsystems  
993 901 San Antonio Road, MS CUP02-201  
994 Palo Alto, CA 94303-4900  
  
995 Telephone: 408-863-3535  
996 E-mail: Nick.Kassem@eng.sun.com