



1 Web Services Reliable Messaging 2 (WS-Reliable Messaging)

3 Committee Draft 01, September 26th 2005

4 **Document identifier:**

5 wsrm-1.1-spec-cd-01

6 **Location:**

7 <http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1-spec-cd-01.pdf>

8 **Editors:**

9 Gilbert Pilz, BEA <gilbert.pilz@bea.com>

10 Doug Davis, IBM <dug@us.ibm.com>

11 Anish Karmarkar, Oracle <Anish.Karmarkar@oracle.com>

12 TBD

13 **Contributors:**

14 TBD

15 **Abstract:**

16 This specification (WS-ReliableMessaging) describes a protocol that allows messages to
17 be delivered reliably between distributed applications in the presence of software
18 component, system, or network failures. The protocol is described in this specification in a
19 transport-independent manner allowing it to be implemented using different network
20 technologies. To support interoperable Web services, a SOAP binding is defined within
21 this specification.

22 The protocol defined in this specification depends upon other Web services specifications
23 for the identification of service endpoint addresses and policies. How these are identified
24 and retrieved are detailed within those specifications and are out of scope for this
25 document.

26 By using the SOAP [[SOAP](#)] and WSDL [[WSDL](#)] extensibility model, SOAP-based and
27 WSDL-based specifications are designed to be composed with each other to define a rich
28 Web services environment. As such, WS-ReliableMessaging by itself does not define all
29 the features required for a complete messaging solution. WS-ReliableMessaging is a
30 building block that is used in conjunction with other specifications and application-specific
31 protocols to accommodate a wide variety of protocols related to the operation of
32 distributed Web services.

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

33 **Status:**

34 This document is a Committee Draft.

35 This document was last revised or approved by the OASIS WS-RX Technical Committee
36 on the above date. The level of approval is also listed above. Check the current location
37 noted above for possible later revisions of this document.

38 For information on whether any patents have been disclosed that may be essential to
39 implementing this specification and any offers of patent licensing terms please refer to the
40 Intellectual Property Rights section of the Technical Committee web page
41 (<http://www.oasis-open.org/committees/ws-rx/ipr.php>).

42 Table of Contents

43	1INTRODUCTION.....	5
44	1.1GOALS AND REQUIREMENTS.....	5
45	1.1.1Requirements.....	5
46	1.2NOTATIONAL CONVENTIONS.....	5
47	1.3NAMESPACE.....	6
48	1.4COMPLIANCE.....	7
49	2RELIABLE MESSAGING MODEL.....	8
50	2.1GLOSSARY.....	9
51	2.2PROTOCOL PRECONDITIONS.....	10
52	2.3PROTOCOL INVARIANTS.....	10
53	2.4EXAMPLE MESSAGE EXCHANGE.....	11
54	3RM PROTOCOL ELEMENTS.....	13
55	3.1SEQUENCES.....	13
56	3.2SEQUENCE ACKNOWLEDGEMENT.....	15
57	3.3REQUEST ACKNOWLEDGEMENT.....	17
58	3.4SEQUENCE CREATION.....	19
59	3.5SEQUENCE TERMINATION.....	22
60	3.6CLOSING A SEQUENCE.....	23
61	4FAULTS.....	26
62	4.1SEQUENCEFAULT ELEMENT.....	28
63	4.2SEQUENCE TERMINATED.....	29
64	4.3UNKNOWN SEQUENCE.....	29
65	4.4INVALID ACKNOWLEDGEMENT.....	29
66	4.5MESSAGE NUMBER ROLLOVER.....	30
67	4.6LAST MESSAGE NUMBER EXCEEDED.....	30
68	4.7CREATE SEQUENCE REFUSED.....	31
69	4.8SEQUENCE CLOSED.....	31
70	5SECURITY CONSIDERATIONS.....	32
71	6REFERENCES.....	34
72	6.1NORMATIVE.....	34

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

73	6.2NON-NORMATIVE.....	34
74	APPENDIX A.SHEMA	36
75	APPENDIX B.MESSAGE EXAMPLES.....	41
76	B.1.CREATE SEQUENCE.....	42
77	B.2. INITIAL TRANSMISSION.....	44
78	B.3.FIRST ACKNOWLEDGEMENT.....	46
79	B.4.RETRANSMISSION.....	47
80	B.5.TERMINATION.....	48
81	APPENDIX C.WSDL.....	50
82	APPENDIX D.ACKNOWLEDGMENTS.....	52
83	APPENDIX E.REVISION HISTORY.....	53
84	APPENDIX F.NOTICES.....	54

85 **1 Introduction**

86 It is often a requirement for two Web services that wish to communicate to do so
87 reliably in the presence of software component, system, or network failures. The
88 primary goal of this specification is to create a modular mechanism for reliable
89 message delivery. It defines a messaging protocol to identify, track, and manage the
90 reliable delivery of messages between exactly two parties, a source and a
91 destination. It also defines a SOAP binding that is required for interoperability.
92 Additional bindings may be defined.

93 This mechanism is extensible allowing additional functionality, such as security, to be
94 tightly integrated. This specification integrates with and complements the WS-
95 Security, WS-Policy, and other Web services specifications. Combined, these allow
96 for a broad range of reliable, secure messaging options.

97 **1.1 Goals and Requirements**

98 **1.1.1 Requirements**

99 **1.2 Notational Conventions**

100 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
101 "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this
102 document are to be interpreted as described in RFC 2119 [[KEYWORDS](#)].

103 This specification uses the following syntax to define normative outlines for
104 messages:

- 105 • The syntax appears as an XML instance, but values in italics indicate data types instead
106 of values.
- 107 • Characters are appended to elements and attributes to indicate cardinality:
 - 108 ○ "?" (0 or 1)
 - 109 ○ "*" (0 or more)
 - 110 ○ "+" (1 or more)
- 111 • The character "|" is used to indicate a choice between alternatives.
- 112 • The characters "[" and "]" are used to indicate that contained items are to be treated as a
113 group with respect to cardinality or choice.

- 114 • An ellipsis (i.e. "...") indicates a point of extensibility that allows other child, or attribute,
115 content. Additional children elements and/or attributes MAY be added at the indicated
116 extension points but MUST NOT contradict the semantics of the parent and/or owner,
117 respectively. If an extension is not recognized it SHOULD be ignored.
- 118 • XML namespace prefixes (See Section [Namespace](#)) are used to indicate the namespace
119 of the element being defined.

120 1.3 Namespace

121 The XML namespace [[XML-ns](#)] URI that MUST be used by implementations of this
122 specification is:

123 <http://docs.oasis-open.org/wsrn/200510/>

124 Table 1 lists XML namespaces that are used in this specification. The choice of any
125 namespace prefix is arbitrary and not semantically significant.

126 The following namespaces are used in this document:

127 *Table Number range Table*

Prefix	Namespace
s	http://www.w3.org/2003/05/soap-envelope
S11	http://schemas.xmlsoap.org/soap/envelope/
wsrn	http://docs.oasis-open.org/wsrn/200510/
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
xs	http://www.w3.org/2001/XMLSchema

128 The normative schema for WS-Reliable Messaging can be found at:

129 <http://docs.oasis-open.org/wsrn/200510/wsrn.xsd>

130 All sections explicitly noted as examples are informational and are not to be
131 considered normative.

132 If an action URI is used, and one is not already defined per the rules of the WS-
133 Addressing specification [[WS-Addressing](#)], then the action URI MUST consist of the
134 reliable messaging namespace URI concatenated with the element name. For
135 example:

File name

Date

136 <http://docs.oasis-open.org/wsrn/200510/SequenceAcknowledgement>

137 **1.4 Compliance**

138 An implementation is not compliant with this specification if it fails to satisfy one or
139 more of the MUST or REQUIRED level requirements defined herein. A SOAP Node
140 MUST NOT use the XML namespace identifier for this specification (listed in
141 Section [Namespace](#)) within SOAP Envelopes unless it is compliant with this
142 specification.

143 Normative text within this specification takes precedence over normative outlines,
144 which in turn take precedence over the XML Schema [[XML Schema Part 1](#), [Part 2](#)]
145 descriptions.

146 **2 Reliable Messaging Model**

147 Many errors may interrupt a conversation. Messages may be lost, duplicated or
148 reordered. Further the host systems may experience failures and lose volatile state.

149

150 The WS-ReliableMessaging specification defines an interoperable protocol that
151 requires a Reliable Messaging (RM) Source and Reliable Messaging (RM) Destination
152 to ensure that each message transmitted by the RM Source is successfully received
153 by an RM Destination, or barring successful receipt, that an RM Source can, except in
154 the most extreme circumstances, accurately determine the disposition of each
155 message transmitted as perceived by the RM Destination, so as to resolve any in-
156 doubt status.

157 In addition, The protocol allows the RM Source and RM Destination to provide their
158 respective Application Source and Application Destination a guarantee that a
159 message that is sent by an Application Source will be delivered to the Application
160 Destination.

161 This guarantee is specified as a delivery assurance. It is the responsibility of the RM
162 Source and RM Destination to fulfill the delivery assurances on behalf of their
163 respective Application counterparts, or raise an error. The protocol defined here
164 allows endpoints to meet this guarantee for the delivery assurances defined below.
165 However, the means by which these delivery assurances are manifested by either the
166 RM Source or RM Destination roles is an implementation concern, and is out of scope
167 of this specification.

168 Note that the underlying protocol defined in this specification remains the same
169 regardless of the delivery assurance.

170 Persistence considerations related to an endpoint's ability to satisfy the delivery
171 assurances defined below are the responsibility of the implementation and do not
172 affect the wire protocol. As such, they are out of scope of this specification.

173 There are four basic delivery assurances that endpoints can provide:

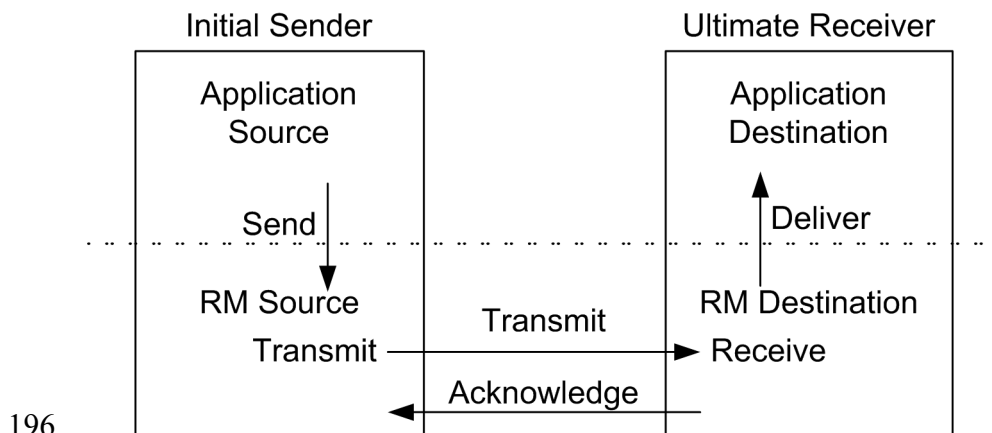
174 **AtMostOnce** Messages will be delivered at most once without duplication or an error
175 will be raised on at least one endpoint. It is possible that some messages in a
176 sequence may not be delivered.

177 **AtLeastOnce** Every message sent will be delivered or an error will be raised on at
178 least one endpoint. Some messages may be delivered more than once.

179 **ExactlyOnce** Every message sent will be delivered without duplication or an error
180 will be raised on at least one endpoint. This delivery assurance is the logical "and" of
181 the two prior delivery assurances.

182 **InOrder** Messages will be delivered in the order that they were sent. This delivery
183 assurance may be combined with any of the above delivery assurances. It requires
184 that the messages within a Sequence will be delivered in an order so that the
185 message numbers are monotonically increasing. Note that this assurance says
186 nothing about duplications or omissions. Note also that it is only applicable to
187 messages in the same Sequence. Cross Sequence ordering of messages is not in the
188 scope of this specification.

189 Figure 1 below illustrates the entities and events in a simple reliable message
190 exchange. First, the Application Source Sends a message for reliable delivery. The
191 Reliable Messaging (RM) Source accepts the message and Transmits it one or more
192 times. After receiving the message, the RM Destination Acknowledges it. Finally,
193 the RM Destination delivers the message to the Application Destination. The exact
194 roles the entities play and the complete meaning of the events will be defined
195 throughout this specification.



197 Figure 1: Reliable Messaging Model

198 2.1 Glossary

199 The following definitions are used throughout this specification:

200 **Endpoint:** A referencable entity, processor, or resource where Web service messages
201 are originated or targeted.

202 **Application Source:** The endpoint that Sends a message.

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

203 **Application Destination:** The endpoint to which a message is Delivered.
204 **Delivery Assurance:** The guarantee that the messaging infrastructure provides on
205 the delivery of a message.
206 **Receive:** The act of reading a message from a network connection and qualifying it
207 as relevant to RM Destination functions.
208 **RM Source:** The endpoint that transmits the message.
209 **RM Destination:** The endpoint that receives the message.
210 **Send:** The act of submitting a message to the RM Source for reliable delivery. The
211 reliability guarantee begins at this point.
212 **Deliver:** The act of transferring a message from the RM Destination to the
213 Application Destination. The reliability guarantee is fulfilled at this point.
214 **Transmit:** The act of writing a message to a network connection.
215 **Receive:** The act of reading a message from a network connection.
216 **Acknowledgement:** The communication from the RM Destination to the RM Source
217 indicating the successful receipt of a message.

218 **2.2 Protocol Preconditions**

219 The correct operation of the protocol requires that a number of preconditions **MUST**
220 be established prior to the processing of the initial sequenced message:

- 221 • The RM Source **MUST** have an endpoint reference that uniquely identifies the RM Destination
222 endpoint; correlations across messages addressed to the unique endpoint **MUST** be
223 meaningful.
- 224 • The RM Source **MUST** have knowledge of the destination's policies, if any, and the RM
225 Source **MUST** be capable of formulating messages that adhere to this policy.

226 If a secure exchange of messages is required, then the RM Source and RM
227 Destination **MUST** have a security context.

228 **2.3 Protocol Invariants**

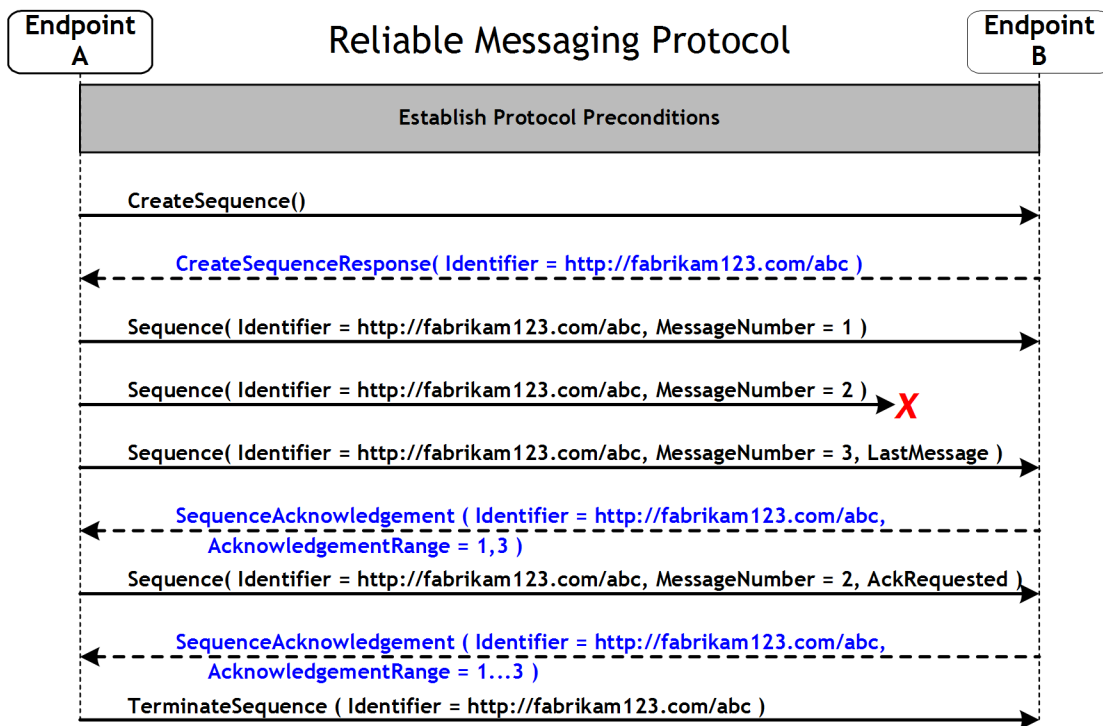
229 During the lifetime of the protocol, two invariants are **REQUIRED** for correctness:

- 230 • The RM Source **MUST** assign each reliable message a sequence number (defined below)
231 beginning at 1 and increasing by exactly 1 for each subsequent reliable message.

232 Every acknowledgement issued by the RM Destination MUST include within an
 233 acknowledgement range or ranges the sequence number of every message
 234 successfully received by the RM Destination and MUST exclude sequence numbers of
 235 any messages not yet received.

236 2.4 Example Message Exchange

237 Figure 2 illustrates a possible message exchange between two reliable messaging
 238 endpoints A and B.



239 Figure 2: The WS-ReliableMessaging Protocol

- 240 1. The protocol preconditions are established. These include policy exchange,
- 241 endpoint resolution, establishing trust.
- 242 2. The RM Source requests creation of a new Sequence.
- 243 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 244 4. The RM Source begins sending messages beginning with MessageNumber 1. In
- 245 the figure the RM Source sends 3 messages.

- 246 5. Since the 3rd message is the last in this exchange, the RM Source includes a
247 <wsrm:LastMessage> token.
- 248 6. The 2nd message is lost in transit.
- 249 7. The RM Destination acknowledges receipt of message numbers 1 and 3 in
250 response to the RM Source's <wsrm:LastMessage> token.
- 251 8. The RM Source retransmits the 2nd message. This is a new message on the
252 underlying transport, but since it has the same sequence identifier and message
253 number so the RM Destination can recognize it as equivalent to the earlier
254 message, in case both are received.
- 255 9. The RM Source includes an <wsrm:AckRequested> element so the RM Destination
256 will expedite an acknowledgement.
- 257 10. The RM Destination receives the second transmission of the message with
258 MessageNumber 2 and acknowledges receipt of message numbers 1, 2, and 3
259 which carried the <wsrm:LastMessage> token.
- 260 11. The RM Source receives this acknowledgement and sends a TerminateSequence
261 message to the RM Destination indicating that the sequence is completed and
262 reclaims any resources associated with the Sequence.
- 263 12. The RM Destination receives the TerminateSequence message indicating that the
264 RM Source will not be sending any more messages, and reclaims any resources
265 associated with the Sequence.
- 266 Now that the basic model has been outlined, the details of the elements used in this
267 protocol are now provided in Section 3.

268 **3 RM Protocol Elements**

269 The protocol elements define extensibility points at various places. Additional
270 children elements and/or attributes MAY be added at the indicated extension points
271 but MUST NOT contradict the semantics of the parent and/or owner, respectively. If a
272 receiver does not recognize an extension, the receiver SHOULD ignore the extension.

273 **3.1 Sequences**

274 The RM protocol uses a `<wsrm:Sequence>` header block to track and manage the
275 reliable delivery of messages. Messages for which the delivery assurance applies
276 MUST contain a `<wsrm:Sequence>` header block. Each Sequence MUST have a
277 unique `<wsrm:Identifier>` element and each message within a Sequence MUST
278 have a `<wsrm:MessageNumber>` element that increments by 1 from an initial value of
279 1. These values are contained within a `<wsrm:Sequence>` header block accompanying
280 each message being delivered in the context of a Sequence. In addition to mandatory
281 `<wsrm:Identifier>` and `<wsrm:MessageNumber>` elements, the header MAY include a
282 `<wsrm:LastMessage>` element.

283 There MUST be no more than one `<wsrm:Sequence>` header block in any message.

284 The purpose of the `<wsrm:LastMessage>` element is to signal to the RM Destination
285 that the message represents the last message in the Sequence.

286 A following exemplar defines its syntax:

```
287 <wsrm:Sequence ...>  
288   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
289   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>  
290   <wsrm:LastMessage/>?  
291   ...  
292 </wsrm:Sequence>
```

293 The following describes the content model of the Sequence header block.

294 `/wsrm:Sequence`

295 This is the element containing Sequence information for WS-ReliableMessaging. The
296 `<wsrm:Sequence>` element MUST be understood by the RM Destination. The `<wsrm:Sequence>`
297 element MUST have a `mustUnderstand` attribute with a value 1/true from the namespace
298 corresponding to the version of SOAP to which the `<wsrm:Sequence>` SOAP header block is
299 bound.

300 `/wsrm:Sequence/wsrm:Identifier`

301 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely
302 identifies the Sequence.

303 /wsrm:Sequence/wsrm:Identifier/@{any}

304 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
305 to the element.

306 /wsrm:Sequence/wsrm:MessageNumber

307 This REQUIRED element MUST contain an xs:unsignedLong representing the ordinal position of
308 the message within a Sequence. Sequence MessageNumbers start at 1 and monotonically
309 increase throughout the Sequence. If the message number exceeds the internal limitations of an
310 RM Source or RM Destination or reaches the maximum value of an xs:unsignedLong
311 (18,446,744,073,709,551,615), the RM Source or Destination MUST issue a
312 MessageNumberRollover fault.

313 /wsrm:Sequence/wsrm:LastMessage

314 This element MAY be included by the RM Source endpoint. The <wsrm:LastMessage> element
315 has no content.

316 /wsrm:Sequence/{any}

317 This is an extensibility mechanism to allow different types of information, based on a schema, to
318 be passed.

319 /wsrm:Sequence/@{any}

320 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
321 to the element.

322 A RM Source endpoint MUST include a <wsrm:LastMessage> element in the
323 <wsrm:Sequence> element for the last message in a Sequence. An RM Destination
324 endpoint MUST respond with a <wsrm:SequenceAcknowledgement> upon receipt of a
325 <wsrm:LastMessage> element. A Sequence MUST NOT use a <wsrm:MessageNumber>
326 value greater than that which accompanies a <wsrm:LastMessage> element. An RM
327 Destination MUST generate a LastMessageNumberExceeded (See Section Last
328 Message Number Exceeded) fault upon receipt of such a message. In the event that
329 an RM Source needs to close a Sequence and there is no application message, the
330 RM Source MAY send a message with an empty body containing <wsrm:Sequence>
331 header with the <wsrm:LastMessage> element. In this usage, the action URI MUST
332 be:

333 <http://docs.oasis-open.org/wsrm/200510/LastMessage>

334 in preference to the pattern defined in Section 1.2.

335 The following example illustrates a Sequence header block.

```
336 <wsrm:Sequence>
337   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
338   <wsrm:MessageNumber>10</wsrm:MessageNumber>
339   <wsrm:LastMessage/>
340 </wsrm:Sequence>
```

341 3.2 Sequence Acknowledgement

342 The RM Destination informs the RM Source of successful message receipt using a
343 <wsrm:SequenceAcknowledgement> header block. The
344 <wsrm:SequenceAcknowledgement> header block MAY be transmitted independently
345 or included on return messages. The RM Destination MAY send a
346 <wsrm:SequenceAcknowledgement> header block at any point during which the
347 sequence is valid. The timing of acknowledgements can be advertised using policy
348 and acknowledgements can be explicitly requested using the <wsrm:AckRequested>
349 directive (see Section RequestAcknowledgement). If a non-mustUnderstand fault
350 occurs when processing an RM Header that was piggy-backed on another message,
351 a fault MUST be generated, but the processing of the original message MUST NOT be
352 affected.

353 The following exemplar defines its syntax:

```
354 <wsrm:SequenceAcknowledgement ...>
355   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
356   [ [ <wsrm:AcknowledgementRange ...
357     Upper="xs:unsignedLong"
358     Lower="xs:unsignedLong"/> +
359     <wsrm:Final/> ? ]
360   | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> +
361   | <wsrm:None/> ]
362   ...
363 </wsrm:SequenceAcknowledgement>
```

364 The following describes the content model of the <wsrm:SequenceAcknowledgement>
365 header block.

366 /wsrm:SequenceAcknowledgement

367 This element contains the Sequence acknowledgement information.

368 /wsrm:SequenceAcknowledgement/wsrm:Identifier

369 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely
370 identifies the Sequence.

File name

Date

371 /wsrm:SequenceAcknowledgement/wsrm:Identifier/@{any}
372 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
373 to the element.

374 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange
375 This OPTIONAL element, if present, can occur 1 or more times. It contains a range of message
376 Sequence MessageNumbers successfully received by the receiving endpoint manager. The
377 ranges SHOULD NOT overlap. This element MUST NOT be present if either the <wsrm:Nack>
378 or <wsrm:None> elements are also present as a child of
379 <wsrm:SequenceAcknowledgement>.

380 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper
381 This REQUIRED attribute contains an xs:unsignedLong representing the
382 <wsrm:MessageNumber> of the highest contiguous message in a Sequence range received by
383 the RM Destination.

384 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower
385 This REQUIRED attribute contains an xs:unsignedLong representing the
386 <wsrm:MessageNumber> of the lowest contiguous message in a Sequence range received by
387 the RM Destination.

388 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@{any}
389 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
390 to the element.

391 /wsrm:SequenceAcknowledgement/wsrm:Final
392 This OPTIONAL element, if present, indicates that the RM Destination is not receiving new
393 messages for the specified Sequence. The RM Source can be assured that the ranges of
394 messages acknowledged by this SequenceAcknowledgement header block will not change in the
395 future. This element MUST be present when the Sequence is no longer receiving new message
396 for the specified sequence. Note: this element MUST NOT be used when sending a Nack, it can
397 only be used when sending AcknowledgementRanges.

398 /wsrm:SequenceAcknowledgement/wsrm:Nack
399 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the
400 <wsrm:MessageNumber> of an unreceived message in a Sequence. This element permits the
401 gap analysis of the <wsrm:AcknowledgementRange> elements to be performed at the RM
402 Destination rather than at the RM Source which may yield performance benefits in certain
403 environments. The <wsrm:Nack> element MUST NOT be present if either the
404 <wsrm:AcknowledgementRange> or <wsrm:None> elements are also present as a child of
405 <wsrm:SequenceAcknowledgement>. Upon the receipt of a Nack, an RM Source SHOULD

406 retransmit the message identified by the Nack. The RM Destination MUST NOT issue a
407 <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a message that it has
408 previously acknowledged within a <wsrm:AcknowledgementRange>. The RM Source SHOULD
409 ignore a <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a message
410 that has previously been acknowledged within a <wsrm:AcknowledgementRange>.

411 /wsrm:SequenceAcknowledgement/wsrm:None

412 This OPTIONAL element, if present, MUST be used when the RM Destination has not received
413 any messages for the specified sequence. The <wsrm:None> element MUST NOT be present if
414 either the <wsrm:AcknowledgementRange> or <wsrm:Nack> elements are also present as a
415 child of the <wsrm:SequenceAcknowledgement>.

416 /wsrm:SequenceAcknowledgement/{any}

417 This is an extensibility mechanism to allow different (extensible) types of information, based on a
418 schema, to be passed.

419 /wsrm:SequenceAcknowledgement/@{any}

420 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
421 to the element.

422 The following examples illustrate <wsrm:SequenceAcknowledgement> elements:

- 423 • Message numbers 1...10 inclusive in a Sequence have been received by the RM Destination.

```
424 <wsrm:SequenceAcknowledgement>  
425     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
426     <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
427 </wsrm:SequenceAcknowledgement>
```

- 428 • Message numbers 1..2, 4..6, and 8..10 inclusive in a Sequence have been received by the
429 RM Destination, messages 3 and 7 have not been received.

```
430 <wsrm:SequenceAcknowledgement>  
431     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
432     <wsrm:AcknowledgementRange Upper="2" Lower="1"/>  
433     <wsrm:AcknowledgementRange Upper="6" Lower="4"/>  
434     <wsrm:AcknowledgementRange Upper="10" Lower="8"/>  
435 </wsrm:SequenceAcknowledgement>
```

- 436 • Message number 3 in a Sequence has not been received by the RM Destination.

```
437 <wsrm:SequenceAcknowledgement>  
438     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
439     <wsrm:Nack>3</wsrm:Nack>
```

440 `</wsrm:SequenceAcknowledgement>`

441 **3.3 Request Acknowledgement**

442 The purpose of the `<wsrm:AckRequested>` header block is to signal to the RM
443 Destination that the RM Source is requesting that a
444 `<wsrm:SequenceAcknowledgement>` be returned.

445 At any time, the RM Source may request an acknowledgement message from the RM
446 Destination endpoint using an `<wsrm:AckRequested>` header block.

447 The RM Source endpoint requests this acknowledgement by including an
448 `<wsrm:AckRequested>` header block in the message. An RM Destination that receives
449 a message that contains an `<wsrm:AckRequested>` header block MUST respond with
450 a message containing a `<wsrm:SequenceAcknowledgement>` header block. If a non-
451 mustUnderstand fault occurs when processing an RM Header that was piggy-backed
452 on another message, a fault MUST be generated, but the processing of the original
453 message MUST NOT be affected.

454 The following exemplar defines its syntax:

```
455 <wsrm:AckRequested ...>  
456   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
457   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber> ?  
458   ...  
459 </wsrm:AckRequested>
```

460 `/wsrm:AckRequested`

461 This element requests an acknowledgement for the identified sequence.

462 `/wsrm:AckRequested/wsrm:Identifier`

463 This REQUIRED element MUST contain an absolute URI, conformant with RFC2396, that
464 uniquely identifies the Sequence to which the request applies.

465 `/wsrm:AckRequested/wsrm:Identifier/@{any}`

466 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
467 to the element.

468 `/wsrm:AckRequested/wsrm:MessageNumber`

469 This OPTIONAL element, if present, MUST contain an `xs:unsignedLong` representing the highest
470 `<wsrm:MessageNumber>` sent by the RM Source within the Sequence. If present, it MAY be
471 treated as a hint to the RM Destination as an optimization to the process of preparing to transmit a
472 `<wsrm:SequenceAcknowledgement>`.

473 /wsmr:AckRequested/{any}

474 This is an extensibility mechanism to allow different (extensible) types of information, based on a
475 schema, to be passed.

476 /wsmr:AckRequested/@{any}

477 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
478 to the element.

479 **3.4 Sequence Creation**

480 The RM Source MUST request creation of an outbound Sequence by sending a
481 <wsmr:CreateSequence> element in the body of a message to the RM Destination
482 which in turn responds either with a <wsmr:CreateSequenceResponse> or a
483 CreateSequenceRefused fault in the body of the response message.
484 <wsmr:CreateSequence> MAY carry an offer to create an inbound sequence which is
485 either accepted or rejected in the <wsmr:CreateSequenceResponse>.

486 The RM Destination of the outbound sequence is the WS-Addressing
487 EndpointReference [WS-Addressing] to which <wsmr:CreateSequence> is sent. The
488 RM Destination of the inbound sequence is the WS-Addressing <wsa:ReplyTo> of the
489 <wsmr:CreateSequence>.

490 The following exemplar defines the <wsmr:CreateSequence> syntax:

```
491 <wsmr:CreateSequence ...>  
492   <wsmr:AcksTo ...> wsa:EndpointReferenceType </wsmr:AcksTo>  
493   <wsmr:Expires ...> xs:duration </wsmr:Expires> ?  
494   <wsmr:Offer ...>  
495     <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
496     <wsmr:Expires ...> xs:duration </wsmr:Expires> ?  
497     ...  
498   </wsmr:Offer> ?  
499   ...  
500 </wsmr:CreateSequence>
```

501 /wsmr:CreateSequence

502 This element requests creation of a new Sequence between the RM Source that sends it, and the
503 RM Destination to which it is sent. This element MUST NOT be sent as a header block. The RM
504 Destination MUST respond either with a <wsmr:CreateSequenceResponse> response
505 message or a CreateSequenceRefused fault.

506 /wsmr:CreateSequence/wsmr:AcksTo

507 This REQUIRED element, of type `wsa:EndpointReferenceType` as specified by WS-Addressing
508 [WS-Addressing] specifies the endpoint reference to which
509 `<wsrm:SequenceAcknowledgement>` messages and faults related to the created Sequence
510 are to be sent.

511 `/wsrm:CreateSequence/wsrm:Expires`

512 This element, if present, of type `xs:duration` specifies the RM Source's requested duration for
513 the Sequence. The RM Destination MAY either accept the requested duration or assign a lesser
514 value of its choosing. A value of 'PT0S' indicates that the Sequence will never expire. Absence of
515 the element indicates an implied value of 'PT0S'.

516 `/wsrm:CreateSequence/wsrm:Expires/@{any}`

517 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
518 to the element.

519 `/wsrm:CreateSequence/wsrm:Offer`

520 This element, if present, enables an RM Source to offer a corresponding Sequence for the reliable
521 exchange of messages transmitted from RM Destination to RM Source.

522 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier`

523 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely
524 identifies the offered Sequence.

525 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier/@{any}`

526 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
527 to the element.

528 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Expires`

529 This element, if present, of type `xs:duration` specifies the duration for the Sequence. A value
530 of 'PT0S' indicates that the Sequence will never expire. Absence of the element indicates an
531 implied value of 'PT0S'.

532 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Expires/@{any}`

533 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
534 to the element.

535 `/wsrm:CreateSequence/wsrm:Offer/{any}`

536 This is an extensibility mechanism to allow different (extensible) types of information, based on a
537 schema, to be passed.

538 `/wsrm:CreateSequence/wsrm:Offer/@{any}`

539 This is an extensibility mechanism to allow different (extensible) types of information, based on a
540 schema, to be passed.

541 OPTIONAL/wsrn:CreateSequence/{any}

542 This is an extensibility mechanism to allow different (extensible) types of information, based on a
543 schema, to be passed.

544 /wsrm:CreateSequence/@{any}

545 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
546 to the element.

547 A <wsrm:CreateSequenceResponse> is sent in the body of a response message by an
548 RM Destination in response to receipt of a <wsrm:CreateSequence> request
549 message. It carries the <wsrm:Identifier> of the created Sequence and indicates
550 that the RM Source may begin sending messages in the context of the identified
551 Sequence.

552 The following exemplar defines the <wsrm:CreateSequenceResponse> syntax:

```
553 <wsrm:CreateSequenceResponse ...>  
554   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
555   <wsrm:Expires> xs:duration </wsrm:Expires> ?  
556   <wsrm:Accept ...>  
557     <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>  
558     ...  
559   </wsrm:Accept> ?  
560   ...  
561 </wsrm:CreateSequenceResponse>
```

562 /wsrm:CreateSequenceResponse

563 This element is sent in the body of the response message in response to a
564 <wsrm:CreateSequence> request message. It indicates that the RM Destination has created
565 a new Sequence at the request of the RM Source. This element MUST NOT be sent as a header
566 block.

567 /wsrm:CreateSequenceResponse/wsrn:Identifier

568 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 of the
569 Sequence that has been created by the RM Destination.

570 /wsrm:CreateSequenceResponse/wsrn:Identifier/@{any}

571 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
572 to the element.

573 /wsmr:CreateSequenceResponse/wsmr:Expires
574 This element, if present, of type `xs:duration` accepts or refines the RM Source's requested
575 duration for the Sequence. A value of 'PT0S' indicates that the Sequence will never expire.
576 Absence of the element indicates an implied value of 'PT0S'. This value MUST be equal or lesser
577 than the value requested by the RM Source in the corresponding `<wsmr:CreateSequence>`
578 message.

579 /wsmr:CreateSequenceResponse/wsmr:Expires/@{any}
580 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
581 to the element.

582 /wsmr:CreateSequenceResponse/wsmr:Accept
583 This element, if present, enables an RM Destination to accept the offer of a corresponding
584 Sequence for the reliable exchange of messages transmitted from RM Destination to RM Source.
585 This element MUST be present if the corresponding `<wsmr:CreateSequence>` message
586 contained an `<wsmr:Offer>` element.

587 /wsmr:CreateSequenceResponse/wsmr:Accept/wsmr:AcksTo
588 This REQUIRED element, of type `wsa:EndpointReferenceType` as specified by WS-Addressing
589 [WS-Addressing], specifies the endpoint reference to which
590 `<wsmr:SequenceAcknowledgement>` messages related to the accepted Sequence are to be
591 sent.

592 /wsmr:CreateSequenceResponse/wsmr:Accept/{any}
593 This is an extensibility mechanism to allow different (extensible) types of information, based on a
594 schema, to be passed.

595 /wsmr:CreateSequenceResponse/wsmr:Accept/@{any}
596 This is an extensibility mechanism to allow different (extensible) types of information, based on a
597 schema, to be passed.

598 /wsmr:CreateSequenceResponse/{any}
599 This is an extensibility mechanism to allow different (extensible) types of information, based on a
600 schema, to be passed.

601 /wsmr:CreateSequenceResponse/@{any}
602 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
603 to the element.

604 3.5 Sequence Termination

605 When the RM Source has completed its use of the Sequence, it sends a
606 `<wsrm:TerminateSequence>` element, in the body of a message to the RM
607 Destination to indicate that the Sequence is complete, and that it will not be sending
608 any further messages related to the Sequence. The RM Destination can safely reclaim
609 any resources associated with the Sequence upon receipt of the
610 `<wsrm:TerminateSequence>` message. Note, under normal usage the RM source will
611 complete its use of the sequence when all of the messages in the Sequence have
612 been acknowledged. However, the RM Source is free to Terminate or Close a
613 Sequence at any time regardless of the acknowledgement state of the messages.

614 The following exemplar defines the TerminateSequence syntax:

```
615 <wsrm:TerminateSequence ...>  
616   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
617   ...  
618 </wsrm:TerminateSequence>
```

619 `/wsrm:TerminateSequence`

620 This element is sent by an RM Source to indicate it has completed its use of the Sequence, i.e. it
621 MUST NOT send any additional message to the RM Destination referencing this sequence. It
622 indicates that the RM Destination can safely reclaim any resources related to the identified
623 Sequence. This element MUST NOT be sent as a header block.

624 `/wsrm:TerminateSequence/wsrm:Identifier`

625 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 of the
626 Sequence that is being terminated.

627 `/wsrm:TerminateSequence/wsrm:Identifier/@{any}`

628 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
629 to the element.

630 `/wsrm:TerminateSequence/{any}`

631 This is an extensibility mechanism to allow different (extensible) types of information, based on a
632 schema, to be passed.

633 `/wsrm:TerminateSequence/@{any}`

634 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
635 to the element.

636 3.6 Closing A Sequence

637 There may be times during the use of an RM Sequence that the RM Source or RM
638 Destination will wish to discontinue using a Sequence even if some of the messages
639 have not been successfully delivered to the RM Destination.

640 In the case where the RM Source wishes to discontinue use of a sequence, while it
641 can send a TerminateSequence to the RM Destination, since this is a one-way
642 message and due to the possibility of late arriving (or lost) messages and
643 Acknowledgements, this would leave the RM Source unsure of the final ranges of
644 messages that were successfully delivered to the RM Destination.

645 To alleviate this, the RM Source can send a <wsrm:CloseSequence> element, in the
646 body of a message, to the RM Destination to indicate that RM Destination MUST NOT
647 receive any new messages for the specified sequence, other than those already
648 received at the time the <wsrm:CloseSequence> element is interpreted by the RMD.
649 Upon receipt of this message the RM Destination MUST send a
650 SequenceAcknowledgement to the RM Source. Note, this
651 SequenceAcknowledgement MUST include the <wsrm:Final> element.

652 While the RM Destination MUST NOT receive any new messages for the specified
653 sequence it MUST still process RM protocol messages. For example, it MUST respond
654 to AckRequested, TerminateSequence as well as CloseSequence messages. Note,
655 subsequent CloseSequence messages have no effect on the state of the sequence.

656 In the case where the RM Destination wishes to discontinue use of a sequence it may
657 'close' the sequence itself. Please see wsrm:Final above and the SequenceClosed
658 fault below. Note, the SequenceClosed Fault SHOULD be used in place of the
659 SequenceTerminated Fault, whenever possible, to allow the RM Source to still receive
660 Acknowledgements.

661 The following exemplar defines the CloseSequence syntax:

```
662 <wsrm:CloseSequence wsrm:Identifier="xs:anyURI"/>
```

663 /wsrm:CloseSequence

664 This element is sent by an RM Source to indicate that the RM Destination MUST NOT receive any
665 new messages for this sequence. A SequenceClosed fault MUST be generated by the RM
666 Destination when it receives a message for a sequence that is closed.

667 /wsrm:CloseSequence@Identifier

668 This REQUIRED attribute contains an absolute URI conformant with RFC2396 that uniquely
669 identifies the sequence.

670 /wsrm:CloseSequence/{any}

671 This is an extensibility mechanism to allow different (extensible) types of information, based on a
672 schema, to be passed.

673 `/wsrm:CloseSequence@{any}`

674 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
675 to the element.

676 A `<wsrm:CloseSequenceResponse>` is sent in the body of a response message by an
677 RM Destination in response to receipt of a `<wsrm:CloseSequence>` request message.
678 It indicates that the RM Destination has closed the sequence.

679 The following exemplar defines the `<wsrm:CloseSequenceResponse>` syntax:

680 `/wsrm:CloseSequenceResponse`

681 `/wsrm:CloseSequenceResponse`

682 This element is sent in the body of a response message by an RM Destination in response to
683 receipt of a `<wsrm:CloseSequence>` request message. It indicates that the RM Destination has
684 closed the sequence.

685 `/wsrm:CloseSequenceResponse/{any}`

686 This is an extensibility mechanism to allow different (extensible) types of information, based on a
687 schema, to be passed.

688 `/wsrm:CloseSequenceResponse@{any}`

689 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
690 to the element.

691 4 Faults

692 The fault definitions defined in this section reference certain abstract properties, such
693 as [fault endpoint], that are defined in section 3 of the WS-Addressing [WS-
694 Addressing] specification. Endpoints compliant with this specification MUST include
695 required Message Addressing Properties on all fault messages.

696 Sequence creation uses a CreateSequence, CreateSequenceResponse request-
697 response pattern. Faults for this operation are treated as defined in WS-Addressing.
698 CreateSequenceRefused is a possible fault reply for this operation.

699 UnknownSequence is a fault generated by endpoints when messages carrying RM
700 header blocks targeted at unrecognized sequences are detected, these faults are also
701 treated as defined in WS-Addressing. All other faults in this section relate to the
702 processing of RM header blocks targeted at known sequences and are collectively
703 referred to as sequence faults. Sequence faults SHOULD be sent to the same
704 [destination] as <wsrm:SequenceAcknowledgement> messages. These faults are
705 correlated using the Sequence identifier carried in the detail.

706 WS-ReliableMessaging faults MUST include as the [action] property the default fault
707 action URI defined in the version of WS-Addressing used in the message. The value
708 from the current version is below for informational purposes:

709 `http://schemas.xmlsoap.org/ws/2004/08/addressing/fault`

710 The faults defined in this section are generated if the condition stated in the
711 preamble is met. Fault handling rules are defined in section 4 of WS-Addressing.

712 The definitions of faults use the following properties:

713 [Code] The fault code.

714 [Subcode] The fault subcode.

715 [Reason] The English language reason element.

716 [Detail] The detail element. If absent, no detail element is defined for the fault.

717 The [Code] property MUST be either "Sender" or "Receiver". These properties are
718 serialized into text XML as follows:

SOAP Version	Sender	Receiver
SOAP 1.1	S11:Client	S11:Server
SOAP 1.2	S:Sender	S:Receiver

719 The properties above bind to a SOAP 1.2 fault as follows:

```
720 <S:Envelope>
721 <S:Header>
722 <wsa:Action>
723 http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
724 </wsa:Action>
725 <!-- Headers elided for clarity. -->
726 </S:Header>
727 <S:Body>
728 <S:Fault>
729 <S:Code>
730 <S:Value> [Code] </S:Value>
731 <S:Subcode>
732 <S:Value> [Subcode] </S:Value>
733 </S:Subcode>
734 </S:Code>
735 <S:Reason>
736 <S:Text xml:lang="en"> [Reason] </S:Text>
737 </S:Reason>
738 <S:Detail>
739 [Detail]
740 ...
741 </S:Detail>
742 </S:Fault>
743 </S:Body>
744 </S:Envelope>
```

745 The properties above bind to a SOAP 1.1 fault as follows when the fault is triggered
746 by processing an RM header block:

```
747 <S11:Envelope>
748 <S11:Header>
749 <wsrm:SequenceFault>
750 <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
751 ...
752 </wsrm:SequenceFault>
753 <!-- Headers elided for clarity. -->
754 </S11:Header>
755 <S11:Body>
756 <S11:Fault>
757 <faultcode> [Code] </faultcode>
```

```
758     <faultstring> [Reason] </faultstring>
759   </S11:Fault>
760 </S11:Body>
761 </S11:Envelope>
```

762 The properties bind to a SOAP 1.1 fault as follows when the fault is generated as a
763 result of processing a <wsrm:CreateSequence> request message:

```
764 <S11:Envelope>
765   <S11:Body>
766     <S11:Fault>
767       <faultcode> [Subcode] </faultcode>
768       <faultstring xml:lang="en"> [Reason] </faultstring>
769     </S11:Fault>
770   </S11:Body>
771 </S11:Envelope>
```

772 4.1 SequenceFault Element

773 The purpose of the <wsrm:SequenceFault> element is to carry the specific details of
774 a fault generated during the reliable messaging specific processing of a message
775 belonging to a Sequence. The <wsrm:SequenceFault> container MUST only be used
776 in conjunction with the SOAP1.1 fault mechanism. It MUST NOT be used in
777 conjunction with the SOAP1.2 binding.

778 The following exemplar defines its syntax:

```
779 <wsrm:SequenceFault ...>
780   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
781   ...
782 </wsrm:SequenceFault>
```

783 The following describes the content model of the `SequenceFault` element.

784 /wsrm:SequenceFault

785 This is the element containing Sequence information for WS-ReliableMessaging

786 /wsrm:SequenceFault/wsrm:FaultCode

787 This element, if present, MUST contain a qualified name from the set of fault codes defined
788 below.

789 /wsrm:SequenceFault/{any}

790 This is an extensibility mechanism to allow different (extensible) types of information, based on a
791 schema, to be passed.

792 /wsrm:SequenceFault/@{any}

793 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
794 to the element.

795 **4.2 Sequence Terminated**

796 This fault is sent by either the RM Source or the RM Destination to indicate that the
797 endpoint that generated the fault has either encountered an unrecoverable condition,
798 or has detected a violation of the protocol and as a consequence, has chosen to
799 terminate the sequence. The endpoint that generates this fault should make every
800 reasonable effort to notify the corresponding endpoint of this decision.

801 Properties:

802 [Code] Sender or Receiver

803 [Subcode] wsrm:SequenceTerminated

804 [Reason] The Sequence has been terminated due to an unrecoverable error.

805 [Detail]

806 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

807 **4.3 Unknown Sequence**

808 This fault is sent by either the RM Source or the RM Destination in response to a
809 message containing an unknown sequence identifier.

810 Properties:

811 [Code] Sender

812 [Subcode] wsrm:UnknownSequence

813 [Reason] The value of wsrm:Identifier is not a known Sequence identifier.

814 [Detail]

815 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

816 **4.4 Invalid Acknowledgement**

817 This fault is sent by the RM Source in response to a
818 `<wsrm:SequenceAcknowledgement>` that violates the cumulative acknowledgement
819 invariant. An example of such a violation would be a `SequenceAcknowledgement`
820 covering messages that have not been sent.

821 [Code] Sender

822 [Subcode] `wsrm:InvalidAcknowledgement`

823 [Reason] The `SequenceAcknowledgement` violates the cumulative acknowledgement
824 invariant.

825 [Detail]

826 `<wsrm:SequenceAcknowledgement ...> ... </wsrm:SequenceAcknowledgement>`

827 **4.5 Message Number Rollover**

828 This fault is sent to indicate that message numbers for a sequence have been
829 exhausted.

830 Properties:

831 [Code] Sender

832 [Subcode] `wsrm:MessageNumberRollover`

833 [Reason] The maximum value for `wsrm:MessageNumber` has been exceeded.

834 [Detail]

835 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

836 **4.6 Last Message Number Exceeded**

837 This fault is sent by an RM Destination to indicate that it has received a message that
838 has a `<wsrm:MessageNumber>` within a `Sequence` that exceeds the value of the
839 `<wsrm:MessageNumber>` element that accompanied a `<wsrm:LastMessage>` element
840 for the `Sequence`.

841 Properties:

842 [Code] Sender

843 [Subcode] `wsrm:LastMessageNumberExceeded`

File name

Date

844 [Reason] The value for wsrn:MessageNumber exceeds the value of the
845 MessageNumber accompanying a LastMessage element in this Sequence.

846 [Detail]

847 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

848 **4.7 Create Sequence Refused**

849 This fault is sent in response to a create sequence request that cannot be satisfied.

850 Properties:

851 [Code] Sender

852 [Subcode] wsrn:CreateSequenceRefused

853 [Reason] The create sequence request has been refused by the RM Destination.

854 [Detail] empty

855 **4.8 Sequence Closed**

856 This fault is sent by an RM Destination to indicate that the specified sequence has
857 been closed. This fault MUST be generated when an RM Destination is asked to
858 receive a message for a sequence that is closed.

859 Properties:

860 [Code] Sender

861 [Subcode] wsrn:SequenceClosed

862 [Reason] The sequence is closed and can not receive new messages.

863 [Detail] `<wsrm:Identifier...> xs:anyURI </wsrm:Identifier>`

864 **5 Security Considerations**

865 It is strongly recommended that the communication between services be secured
866 using the mechanisms described in WS-Security. In order to properly secure
867 messages, the body and all relevant headers need to be included in the signature.
868 Specifically, the `<wsrm:Sequence>` header needs to be signed with the body in order
869 to "bind" the two together. The `<wsrm:SequenceAcknowledgement>` header may be
870 signed independently because a reply independent of the message is not a security
871 concern.

872 Because Sequences are expected to exchange a number of messages, it is
873 recommended that a security context be established using the mechanisms described
874 in WS-Trust and WS-SecureConversation [SecureConversation]. If a Sequence is
875 bound to a specific endpoint, then the security context needs to be established or
876 shared with the endpoint servicing the Sequence. While the context can be
877 established at any time, it is critical that the messages establishing the Sequence be
878 secured even if they precede security context establishment. However, it is
879 recommended that the security context be established first. Security contexts are
880 independent of reliable messaging Sequences. Consequently, security contexts can
881 come and go independent of the lifetime of the Sequence. In fact, it is
882 recommended that the lifetime of a security context be less than the lifetime of the
883 Sequence unless the Sequence is very short-lived.

884 It is common for message Sequences to exchange a number of messages (or a large
885 amount of data). As a result, the usage profile of a Sequence is such that it is
886 susceptible to key attacks. For this reason it is strongly recommended that the keys
887 be changed frequently. This "re-keying" can be effected a number of ways. The
888 following list outlines four common techniques:

- 889 • Closing and re-establishing a security context
- 890 • Exchanging new secrets between the parties
- 891 • Using a derived key sequence and switch "generations"
- 892 • Attaching a nonce to each message and using it in a derived key function with the shared
893 secret

894 The security context may be re-established using the mechanisms described in WS-
895 Trust and WS-SecureConversation. Similarly, secrets can be exchanged using the
896 mechanisms described in WS-Trust. Note, however, that the current shared secret
897 should not be used to encrypt the new shared secret. Derived keys, the preferred
898 solution from this list, can be specified using the mechanisms described in WS-
899 SecureConversation.

900 There is a core tension between security and reliable messaging that can be
901 problematic if not considered in implementations. That is, one aspect of security is
902 to prevent message replay and the core tenet of reliable messaging is to replay
903 messages until they are acknowledged. Consequently, if the security sub-system
904 processes a message but a failure occurs before the reliable messaging sub-system
905 records the message (or the message is considered "processed"), then it is possible
906 (and likely) that the security sub-system will treat subsequent copies as replays and
907 discard them. At the same time, the reliable messaging sub-system will likely
908 continue to expect and even solicit the missing message(s). Care should be taken to
909 avoid and prevent this rare condition.

910 The following list summarizes common classes of attacks that apply to this protocol
911 and identifies the mechanism to prevent/mitigate the attacks:

- 912 • **Message alteration** – Alteration is prevented by including signatures of the message
913 information using WS-Security.
- 914 • **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-
915 Security.
- 916 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by
917 comparing secured policies – see WS-Policy and WS-SecurityPolicy).
- 918 • **Authentication** – Authentication is established using the mechanisms described in WS-
919 Security and WS-Trust. Each message is authenticated using the mechanisms described in
920 WS-Security.
- 921 • **Accountability** – Accountability is a function of the type of and string of the key and
922 algorithms being used. In many cases, a strong symmetric key provides sufficient
923 accountability. However, in some environments, strong PKI signatures are required.
- 924 • **Availability** – All reliable messaging services are subject to a variety of availability attacks.
925 Replay detection is a common attack and it is recommended that this be addressed by the
926 mechanisms described in WS-Security. (Note that because of legitimate message replays,
927 detection should include a differentiator besides message id such as a timestamp). Other
928 attacks, such as network-level denial of service attacks are harder to avoid and are outside
929 the scope of this specification. That said, care should be taken to ensure that minimal state is
930 saved prior to any authenticating sequences.

931 **6 References**

932 **6.1 Normative**

933 **[KEYWORDS]**

934 S. Bradner, "[Key words for use in RFCs to Indicate Requirement Levels](#)," RFC 2119, Harvard
935 University, March 1997

936 **[SOAP]**

937 W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.

938 **[URI]**

939 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#),"
940 RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.

941 **[XML-ns]**

942 W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.

943 **[XML-Schema1]**

944 W3C Recommendation, "[XML Schema Part 1: Structures](#)," 2 May 2001.

945 **[XML-Schema2]**

946 W3C Recommendation, "[XML Schema Part 2: Datatypes](#)," 2 May 2001.

947 **[WSSecurity]**

948 "[OASIS Web Services Security: SOAP Message Security 1.0 \(WS-Security 2004\)](#)",
949 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds, OASIS
950 Standard 200401, March 2004.

951 **[Tanenbaum]**

952 "Computer Networks," Andrew S. Tanenbaum, Prentice Hall PTR, 2003.

953 **[WSDL]**

954 W3C Note, "[Web Services Description Language \(WSDL 1.1\)](#)," 15 March 2001.

955 **[WS-Addressing]**

956 D. Box, et al, "[Web Services Addressing \(WS-Addressing\)](#)," August 2004.

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

957 **6.2 Non-Normative**

958 **[WS-Policy]**

959 D. Box, et al, "[Web Services Policy Framework \(WS-Policy\)](#)," September 2004.

960 **[WS-PolicyAttachment]**

961 D. Box, et al, "[Web Services Policy Attachment \(WS-PolicyAttachment\)](#)," September 2004.

962 **[SecurityPolicy]**

963 G. Della-Libra, "[Web Services Security Policy Language \(WS-SecurityPolicy\)](#)," December 2002.

964 **[SecureConversation]**

965 S. Anderson, et al, "[Web Services Secure Conversation Language \(WS-SecureConversation\)](#),"
966 May 2004.

967

968 Appendix A.Schema

969 The normative schema for WS-ReliableMessaging is located at:

970 `http://docs.oasis-open.org/wsrn/200510/wsrn.xsd`

971 The following copy is provided for reference.

```
972 <xs:schema targetNamespace="http://docs.oasis-open.org/wsrn/200510/"
973 xmlns:xs="http://www.w3.org/2001/XMLSchema"
974 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
975 xmlns:wsrm="http://docs.oasis-open.org/wsrn/200510/"
976 elementFormDefault="qualified" attributeFormDefault="unqualified">
977   <xs:import
978 namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
979 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
980   <!-- Protocol Elements -->
981   <xs:complexType name="SequenceType">
982     <xs:sequence>
983       <xs:element ref="wsrm:Identifier"/>
984       <xs:element name="MessageNumber" type="xs:unsignedLong"/>
985       <xs:element name="LastMessage" minOccurs="0">
986         <xs:complexType>
987           <xs:sequence/>
988         </xs:complexType>
989       </xs:element>
990       <xs:any namespace="##other" processContents="lax" minOccurs="0"
991 maxOccurs="unbounded"/>
992     </xs:sequence>
993     <xs:anyAttribute namespace="##other" processContents="lax"/>
994   </xs:complexType>
995   <xs:element name="Sequence" type="wsrm:SequenceType"/>
996   <xs:element name="SequenceAcknowledgement">
997     <xs:complexType>
998       <xs:sequence>
999         <xs:element ref="wsrm:Identifier"/>
1000         <xs:choice>
1001           <ws:sequence>
1002             <xs:element name="AcknowledgementRange"
1003 maxOccurs="unbounded">
```

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

```

1004         <xs:complexType>
1005             <xs:sequence/>
1006             <xs:attribute name="Upper" type="xs:unsignedLong"
1007 use="required"/>
1008             <xs:attribute name="Lower" type="xs:unsignedLong"
1009 use="required"/>
1010             <xs:anyAttribute namespace="##other"
1011 processContents="lax"/>
1012         </xs:complexType>
1013     </xs:element>
1014     <ws:element name="Final" minOccurs="0">
1015         <xs:complexType>
1016             <xs:sequence/>
1017         </xs:complexType>
1018     </ws:element>
1019 </ws:sequence>
1020     <xs:element name="Nack" type="xs:unsignedLong"
1021 minOccurs="unbounded"/>
1022     <xs:element name="None" minOccurs="0">
1023         <xs:complexType>
1024             <xs:sequence/>
1025         </xs:complexType>
1026     </xs:element>
1027 </xs:choice>
1028     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1029 minOccurs="unbounded"/>
1030 </xs:sequence>
1031     <xs:anyAttribute namespace="##other" processContents="lax"/>
1032 </xs:complexType>
1033 </xs:element>
1034 <xs:complexType name="AckRequestedType">
1035     <xs:sequence>
1036         <xs:element ref="wsrm:Identifier"/>
1037         <xs:element name="MessageNumber" type="xs:unsignedLong"
1038 minOccurs="0"/>
1039         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1040 minOccurs="unbounded"/>
1041     </xs:sequence>
1042     <xs:anyAttribute namespace="##other" processContents="lax"/>
1043 </xs:complexType>

```

```

1044 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1045 <xs:element name="Identifier">
1046   <xs:complexType>
1047     <xs:annotation>
1048       <xs:documentation>
1049 This type is for elements whose [children] is an anyURI and can have
1050 arbitrary attributes.
1051       </xs:documentation>
1052     </xs:annotation>
1053     <xs:simpleContent>
1054       <xs:extension base="xs:anyURI">
1055         <xs:anyAttribute namespace="##other" processContents="lax"/>
1056       </xs:extension>
1057     </xs:simpleContent>
1058   </xs:complexType>
1059 </xs:element>
1060 <!-- Fault Container and Codes -->
1061 <xs:simpleType name="FaultCodes">
1062   <xs:restriction base="xs:QName">
1063     <xs:enumeration value="wsrm:UnknownSequence"/>
1064     <xs:enumeration value="wsrm:SequenceTerminated"/>
1065     <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1066     <xs:enumeration value="wsrm:MessageNumberRollover"/>
1067     <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1068     <xs:enumeration value="wsrm:LastMessageNumberExceeded"/>
1069   </xs:restriction>
1070 </xs:simpleType>
1071 <xs:complexType name="SequenceFaultType">
1072   <xs:sequence>
1073     <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1074     <xs:any namespace="##any" processContents="lax" minOccurs="0"
1075 maxOccurs="unbounded"/>
1076   </xs:sequence>
1077   <xs:anyAttribute namespace="##any" processContents="lax"/>
1078 </xs:complexType>
1079 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1080 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1081 <xs:element name="CreateSequenceResponse"
1082 type="wsrm:CreateSequenceResponseType"/>
1083 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>

```

```

1084     <xs:element name="CloseSequenceResponse"
1085 type="wsrm:CloseSequenceResponseType"/>
1086     <xs:element name="TerminateSequence"
1087 type="wsrm:TerminateSequenceType"/>
1088     <xs:complexType name="CreateSequenceType">
1089       <xs:sequence>
1090         <xs:element ref="wsrm:AcksTo"/>
1091         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1092         <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1093         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1094 maxOccurs="unbounded">
1095           <xs:annotation>
1096             <xs:documentation>
1097 It is the authors intent that this extensibility be used to transfer a
1098 Security Token Reference as defined in WS-Security.
1099 </xs:documentation>
1100           </xs:annotation>
1101         </xs:any>
1102       </xs:sequence>
1103       <xs:anyAttribute namespace="##other" processContents="lax"/>
1104     </xs:complexType>
1105     <xs:complexType name="CreateSequenceResponseType">
1106       <xs:sequence>
1107         <xs:element ref="wsrm:Identifier"/>
1108         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1109         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1110         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1111 maxOccurs="unbounded"/>
1112       </xs:sequence>
1113       <xs:anyAttribute namespace="##other" processContents="lax"/>
1114     </xs:complexType>
1115     <xs:complexType name="CloseSequenceType">
1116       <xs:sequence>
1117         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1118 maxOccurs="unbounded"/>
1119       </xs:sequence>
1120       <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1121       <xs:anyAttribute namespace="##other" processContents="lax"/>
1122     </xs:complexType>
1123     <xs:complexType name="CloseSequenceResponseType">

```

```

1124     <xs:sequence>
1125         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1126 maxOccurs="unbounded"/>
1127     </xs:sequence>
1128     <xs:anyAttribute namespace="##other" processContents="lax"/>
1129 </xs:complexType>
1130 <xs:complexType name="TerminateSequenceType">
1131     <xs:sequence>
1132         <xs:element ref="wsrm:Identifier"/>
1133         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1134 maxOccurs="unbounded"/>
1135     </xs:sequence>
1136     <xs:anyAttribute namespace="##other" processContents="lax"/>
1137 </xs:complexType>
1138 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1139 <xs:complexType name="OfferType">
1140     <xs:sequence>
1141         <xs:element ref="wsrm:Identifier"/>
1142         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1143         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1144 maxOccurs="unbounded"/>
1145     </xs:sequence>
1146     <xs:anyAttribute namespace="##other" processContents="lax"/>
1147 </xs:complexType>
1148 <xs:complexType name="AcceptType">
1149     <xs:sequence>
1150         <xs:element ref="wsrm:AcksTo"/>
1151         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1152 maxOccurs="unbounded"/>
1153     </xs:sequence>
1154     <xs:anyAttribute namespace="##other" processContents="lax"/>
1155 </xs:complexType>
1156 <xs:element name="Expires">
1157     <xs:complexType>
1158         <xs:simpleContent>
1159             <xs:extension base="xs:duration">
1160                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1161             </xs:extension>
1162         </xs:simpleContent>
1163     </xs:complexType>

```


1164
1165

```
</xs:element>  
</xs:schema>
```

1166 **Appendix B.Message Examples**

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

1167 B.1.Create Sequence

1168 Create Sequence

```
1169 <?xml version="1.0" encoding="UTF-8"?>
1170 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1171 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1172 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1173   <S:Header>
1174     <wsa:MessageID>
1175       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1176     </wsa:MessageID>
1177     <wsa:To>http://example.com/serviceB/123</wsa:To>
1178     <wsa:Action>http://docs.oasis-
1179 open.org/wsmr/200510/CreateSequence</wsa:Action>
1180     <wsa:ReplyTo>
1181       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1182     </wsa:ReplyTo>
1183   </S:Header>
1184   <S:Body>
1185     <wsmr:CreateSequence>
1186       <wsmr:AcksTo>
1187         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1188       </wsmr:AcksTo>
1189     </wsmr:CreateSequence>
1190   </S:Body>
1191 </S:Envelope>
```

1192 Create Sequence Response

```
1193 <?xml version="1.0" encoding="UTF-8"?>
1194 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1195 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1196 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1197   <S:Header>
1198     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1199     <wsa:RelatesTo>
1200       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1201     </wsa:RelatesTo>
1202     <wsa:Action>
```

File name

Date

```
1203     http://docs.oasis-open.org/wsrn/200510/CreateSequenceResponse
1204     </wsa:Action>
1205 </S:Header>
1206 <S:Body>
1207     <wsrm:CreateSequenceResponse>
1208         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1209     </wsrm:CreateSequenceResponse>
1210 </S:Body>
1211 </S:Envelope>
```

1212 B.2. Initial Transmission

1213 The following example WS-ReliableMessaging headers illustrate the message
1214 exchange in the above figure. The three messages have the following headers; the
1215 third message is identified as the last message in the sequence:

1216 Message 1

```
1217 <?xml version="1.0" encoding="UTF-8"?>
1218 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1219 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1220 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1221   <S:Header>
1222     <wsa:MessageID>
1223       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1224     </wsa:MessageID>
1225     <wsa:To>http://example.com/serviceB/123</wsa:To>
1226     <wsa:From>
1227       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1228     </wsa:From>
1229     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1230     <wsm:Sequence>
1231       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1232       <wsm:MessageNumber>1</wsm:MessageNumber>
1233     </wsm:Sequence>
1234   </S:Header>
1235   <S:Body>
1236     <!-- Some Application Data -->
1237   </S:Body>
1238 </S:Envelope>
```

1239 Message 2

```
1240 <?xml version="1.0" encoding="UTF-8"?>
1241 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1242 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1243 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1244   <S:Header>
1245     <wsa:MessageID>
1246       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1247     </wsa:MessageID>
```

File name

Date

```

1248 <wsa:To>http://example.com/serviceB/123</wsa:To>
1249 <wsa:From>
1250 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1251 </wsa:From>
1252 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1253 <wsrm:Sequence>
1254 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1255 <wsrm:MessageNumber>2</wsrm:MessageNumber>
1256 </wsrm:Sequence>
1257 </S:Header>
1258 <S:Body>
1259 <!-- Some Application Data -->
1260 </S:Body>
1261 </S:Envelope>

```

1262 Message 3

```

1263 <?xml version="1.0" encoding="UTF-8"?>
1264 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1265 xmlns:wsrm="http://docs.oasis-open.org/wsrm/200510/"
1266 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1267 <S:Header>
1268 <wsa:MessageID>
1269 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1270 </wsa:MessageID>
1271 <wsa:To>http://example.com/serviceB/123</wsa:To>
1272 <wsa:From>
1273 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1274 </wsa:From>
1275 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1276 <wsrm:Sequence>
1277 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1278 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1279 <wsrm:LastMessage/>
1280 </wsrm:Sequence>
1281 </S:Header>
1282 <S:Body>
1283 <!-- Some Application Data -->
1284 </S:Body>
1285 </S:Envelope>

```

1286 B.3.First Acknowledgement

1287 Message number 2 has not been received by the RM Destination due to some
1288 transmission error so it responds with an acknowledgement for messages 1 and 3:

```
1289 <?xml version="1.0" encoding="UTF-8"?>
1290 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1291 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1292 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1293   <S:Header>
1294     <wsa:MessageID>
1295       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1296     </wsa:MessageID>
1297     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1298     <wsa:From>
1299       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1300     </wsa:From>
1301     <wsa:Action>
1302       http://docs.oasis-open.org/wsm/200510/SequenceAcknowledgement
1303     </wsa:Action>
1304     <wsm:SequenceAcknowledgement>
1305       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1306       <wsm:AcknowledgementRange Upper="1" Lower="1"/>
1307       <wsm:AcknowledgementRange Upper="3" Lower="3"/>
1308     </wsm:SequenceAcknowledgement>
1309   </S:Header>
1310   <S:Body/>
1311 </S:Envelope>
```

1312 B.4.Retransmission

1313 The sending endpoint discovers that message number 2 was not received so it
1314 resends the message and requests an acknowledgement:

```
1315 <?xml version="1.0" encoding="UTF-8"?>
1316 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1317 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1318 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1319   <S:Header>
1320     <wsa:MessageID>
1321       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1322     </wsa:MessageID>
1323     <wsa:To>http://example.com/serviceB/123</wsa:To>
1324     <wsa:From>
1325       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1326     </wsa:From>
1327     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1328     <wsmr:Sequence>
1329       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1330       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1331     </wsmr:Sequence>
1332     <wsmr:AckRequested>
1333       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1334     </wsmr:AckRequested>
1335   </S:Header>
1336   <S:Body>
1337     <!-- Some Application Data -->
1338   </S:Body>
1339 </S:Envelope>
```


1340 B.5.Termination

1341 The RM Destination now responds with an acknowledgement for the complete
1342 sequence which can then be terminated:

```
1343 <?xml version="1.0" encoding="UTF-8"?>
1344 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1345 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1346 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1347   <S:Header>
1348     <wsa:MessageID>
1349       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1350     </wsa:MessageID>
1351     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1352     <wsa:From>
1353       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1354     </wsa:From>
1355     <wsa:Action>
1356       http://docs.oasis-open.org/wsm/200510/SequenceAcknowledgement
1357     </wsa:Action>
1358     <wsm:SequenceAcknowledgement>
1359       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1360       <wsm:AcknowledgementRange Upper="3" Lower="1"/>
1361     </wsm:SequenceAcknowledgement>
1362   </S:Header>
1363   <S:Body/>
1364 </S:Envelope>
```

1365 Terminate Sequence

```
1366 <?xml version="1.0" encoding="UTF-8"?>
1367 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1368 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1369 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1370   <S:Header>
1371     <wsa:MessageID>
1372       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1373     </wsa:MessageID>
1374     <wsa:To>http://example.com/serviceB/123</wsa:To>
1375     <wsa:Action>
1376       http://docs.oasis-open.org/wsm/200510/TerminateSequence
```

File name

Date

```
1377     </wsa:Action>
1378     <wsa:From>
1379         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1380     </wsa:From>
1381 </S:Header>
1382 <S:Body>
1383     <wsrm:TerminateSequence>
1384         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1385     </wsrm:TerminateSequence>
1386 </S:Body>
1387 </S:Envelope>
```

1388 Appendix C.WSDL

1389 The non-normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1390 <http://docs.oasis-open.org/wsrn/200510/wSDL/wsrn.wSDL>

1391 The following non-normative copy is provided for reference.

```
1392 <wSDL:definitions xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
1393 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1394 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1395 xmlns:rm="http://docs.oasis-open.org/wsrn/200510/"
1396 xmlns:tns="http://docs.oasis-open.org/wsrn/200510/wSDL"
1397 targetNamespace="http://docs.oasis-open.org/wsrn/200510/wSDL">
1398 <wSDL:types>
1399   <xs:schema>
1400     <xs:import namespace="http://docs.oasis-open.org/wsrn/200510/"
1401     schemaLocation="http://docs.oasis-open.org/wsrn/200510/wsrn.xsd"/>
1402     <xs:import
1403     namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1404     schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1405   </xs:schema>
1406 </wSDL:types>
1407 <wSDL:message name="CreateSequence">
1408   <wSDL:part name="create" element="rm:CreateSequence"/>
1409 </wSDL:message>
1410 <wSDL:message name="CreateSequenceResponse">
1411   <wSDL:part name="createResponse"
1412   element="rm:CreateSequenceResponse"/>
1413 </wSDL:message>
1414 <wSDL:message name="CloseSequence">
1415   <wSDL:part name="close" element="rm:CloseSequence"/>
1416 </wSDL:message>
1417 <wSDL:message name="CloseSequenceResponse">
1418   <wSDL:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1419 </wSDL:message>
1420 <wSDL:message name="TerminateSequence">
1421   <wSDL:part name="terminate" element="rm:TerminateSequence"/>
1422 </wSDL:message>
1423 <wSDL:portType name="SequenceAbstractPortType">
1424   <wSDL:operation name="CreateSequence">
```

File name

Date

```
1425     <wsdl:input message="tns:CreateSequence"
1426 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CreateSequence"/>
1427     <wsdl:output message="tns:CreateSequenceResponse"
1428 wsa:Action="http://docs.oasis-
1429 open.org/wsrn/200510/CreateSequenceResponse"/>
1430 </wsdl:operation>
1431 <wsdl:operation name="CloseSequence">
1432     <wsdl:input name="tns:CloseSequence"
1433 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CloseSequence"/>
1434     <wsdl:output name="tns:CloseSequenceResponse"
1435 wsa:Action="http://docs.oasis-
1436 open.org/wsrn/200510/CloseSequenceResponse"/>
1437 </wsdl:operation>
1438 <wsdl:operation name="TerminateSequence">
1439     <wsdl:input message="tns:TerminateSequence"
1440 wsa:Action="http://docs.oasis-
1441 open.org/wsrn/200510/CreateSequenceResponse"/>
1442 </wsdl:operation>
1443 </wsdl:portType>
1444 </wsdl:definitions>
```

1445 **Appendix D.Acknowledgments**

1446 This document is based on initial contribution to OASIS WS-RX Technical Committee by the
1447 following authors: Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft,
1448 Doug Davis, IBM, Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM,
1449 Mary Ann Hondo, IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David
1450 Langworthy, Microsoft (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft,
1451 Steve Lucco, Microsoft, Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham,
1452 BEA, David Orchard, BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John
1453 Shewchuk, Microsoft, Tony Storey, IBM

1454 The following individuals have provided invaluable input into the initial contribution:

1455 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,
1456 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,
1457 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,
1458 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin
1459 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,
1460 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger
1461 Wolter, Microsoft

1462 The following individuals were members of the committee during the development of this
1463 specification:

1464 TBD

1465 **Appendix E.Revision History**

Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.
ws-02	2005-07-21	Doug Davis	I011 (PT0S) added
wd-02	2005-08-16	Anish Karmarkar	Trivial editorial changes
ws-03	2005-09-15	Doug Davis	I019 and i028 (CloseSeq) added
wd-05	2005-09-26	Gilbert Pilz	i005 (Source resend of nacks messages when ack already received) added.
wd-05	2005-09-27	Doug Davis	i027 (InOrder delivery assurance spanning multiple sequences) added
wd-05	2005-09-27	Doug Davis	i020 (Semantics of "At most once" Delivery Assurance) added
wd-05	2005-09-27	Doug Davis	i034 (Fault while processing a piggy-backed RM header) added
wd-05	2005-09-27	Doug Davis	i033 (Processing model of NACKs) added
wd-05	2005-09-27	Doug Davis	i031 (AckRequested schema inconsistency) added
wd-05	2005-09-27	Doug Davis	i025 (SeqAck/None) added
wd-05	2005-09-27	Doug Davis	i029 (Remove dependency on WS-Security) added
wd-05	2005-09-27	Doug Davis	i039 (What does 'have a mU attribute' mean) added
wd-05	2005-09-27	Doug Davis	i040 (Change 'optiona'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrn/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrn:FaultCode element)

File name

Date

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

1466 **Appendix F.Notices**

1467 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1468 that might be claimed to pertain to the implementation or use of the technology described in this
1469 document or the extent to which any license under such rights might or might not be available;
1470 neither does it represent that it has made any effort to identify any such rights. Information on
1471 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1472 website. Copies of claims of rights made available for publication and any assurances of licenses
1473 to be made available, or the result of an attempt made to obtain a general license or permission
1474 for the use of such proprietary rights by implementors or users of this specification, can be
1475 obtained from the OASIS Executive Director.

1476 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1477 applications, or other proprietary rights which may cover technology that may be required to
1478 implement this specification. Please address the information to the OASIS Executive Director.

1479 Copyright (C) OASIS Open (2005). All Rights Reserved.

1480 This document and translations of it may be copied and furnished to others, and derivative works
1481 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1482 published and distributed, in whole or in part, without restriction of any kind, provided that the
1483 above copyright notice and this paragraph are included on all such copies and derivative works.
1484 However, this document itself may not be modified in any way, such as by removing the copyright
1485 notice or references to OASIS, except as needed for the purpose of developing OASIS
1486 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1487 Property Rights document must be followed, or as required to translate it into languages other
1488 than English.

1489 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1490 successors or assigns.

1491 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1492 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1493 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE

File name

Date

Copyright © OASIS Open 2005. All Rights Reserved.
numbers of Statistics

Page Page

1494 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1495 PARTICULAR PURPOSE.