



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

3 **Working Draft 05, September 26th 2005**

Document identifier:

4 WS-ReliableMessaging-1.1-draft-03.doc

5 **Location:**

6 TBD

7 **Editors:**

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

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

10 TBD

11 **Abstract:**

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

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

22 **Composable Architecture:**

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

30 **Status:**

31 TBD

32 Table of Contents

33	1 INTRODUCTION.....	6
34	1.1 GOALS AND REQUIREMENTS.....	6
35	1.1.1 Requirements.....	6
36	1.2 NOTATIONAL CONVENTIONS.....	6
37	1.3 NAMESPACE.....	7
38	1.4 COMPLIANCE.....	8
39	2 RELIABLE MESSAGING MODEL.....	9
40	2.1 GLOSSARY.....	10
41	2.2 PROTOCOL PRECONDITIONS.....	11
42	2.3 PROTOCOL INVARIANTS.....	11
43	2.4 EXAMPLE MESSAGE EXCHANGE.....	11
44	3 RM PROTOCOL ELEMENTS.....	14
45	3.1 SEQUENCES.....	14
46	3.2 SEQUENCE ACKNOWLEDGEMENT.....	16
47	3.3 REQUEST ACKNOWLEDGEMENT.....	18
48	3.4 SEQUENCE CREATION.....	19
49	3.5 SEQUENCE TERMINATION.....	23
50	3.6 CLOSING A SEQUENCE.....	24
51	4 FAULTS.....	27
52	4.1 SEQUENCE FAULT ELEMENT.....	29
53	4.2 SEQUENCE TERMINATED.....	30
54	4.3 UNKNOWN SEQUENCE.....	30
55	4.4 INVALID ACKNOWLEDGEMENT.....	30
56	4.5 MESSAGE NUMBER ROLLOVER.....	31
57	4.6 LAST MESSAGE NUMBER EXCEEDED.....	31
58	4.7 CREATE SEQUENCE REFUSED.....	32
59	4.8 SEQUENCE CLOSED.....	32
60	5 SECURITY CONSIDERATIONS.....	33
61	6 REFERENCES.....	35
62	6.1 NORMATIVE.....	35
63	6.2 NON-NORMATIVE.....	36

64	APPENDIX A.SHEMA	37
65	APPENDIX B.MESSAGE EXAMPLES.....	42
66	B.1.CREATE SEQUENCE.....	43
67	B.2. INITIAL TRANSMISSION.....	45
68	B.3.FIRST ACKNOWLEDGEMENT.....	47
69	B.4.RETRANSMISSION.....	48
70	B.5.TERMINATION.....	49
71	APPENDIX C.WSDL.....	51
72	APPENDIX D.ACKNOWLEDGMENTS.....	53
73	APPENDIX E.REVISION HISTORY.....	54
74	APPENDIX F.NOTICES.....	55

75 **1 Introduction**

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

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

87 **1.1 Goals and Requirements**

88 **1.1.1 Requirements**

89 **1.2 Notational Conventions**

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

93 This specification uses the following syntax to define normative outlines for
94 messages:

- 95 • The syntax appears as an XML instance, but values in italics indicate data types instead
96 of values.
- 97 • Characters are appended to elements and attributes to indicate cardinality:
 - 98 ○ "?" (0 or 1)
 - 99 ○ "*" (0 or more)
 - 100 ○ "+" (1 or more)
- 101 • The character "|" is used to indicate a choice between alternatives.
- 102 • The characters "[" and "]" are used to indicate that contained items are to be treated as a
103 group with respect to cardinality or choice.

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

•

110 1.3 Namespace

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

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

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

116 The following namespaces are used in this document:

117 *Table 1*

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

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

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

120 All sections explicitly noted as examples are informational and are not to be
121 considered normative.

122 If an action URI is used, and one is not already defined per the rules of the WS-
123 Addressing specification [[WS-Addressing](#)], then the action URI MUST consist of the
124 reliable messaging namespace URI concatenated with the "/" character and the
125 element name. For example:

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

127 **1.4 Compliance**

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

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

136 **2 Reliable Messaging Model**

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

139

140 The WS-ReliableMessaging specification defines an interoperable protocol that
141 requires a Reliable Messaging (RM) Source and Reliable Messaging (RM) Destination
142 to ensure that each message transmitted by the RM Source is successfully received
143 by an RM Destination, or barring successful receipt, that an RM Source can, except in
144 the most extreme circumstances, accurately determine the disposition of each
145 message transmitted as perceived by the RM Destination, so as to resolve any in-
146 doubt status. Note that this specification makes no restriction on the scope of the
147 RM Source or RM Destination entities. For example, either may span multiple WSDL
148 Ports or endpoints.

149 In addition, The protocol allows the RM Source and RM Destination to provide their
150 respective Application Source and Application Destination a guarantee that a
151 message that is sent by an Application Source will be delivered to the Application
152 Destination.

153 This guarantee is specified as a delivery assurance. It is the responsibility of the RM
154 Source and RM Destination to fulfill the delivery assurances on behalf of their
155 respective Application counterparts, or raise an error. The protocol defined here
156 allows endpoints to meet this guarantee for the delivery assurances defined below.
157 However, the means by which these delivery assurances are manifested by either the
158 RM Source or RM Destination roles is an implementation concern, and is out of scope
159 of this specification.

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

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

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

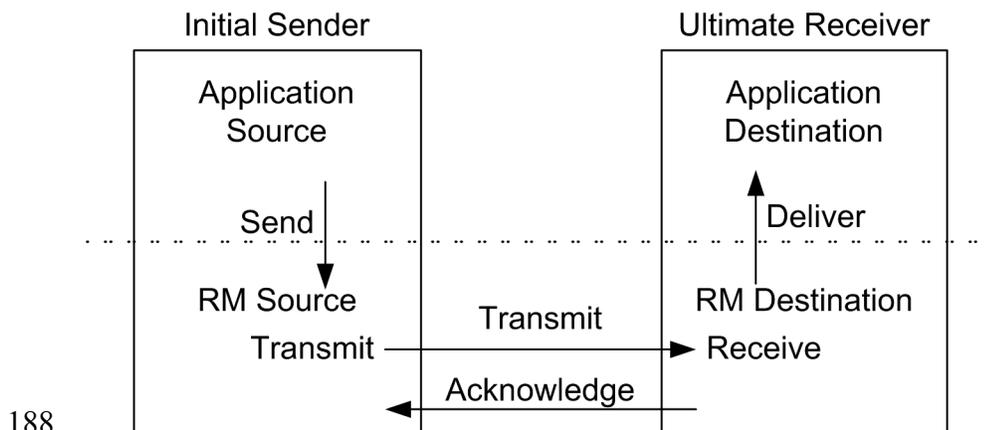
166 **AtMostOnce** Messages will be delivered at most once without duplication or an error
167 will be raised on at least one endpoint. It is possible that some messages in a
168 sequence may not be delivered.

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

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

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

181 Figure 1 below illustrates the entities and events in a simple reliable message
182 exchange. First, the Application Source Sends a message for reliable delivery. The
183 Reliable Messaging (RM) Source accepts the message and Transmits it one or more
184 times. After receiving the message, the RM Destination Acknowledges it. Finally,
185 the RM Destination delivers the message to the Application Destination. The exact
186 roles the entities play and the complete meaning of the events will be defined
187 throughout this specification.



188
189 Figure 1: Reliable Messaging Model

190 2.1 Glossary

191 The following definitions are used throughout this specification:

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

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

195 **Application Destination:** The endpoint to which a message is Delivered.
196 **Delivery Assurance:** The guarantee that the messaging infrastructure provides on
197 the delivery of a message.
198 **Receive:** The act of reading a message from a network connection and qualifying it
199 as relevant to RM Destination functions.
200 **RM Source:** For any one reliably sent message tThe endpoint that transmits the
201 message.
202 **RM Destination:** For any one reliably sent message tThe endpoint that receives the
203 message.
204 **Send:** The act of submitting a message to the RM Source for reliable delivery. The
205 reliability guarantee begins at this point.
206 **Deliver:** The act of transferring a message from the RM Destination to the
207 Application Destination. The reliability guarantee is fulfilled at this point.
208 **Transmit:** The act of writing a message to a network connection.
209 **Receive:** The act of reading a message from a network connection.
210 **Acknowledgement:** The communication from the RM Destination to the RM Source
211 indicating the successful receipt of a message.

212 **2.2 Protocol Preconditions**

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

- 215 • For any single message exchange tThe RM Source MUST have an endpoint reference that
216 uniquely identifies the RM Destination endpoint; correlations across messages addressed to
217 the unique endpoint MUST be meaningful.
- 218 • The RM Source MUST have knowledge of the destination's policies, if any, and the RM
219 Source MUST be capable of formulating messages that adhere to this policy.

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

222 **2.3 Protocol Invariants**

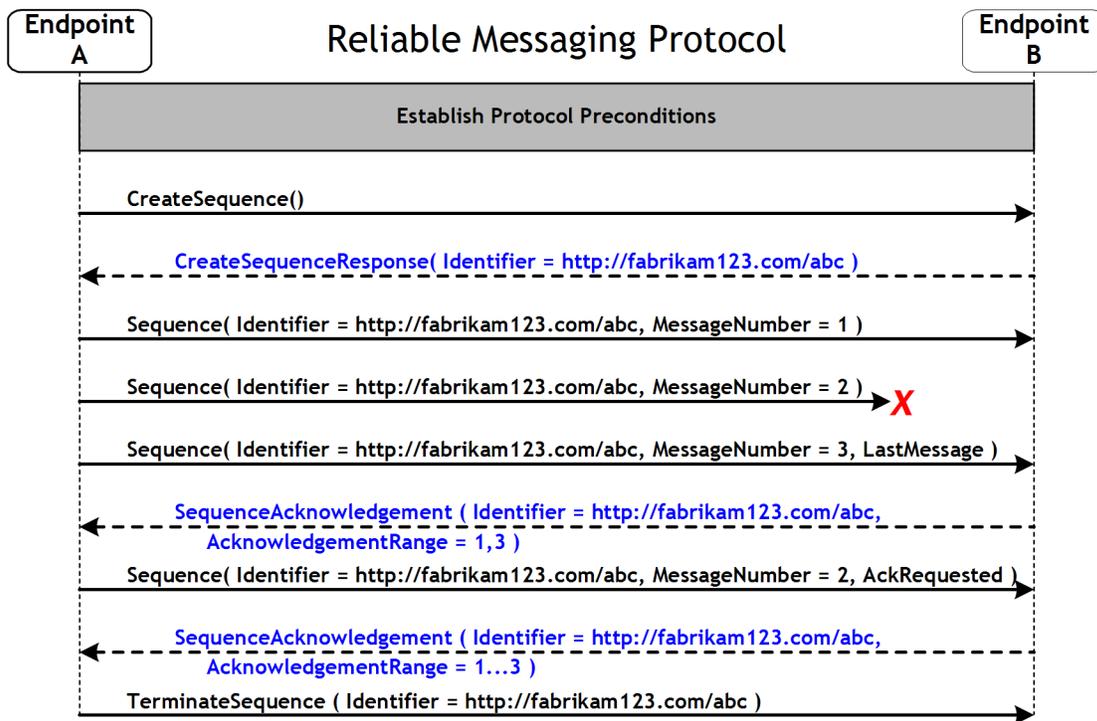
223 During the lifetime of the protocol, two invariants are REQUIRED for correctness:

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

226 Every acknowledgement issued by the RM Destination MUST include within an
 227 acknowledgement range or ranges the sequence number of every message
 228 successfully received by the RM Destination and MUST exclude sequence numbers of
 229 any messages not yet received.

230 2.4 Example Message Exchange

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



233 Figure 2: The WS-ReliableMessaging Protocol

- 234 1. The protocol preconditions are established. These include policy exchange,
 235 endpoint resolution, establishing trust.
- 236 2. The RM Source requests creation of a new Sequence.
- 237 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 238 4. The RM Source begins sending messages beginning with MessageNumber 1. In
 239 the figure the RM Source sends 3 messages.

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

262 3 RM Protocol Elements

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

267 3.1 Sequences

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

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

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

280 A following exemplar defines its syntax:

```
281 <wsrm:Sequence ...>  
282   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
283   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>  
284   <wsrm:LastMessage/>?  
285   ...  
286 </wsrm:Sequence>
```

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

288 `/wsrm:Sequence`

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

294 `/wsrm:Sequence/wsrm:Identifier`

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

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

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

300 /wsrm:Sequence/wsrm:MessageNumber

301 This REQUIRED element MUST contain an xs:unsignedLong representing the ordinal position of
302 the message within a Sequence. Sequence MessageNumbers start at 1 and monotonically
303 increase throughout the Sequence. If the message number exceeds the internal limitations of an
304 RM Source or RM Destination or reaches the maximum value of an xs:unsignedLong
305 (18,446,744,073,709,551,615), the RM Source or Destination MUST issue a
306 MessageNumberRollover fault.

307 /wsrm:Sequence/wsrm:LastMessage

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

310 /wsrm:Sequence/{any}

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

313 /wsrm:Sequence/@{any}

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

316 A RM Source ~~endpoint~~ MUST include a <wsrm:LastMessage> element in the
317 <wsrm:Sequence> element for the last message in a Sequence. An RM Destination
318 ~~endpoint~~ MUST respond with a <wsrm:SequenceAcknowledgement> upon receipt of a
319 <wsrm:LastMessage> element. A Sequence MUST NOT use a <wsrm:MessageNumber>
320 value greater than that which accompanies a <wsrm:LastMessage> element. An RM
321 Destination MUST generate a LastMessageNumberExceeded (See Section 4.6) fault
322 upon receipt of such a message. In the event that an RM Source needs to close a
323 Sequence and there is no application message, the RM Source MAY send a message
324 with an empty body containing <wsrm:Sequence> header with the
325 <wsrm:LastMessage> element. In this usage, the action URI MUST be:

326 `http://docs.oasis-open.org/wsrm/200510/LastMessage`

327 in preference to the pattern defined in Section 1.2.

328 The following example illustrates a Sequence header block.

329 `<wsrm:Sequence>`

```

330     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
331     <wsrm:MessageNumber>10</wsrm:MessageNumber>
332     <wsrm:LastMessage/>
333 </wsrm:Sequence>

```

334 3.2 Sequence Acknowledgement

335 The RM Destination informs the RM Source of successful message receipt using a
336 <wsrm:SequenceAcknowledgement> header block. The
337 <wsrm:SequenceAcknowledgement> header block MAY be transmitted independently
338 or included on return messages. The RM Destination MAY send a
339 <wsrm:SequenceAcknowledgement> header block at any point during which the
340 sequence is valid. The timing of acknowledgements can be advertised using policy
341 and acknowledgements can be explicitly requested using the <wsrm:AckRequested>
342 directive (see Section 3.3). If a non-mustUnderstand fault occurs when processing
343 an RM Header that was piggy-backed on another message, a fault MUST be
344 generated, but the processing of the original message MUST NOT be affected.

345 The following exemplar defines its syntax:

```

346 <wsrm:SequenceAcknowledgement ...>
347   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
348   [ [ <wsrm:AcknowledgementRange ...
349     Upper="xs:unsignedLong"
350     Lower="xs:unsignedLong"/> +
351     <wsrm:Final/> ? ]
352   | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> +
353   | <wsrm:None/> ]
354   ...
355 </wsrm:SequenceAcknowledgement>

```

356 The following describes the content model of the <wsrm:SequenceAcknowledgement>
357 header block.

358 /wsrm:SequenceAcknowledgement

359 This element contains the Sequence acknowledgement information.

360 /wsrm:SequenceAcknowledgement/wsrm:Identifier

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

363 /wsrm:SequenceAcknowledgement/wsrm:Identifier/@{any}

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

366 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange
367 This OPTIONAL element, if present, can occur 1 or more times. It contains a range of message
368 Sequence MessageNumbers successfully received by the **RM Destinationreceiving-endpoint**
369 **manager**. The ranges SHOULD NOT overlap. This element MUST NOT be present if either the
370 <wsrm:Nack> or <wsrm:None> elements are also present as a child of
371 <wsrm:SequenceAcknowledgement>.

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

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

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

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

390 /wsrm:SequenceAcknowledgement/wsrm:Nack
391 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the
392 <wsrm:MessageNumber> of an unreceived message in a Sequence. This element permits the
393 gap analysis of the <wsrm:AcknowledgementRange> elements to be performed at the RM
394 Destination rather than at the RM Source which may yield performance benefits in certain
395 environments. The <wsrm:Nack> element MUST NOT be present if either the
396 <wsrm:AcknowledgementRange> or <wsrm:None> elements are also present as a child of
397 <wsrm:SequenceAcknowledgement>. Upon the receipt of a Nack, an RM Source SHOULD
398 retransmit the message identified by the Nack. The RM Destination MUST NOT issue a
399 <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a message that it has
400 previously acknowledged within a <wsrm:AcknowledgementRange>. The RM Source SHOULD
401 ignore a <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a message
402 that has previously been acknowledged within a <wsrm:AcknowledgementRange>.

403 /wsrm:SequenceAcknowledgement/wsrm:None

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

408 /wsrm:SequenceAcknowledgement/{any}

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

411 /wsrm:SequenceAcknowledgement/@{any}

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

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

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

```
416 <wsrm:SequenceAcknowledgement>  
417     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
418     <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
419 </wsrm:SequenceAcknowledgement>
```

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

```
422 <wsrm:SequenceAcknowledgement>  
423     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
424     <wsrm:AcknowledgementRange Upper="2" Lower="1"/>  
425     <wsrm:AcknowledgementRange Upper="6" Lower="4"/>  
426     <wsrm:AcknowledgementRange Upper="10" Lower="8"/>  
427 </wsrm:SequenceAcknowledgement>
```

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

```
429 <wsrm:SequenceAcknowledgement>  
430     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
431     <wsrm:Nack>3</wsrm:Nack>  
432 </wsrm:SequenceAcknowledgement>
```

433 3.3 Request Acknowledgement

434 The purpose of the <wsrm:AckRequested> header block is to signal to the RM
435 Destination that the RM Source is requesting that a
436 <wsrm:SequenceAcknowledgement> be returned.

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

439 The RM Source **endpoint** requests this acknowledgement by including an
440 `<wsrm:AckRequested>` header block in the message. An RM Destination that receives
441 a message that contains an `<wsrm:AckRequested>` header block MUST respond with
442 a message containing a `<wsrm:SequenceAcknowledgement>` header block. If a non-
443 mustUnderstand fault occurs when processing an RM Header that was piggy-backed
444 on another message, a fault MUST be generated, but the processing of the original
445 message MUST NOT be affected.

446 The following exemplar defines its syntax:

```
447 <wsrm:AckRequested ...>  
448   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
449   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber> ?  
450   ...  
451 </wsrm:AckRequested>
```

452 `/wsrm:AckRequested`

453 This element requests an acknowledgement for the identified sequence.

454 `/wsrm:AckRequested/wsrm:Identifier`

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

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

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

460 `/wsrm:AckRequested/wsrm:MessageNumber`

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

465 `/wsrm:AckRequested/{any}`

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

468 `/wsrm:AckRequested/@{any}`

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

471 3.4 Sequence Creation

472 The RM Source MUST request creation of an outbound Sequence by sending a
473 `<wsrm:CreateSequence>` element in the body of a message to the RM Destination
474 which in turn responds either with a `<wsrm:CreateSequenceResponse>` or a
475 `CreateSequenceRefused` fault in the body of the response message.
476 `<wsrm:CreateSequence>` MAY carry an offer to create an inbound sequence which is
477 either accepted or rejected in the `<wsrm:CreateSequenceResponse>`.

478 ~~The RM Destination of the outbound sequence is the WS-Addressing~~
479 ~~EndpointReference [WS-Addressing] to which `<wsrm:CreateSequence>` is sent. The~~
480 ~~RM Destination of the inbound sequence is the WS-Addressing `<wsa:ReplyTo>` of the~~
481 ~~`<wsrm:CreateSequence>`.~~

482 The following exemplar defines the `<wsrm:CreateSequence>` syntax:

```
483 <wsrm:CreateSequence ...>  
484   <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>  
485   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
486   <wsrm:Offer ...>  
487     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
488     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
489     ...  
490   </wsrm:Offer> ?  
491   ...  
492 </wsrm:CreateSequence>
```

493 `/wsrm:CreateSequence`

494 This element requests creation of a new Sequence between the RM Source that sends it, and the
495 RM Destination to which it is sent. This element MUST NOT be sent as a header block. The RM
496 Destination MUST respond either with a `<wsrm:CreateSequenceResponse>` response
497 message or a `CreateSequenceRefused` fault.

498 `/wsrm:CreateSequence/wsrm:AcksTo`

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

503 `/wsrm:CreateSequence/wsrm:Expires`

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

508 /wsm:CreateSequence/wsm:Expires/@{any}
509 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
510 to the element.

511 /wsm:CreateSequence/wsm:Offer
512 This element, if present, enables an RM Source to offer a corresponding Sequence for the reliable
513 exchange of messages transmitted from RM Destination to RM Source.

514 /wsm:CreateSequence/wsm:Offer/wsm:Identifier
515 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely
516 identifies the offered Sequence.

517 /wsm:CreateSequence/wsm:Offer/wsm:Identifier/@{any}
518 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
519 to the element.

520 /wsm:CreateSequence/wsm:Offer/wsm:Expires
521 This element, if present, of type `xs:duration` specifies the duration for the Sequence. A value
522 of 'PT0S' indicates that the Sequence will never expire. Absence of the element indicates an
523 implied value of 'PT0S'.

524 /wsm:CreateSequence/wsm:Offer/wsm:Expires/@{any}
525 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
526 to the element.

527 /wsm:CreateSequence/wsm:Offer/{any}
528 This is an extensibility mechanism to allow different (extensible) types of information, based on a
529 schema, to be passed.

530 /wsm:CreateSequence/wsm:Offer/@{any}
531 This is an extensibility mechanism to allow different (extensible) types of information, based on a
532 schema, to be passed.

533 OPTIONAL/wsm:CreateSequence/{any}
534 This is an extensibility mechanism to allow different (extensible) types of information, based on a
535 schema, to be passed.

536 /wsm:CreateSequence/@{any}
537 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
538 to the element.

539 A `<wsrm:CreateSequenceResponse>` is sent in the body of a response message by an
540 RM Destination in response to receipt of a `<wsrm:CreateSequence>` request
541 message. It carries the `<wsrm:Identifier>` of the created Sequence and indicates
542 that the RM Source may begin sending messages in the context of the identified
543 Sequence.

544 The following exemplar defines the `<wsrm:CreateSequenceResponse>` syntax:

```
545 <wsrm:CreateSequenceResponse ...>  
546   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
547   <wsrm:Expires> xs:duration </wsrm:Expires> ?  
548   <wsrm:Accept ...>  
549     <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>  
550     ...  
551   </wsrm:Accept> ?  
552   ...  
553 </wsrm:CreateSequenceResponse>
```

554 `/wsrm:CreateSequenceResponse`

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

559 `/wsrm:CreateSequenceResponse/wsrm:Identifier`

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

562 `/wsrm:CreateSequenceResponse/wsrm:Identifier/@{any}`

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

565 `/wsrm:CreateSequenceResponse/wsrm:Expires`

566 This element, if present, of type `xs:duration` accepts or refines the RM Source's requested
567 duration for the Sequence. A value of 'PT0S' indicates that the Sequence will never expire.
568 Absence of the element indicates an implied value of 'PT0S'. This value MUST be equal or lesser
569 than the value requested by the RM Source in the corresponding `<wsrm:CreateSequence>`
570 message.

571 `/wsrm:CreateSequenceResponse/wsrm:Expires/@{any}`

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

574 `/wsrm:CreateSequenceResponse/wsrm:Accept`

575 This element, if present, enables an RM Destination to accept the offer of a corresponding
576 Sequence for the reliable exchange of messages transmitted from RM Destination to RM Source.
577 This element MUST be present if the corresponding `<wsrm:CreateSequence>` message
578 contained an `<wsrm:Offer>` element.

579 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

580 This REQUIRED element, of type `wsa:EndpointReferenceType` as specified by WS-Addressing
581 [WS-Addressing], specifies the endpoint reference to which
582 `<wsrm:SequenceAcknowledgement>` messages related to the accepted Sequence are to be
583 sent.

584 `/wsrm:CreateSequenceResponse/wsrm:Accept/{any}`

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

587 `/wsrm:CreateSequenceResponse/wsrm:Accept/@{any}`

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

590 `/wsrm:CreateSequenceResponse/{any}`

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

593 `/wsrm:CreateSequenceResponse/@{any}`

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

596 **3.5 Sequence Termination**

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

606 The following exemplar defines the TerminateSequence syntax:

```
607 <wsrm:TerminateSequence ...>  
608   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
```

609

610

```
...  
</wsrm:TerminateSequence>
```

611 /wsrm:TerminateSequence

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

616 /wsrm:TerminateSequence/wsrm:Identifier

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

619 /wsrm:TerminateSequence/wsrm:Identifier/@{any}

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

622 /wsrm:TerminateSequence/{any}

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

625 /wsrm:TerminateSequence/@{any}

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

628 **3.6 Closing A Sequence**

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

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

637 To alleviate this, the RM Source can send a <wsrm:CloseSequence> element, in the
638 body of a message, to the RM Destination to indicate that RM Destination MUST NOT
639 receive any new messages for the specified sequence, other than those already
640 received at the time the <wsrm:CloseSequence> element is interpreted by the RMD.
641 Upon receipt of this message the RM Destination MUST send a

642 SequenceAcknowledgement to the RM Source. Note, this
643 SequenceAcknowledgement MUST include the <wsrm:Final> element.

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

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

653 The following exemplar defines the CloseSequence syntax:

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

655 /wsrm:CloseSequence

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

659 /wsrm:CloseSequence@Identifier

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

662 /wsrm:CloseSequence/{any}

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

665 /wsrm:CloseSequence@{any}

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

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

671 The following exemplar defines the <wsrm:CloseSequenceResponse> syntax:

```
672 /wsrm:CloseSequenceResponse
```

673 /wsrm:CloseSequenceResponse

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

677 /wsrm:CloseSequenceResponse/{any}

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

680 /wsrm:CloseSequenceResponse@{any}

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

683 4 Faults

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

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

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

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

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

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

704 The definitions of faults use the following properties:

705 [Code] The fault code.

706 [Subcode] The fault subcode.

707 [Reason] The English language reason element.

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

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

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

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

issue10 10/27/2005

```

712 <S:Envelope>
713   <S:Header>
714     <wsa:Action>
715       http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
716     </wsa:Action>
717     <!-- Headers elided for clarity. -->
718   </S:Header>
719   <S:Body>
720     <S:Fault>
721       <S:Code>
722         <S:Value> [Code] </S:Value>
723         <S:Subcode>
724           <S:Value> [Subcode] </S:Value>
725         </S:Subcode>
726       </S:Code>
727       <S:Reason>
728         <S:Text xml:lang="en"> [Reason] </S:Text>
729       </S:Reason>
730       <S:Detail>
731         [Detail]
732         ...
733       </S:Detail>
734     </S:Fault>
735   </S:Body>
736 </S:Envelope>

```

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

```

739 <S11:Envelope>
740   <S11:Header>
741     <wsrm:SequenceFault>
742       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
743       ...
744     </wsrm:SequenceFault>
745     <!-- Headers elided for clarity. -->
746   </S11:Header>
747   <S11:Body>
748     <S11:Fault>
749       <faultcode> [Code] </faultcode>
750       <faultstring> [Reason] </faultstring>
751     </S11:Fault>
752   </S11:Body>

```

753 `</S11:Envelope>`

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

```
756 <S11:Envelope>
757   <S11:Body>
758     <S11:Fault>
759       <faultcode> [Subcode] </faultcode>
760       <faultstring xml:lang="en"> [Reason] </faultstring>
761     </S11:Fault>
762   </S11:Body>
763 </S11:Envelope>
```

764 4.1 SequenceFault Element

765 The purpose of the `<wsrm:SequenceFault>` element is to carry the specific details of
766 a fault generated during the reliable messaging specific processing of a message
767 belonging to a Sequence. The `<wsrm:SequenceFault>` container MUST only be used
768 in conjunction with the SOAP1.1 fault mechanism. It MUST NOT be used in
769 conjunction with the SOAP1.2 binding.

770 The following exemplar defines its syntax:

```
771 <wsrm:SequenceFault ...>
772   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
773   ...
774 </wsrm:SequenceFault>
```

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

776 `/wsrm:SequenceFault`

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

778 `/wsrm:SequenceFault/wsrm:FaultCode`

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

781 `/wsrm:SequenceFault/{any}`

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

784 `/wsrm:SequenceFault/@{any}`

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

787 **4.2 Sequence Terminated**

788 This fault is sent by either the RM Source or the RM Destination to indicate that ~~the~~
789 ~~endpoint that generated the fault it~~ has either encountered an unrecoverable
790 condition, or has detected a violation of the protocol and as a consequence, has
791 chosen to terminate the sequence. The endpoint that generates this fault should
792 make every reasonable effort to notify the corresponding endpoint of this decision.

793 Properties:

794 [Code] Sender or Receiver

795 [Subcode] wsrn:SequenceTerminated

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

797 [Detail]

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

799 **4.3 Unknown Sequence**

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

802 Properties:

803 [Code] Sender

804 [Subcode] wsrn:UnknownSequence

805 [Reason] The value of wsrn:Identifier is not a known Sequence identifier.

806 [Detail]

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

808 **4.4 Invalid Acknowledgement**

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

813 [Code] Sender

814 [Subcode] wsrn:InvalidAcknowledgement
815 [Reason] The SequenceAcknowledgement violates the cumulative acknowledgement
816 invariant.

817 [Detail]

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

819 **4.5 Message Number Rollover**

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

822 Properties:

823 [Code] Sender

824 [Subcode] wsrn:MessageNumberRollover

825 [Reason] The maximum value for wsrn:MessageNumber has been exceeded.

826 [Detail]

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

828 **4.6 Last Message Number Exceeded**

829 This fault is sent by an RM Destination to indicate that it has received a message that
830 has a <wsrm:MessageNumber> within a Sequence that exceeds the value of the
831 <wsrm:MessageNumber> element that accompanied a <wsrm>LastMessage> element
832 for the Sequence.

833 Properties:

834 [Code] Sender

835 [Subcode] wsrn:LastMessageNumberExceeded

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

838 [Detail]

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

840 **4.7 Create Sequence Refused**

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

842 Properties:

843 [Code] Sender

844 [Subcode] wsrn:CreateSequenceRefused

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

846 [Detail] empty

847 **4.8 Sequence Closed**

848 This fault is sent by an RM Destination to indicate that the specified sequence has
849 been closed. This fault **MUST** be generated when an RM Destination is asked to
850 receive a message for a sequence that is closed.

851 Properties:

852 [Code] Sender

853 [Subcode] wsrn:SequenceClosed

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

855 [Detail] <wsrn:Identifier...> xs:anyURI </wsrn:Identifier>

856 **5 Security Considerations**

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

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

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

- 881 • Closing and re-establishing a security context
- 882 • Exchanging new secrets between the parties
- 883 • Using a derived key sequence and switch "generations"
- 884 • Attaching a nonce to each message and using it in a derived key function with the shared
885 secret

886 The security context may be re-established using the mechanisms described in WS-
887 Trust and WS-SecureConversation. Similarly, secrets can be exchanged using the
888 mechanisms described in WS-Trust. Note, however, that the current shared secret
889 should not be used to encrypt the new shared secret. Derived keys, the preferred
890 solution from this list, can be specified using the mechanisms described in WS-
891 SecureConversation.

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

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

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

923 **6 References**

924 **6.1 Normative**

925 **[KEYWORDS]**

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

928 **[SOAP]**

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

930 **[URI]**

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

933 **[XML-ns]**

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

935 **[XML-Schema1]**

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

937 **[XML-Schema2]**

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

939 **[WSSecurity]**

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

943 **[Tanenbaum]**

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

945 **[WSDL]**

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

947 **[WS-Addressing]**

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

949 **6.2 Non-Normative**

950 **[WS-Policy]**

issue10

10/27/2005

Copyright © OASIS Open 2005. All Rights Reserved.

Page 34 of 54

- 951 D. Box, et al, "[Web Services Policy Framework \(WS-Policy\)](#)," September 2004.
- 952 **[WS-PolicyAttachment]**
- 953 D. Box, et al, "[Web Services Policy Attachment \(WS-PolicyAttachment\)](#)," September 2004.
- 954 **[SecurityPolicy]**
- 955 G. Della-Libra, "[Web Services Security Policy Language \(WS-SecurityPolicy\)](#)," December 2002.
- 956 **[SecureConversation]**
- 957 S. Anderson, et al, "[Web Services Secure Conversation Language \(WS-SecureConversation\)](#),"
- 958 May 2004.
- 959

960 Appendix A.Schema

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

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

963 The following copy is provided for reference.

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

```

998         <xs:attribute name="Upper" type="xs:unsignedLong"
999 use="required"/>
1000         <xs:attribute name="Lower" type="xs:unsignedLong"
1001 use="required"/>
1002         <xs:anyAttribute namespace="##other"
1003 processContents="lax"/>
1004     </xs:complexType>
1005 </xs:element>
1006 <ws:element name="Final" minOccurs="0">
1007     <xs:complexType>
1008         <xs:sequence/>
1009     </xs:complexType>
1010 </ws:element>
1011 </ws:sequence>
1012 <xs:element name="Nack" type="xs:unsignedLong"
1013 minOccurs="unbounded"/>
1014     <xs:element name="None" minOccurs="0">
1015         <xs:complexType>
1016             <xs:sequence/>
1017         </xs:complexType>
1018     </xs:element>
1019 </xs:choice>
1020     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1021 minOccurs="unbounded"/>
1022 </xs:sequence>
1023 <xs:anyAttribute namespace="##other" processContents="lax"/>
1024 </xs:complexType>
1025 </xs:element>
1026 <xs:complexType name="AckRequestedType">
1027     <xs:sequence>
1028         <xs:element ref="wsrm:Identifier"/>
1029         <xs:element name="MessageNumber" type="xs:unsignedLong"
1030 minOccurs="0"/>
1031     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1032 minOccurs="unbounded"/>
1033 </xs:sequence>
1034 <xs:anyAttribute namespace="##other" processContents="lax"/>
1035 </xs:complexType>
1036 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1037 <xs:element name="Identifier">
1038     <xs:complexType>
1039         <xs:annotation>

```

```

1040     <xs:documentation>
1041 This type is for elements whose [children] is an anyURI and can have
1042 arbitrary attributes.
1043     </xs:documentation>
1044 </xs:annotation>
1045 <xs:simpleContent>
1046     <xs:extension base="xs:anyURI">
1047         <xs:anyAttribute namespace="##other" processContents="lax"/>
1048     </xs:extension>
1049 </xs:simpleContent>
1050 </xs:complexType>
1051 </xs:element>
1052 <!-- Fault Container and Codes -->
1053 <xs:simpleType name="FaultCodes">
1054     <xs:restriction base="xs:QName">
1055         <xs:enumeration value="wsrm:UnknownSequence"/>
1056         <xs:enumeration value="wsrm:SequenceTerminated"/>
1057         <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1058         <xs:enumeration value="wsrm:MessageNumberRollover"/>
1059         <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1060         <xs:enumeration value="wsrm:LastMessageNumberExceeded"/>
1061     </xs:restriction>
1062 </xs:simpleType>
1063 <xs:complexType name="SequenceFaultType">
1064     <xs:sequence>
1065         <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1066         <xs:any namespace="##any" processContents="lax" minOccurs="0"
1067 maxOccurs="unbounded"/>
1068     </xs:sequence>
1069     <xs:anyAttribute namespace="##any" processContents="lax"/>
1070 </xs:complexType>
1071 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1072 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1073 <xs:element name="CreateSequenceResponse"
1074 type="wsrm:CreateSequenceResponseType"/>
1075 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1076 <xs:element name="CloseSequenceResponse"
1077 type="wsrm:CloseSequenceResponseType"/>
1078 <xs:element name="TerminateSequence"
1079 type="wsrm:TerminateSequenceType"/>
1080 <xs:complexType name="CreateSequenceType">
1081     <xs:sequence>

```

```

1082     <xs:element ref="wsrm:AcksTo"/>
1083     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1084     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1085     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1086 maxOccurs="unbounded">
1087         <xs:annotation>
1088             <xs:documentation>
1089 It is the authors intent that this extensibility be used to transfer a
1090 Security Token Reference as defined in WS-Security.
1091 </xs:documentation>
1092         </xs:annotation>
1093     </xs:any>
1094 </xs:sequence>
1095 <xs:anyAttribute namespace="##other" processContents="lax"/>
1096 </xs:complexType>
1097 <xs:complexType name="CreateSequenceResponseType">
1098     <xs:sequence>
1099         <xs:element ref="wsrm:Identifier"/>
1100         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1101         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1102         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1103 maxOccurs="unbounded"/>
1104     </xs:sequence>
1105     <xs:anyAttribute namespace="##other" processContents="lax"/>
1106 </xs:complexType>
1107 <xs:complexType name="CloseSequenceType">
1108     <xs:sequence>
1109         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1110 maxOccurs="unbounded"/>
1111     </xs:sequence>
1112     <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1113     <xs:anyAttribute namespace="##other" processContents="lax"/>
1114 </xs:complexType>
1115 <xs:complexType name="CloseSequenceResponseType">
1116     <xs:sequence>
1117         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1118 maxOccurs="unbounded"/>
1119     </xs:sequence>
1120     <xs:anyAttribute namespace="##other" processContents="lax"/>
1121 </xs:complexType>
1122 <xs:complexType name="TerminateSequenceType">
1123     <xs:sequence>

```

```

1124     <xs:element ref="wsrm:Identifier"/>
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:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1131 <xs:complexType name="OfferType">
1132   <xs:sequence>
1133     <xs:element ref="wsrm:Identifier"/>
1134     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1135     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1136 maxOccurs="unbounded"/>
1137   </xs:sequence>
1138   <xs:anyAttribute namespace="##other" processContents="lax"/>
1139 </xs:complexType>
1140 <xs:complexType name="AcceptType">
1141   <xs:sequence>
1142     <xs:element ref="wsrm:AcksTo"/>
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:element name="Expires">
1149   <xs:complexType>
1150     <xs:simpleContent>
1151       <xs:extension base="xs:duration">
1152         <xs:anyAttribute namespace="##other" processContents="lax"/>
1153       </xs:extension>
1154     </xs:simpleContent>
1155   </xs:complexType>
1156 </xs:element>
1157 </xs:schema>

```

1158 **Appendix B.Message Examples**

1159 B.1.Create Sequence

1160 Create Sequence

```
1161 <?xml version="1.0" encoding="UTF-8"?>
1162 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1163 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1164 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1165   <S:Header>
1166     <wsa:MessageID>
1167       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1168     </wsa:MessageID>
1169     <wsa:To>http://example.com/serviceB/123</wsa:To>
1170     <wsa:Action>http://docs.oasis-
1171 open.org/wsmr/200510/CreateSequence</wsa:Action>
1172     <wsa:ReplyTo>
1173       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1174     </wsa:ReplyTo>
1175   </S:Header>
1176   <S:Body>
1177     <wsmr:CreateSequence>
1178       <wsmr:AcksTo>
1179         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1180       </wsmr:AcksTo>
1181     </wsmr:CreateSequence>
1182   </S:Body>
1183 </S:Envelope>
```

1184 Create Sequence Response

```
1185 <?xml version="1.0" encoding="UTF-8"?>
1186 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1187 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1188 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1189   <S:Header>
1190     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1191     <wsa:RelatesTo>
1192       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1193     </wsa:RelatesTo>
1194     <wsa:Action>
1195       http://docs.oasis-open.org/wsmr/200510/CreateSequenceResponse
```

```
1196     </wsa:Action>
1197 </S:Header>
1198 <S:Body>
1199     <wsrm:CreateSequenceResponse>
1200         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1201     </wsrm:CreateSequenceResponse>
1202 </S:Body>
1203 </S:Envelope>
```

1204 B.2. Initial Transmission

1205 The following example WS-ReliableMessaging headers illustrate the message
1206 exchange in the above figure. The three messages have the following headers; the
1207 third message is identified as the last message in the sequence:

1208 Message 1

```
1209 <?xml version="1.0" encoding="UTF-8"?>
1210 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1211 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1212 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1213   <S:Header>
1214     <wsa:MessageID>
1215       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1216     </wsa:MessageID>
1217     <wsa:To>http://example.com/serviceB/123</wsa:To>
1218     <wsa:From>
1219       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1220     </wsa:From>
1221     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1222     <wsm:Sequence>
1223       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1224       <wsm:MessageNumber>1</wsm:MessageNumber>
1225     </wsm:Sequence>
1226   </S:Header>
1227   <S:Body>
1228     <!-- Some Application Data -->
1229   </S:Body>
1230 </S:Envelope>
```

1231 Message 2

```
1232 <?xml version="1.0" encoding="UTF-8"?>
1233 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1234 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1235 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1236   <S:Header>
1237     <wsa:MessageID>
1238       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1239     </wsa:MessageID>
1240     <wsa:To>http://example.com/serviceB/123</wsa:To>
```

```
1241 <wsa:From>
1242   <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1243 </wsa:From>
1244 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1245 <wsrm:Sequence>
1246   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1247   <wsrm:MessageNumber>2</wsrm:MessageNumber>
1248 </wsrm:Sequence>
1249 </S:Header>
1250 <S:Body>
1251   <!-- Some Application Data -->
1252 </S:Body>
1253 </S:Envelope>
```

1254 Message 3

```
1255 <?xml version="1.0" encoding="UTF-8"?>
1256 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1257 xmlns:wsrm="http://docs.oasis-open.org/wsrn/200510/"
1258 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1259   <S:Header>
1260     <wsa:MessageID>
1261       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1262     </wsa:MessageID>
1263     <wsa:To>http://example.com/serviceB/123</wsa:To>
1264     <wsa:From>
1265       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1266     </wsa:From>
1267     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1268     <wsrm:Sequence>
1269       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1270       <wsrm:MessageNumber>3</wsrm:MessageNumber>
1271       <wsrm:LastMessage/>
1272     </wsrm:Sequence>
1273   </S:Header>
1274   <S:Body>
1275     <!-- Some Application Data -->
1276   </S:Body>
1277 </S:Envelope>
```

1278 B.3.First Acknowledgement

1279 Message number 2 has not been received by the RM Destination due to some
1280 transmission error so it responds with an acknowledgement for messages 1 and 3:

```
1281 <?xml version="1.0" encoding="UTF-8"?>
1282 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1283 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1284 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1285   <S:Header>
1286     <wsa:MessageID>
1287       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1288     </wsa:MessageID>
1289     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1290     <wsa:From>
1291       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1292     </wsa:From>
1293     <wsa:Action>
1294       http://docs.oasis-open.org/wsm/200510/SequenceAcknowledgement
1295     </wsa:Action>
1296     <wsm:SequenceAcknowledgement>
1297       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1298       <wsm:AcknowledgementRange Upper="1" Lower="1"/>
1299       <wsm:AcknowledgementRange Upper="3" Lower="3"/>
1300     </wsm:SequenceAcknowledgement>
1301   </S:Header>
1302   <S:Body/>
1303 </S:Envelope>
```

1304 B.4.Retransmission

1305 The ~~sending-endpoint~~RM Source discovers that message number 2 was not received
1306 so it resends the message and requests an acknowledgement:

```
1307 <?xml version="1.0" encoding="UTF-8"?>
1308 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1309 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1310 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1311   <S:Header>
1312     <wsa:MessageID>
1313       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1314     </wsa:MessageID>
1315     <wsa:To>http://example.com/serviceB/123</wsa:To>
1316     <wsa:From>
1317       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1318     </wsa:From>
1319     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1320     <wsmr:Sequence>
1321       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1322       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1323     </wsmr:Sequence>
1324     <wsmr:AckRequested>
1325       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1326     </wsmr:AckRequested>
1327   </S:Header>
1328   <S:Body>
1329     <!-- Some Application Data -->
1330   </S:Body>
1331 </S:Envelope>
```

1332 B.5.Termination

1333 The RM Destination now responds with an acknowledgement for the complete
1334 sequence which can then be terminated:

```
1335 <?xml version="1.0" encoding="UTF-8"?>
1336 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1337 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1338 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1339   <S:Header>
1340     <wsa:MessageID>
1341       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1342     </wsa:MessageID>
1343     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1344     <wsa:From>
1345       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1346     </wsa:From>
1347     <wsa:Action>
1348       http://docs.oasis-open.org/wsm/200510/SequenceAcknowledgement
1349     </wsa:Action>
1350     <wsm:SequenceAcknowledgement>
1351       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1352       <wsm:AcknowledgementRange Upper="3" Lower="1"/>
1353     </wsm:SequenceAcknowledgement>
1354   </S:Header>
1355   <S:Body/>
1356 </S:Envelope>
```

1357 Terminate Sequence

```
1358 <?xml version="1.0" encoding="UTF-8"?>
1359 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1360 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1361 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1362   <S:Header>
1363     <wsa:MessageID>
1364       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1365     </wsa:MessageID>
1366     <wsa:To>http://example.com/serviceB/123</wsa:To>
1367     <wsa:Action>
1368       http://docs.oasis-open.org/wsm/200510/TerminateSequence
1369     </wsa:Action>
```

```
1370     <wsa:From>
1371     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1372     </wsa:From>
1373 </S:Header>
1374 <S:Body>
1375     <wsrm:TerminateSequence>
1376     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1377     </wsrm:TerminateSequence>
1378 </S:Body>
1379 </S:Envelope>
```

1380 Appendix C.WSDL

1381 The non-normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1382 <http://docs.oasis-open.org/wsrn/200510/wsd/wsrn.wsd>

1383 The following non-normative copy is provided for reference.

```
1384 <wSDL:definitions xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
1385 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1386 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1387 xmlns:rm="http://docs.oasis-open.org/wsrn/200510/"
1388 xmlns:tns="http://docs.oasis-open.org/wsrn/200510/wsd"
1389 targetNamespace="http://docs.oasis-open.org/wsrn/200510/wsd">
1390 <wSDL:types>
1391   <xs:schema>
1392     <xs:import namespace="http://docs.oasis-open.org/wsrn/200510/"
1393     schemaLocation="http://docs.oasis-open.org/wsrn/200510/wsrn.xsd"/>
1394     <xs:import
1395     namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1396     schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1397   </xs:schema>
1398 </wSDL:types>
1399 <wSDL:message name="CreateSequence">
1400   <wSDL:part name="create" element="rm:CreateSequence"/>
1401 </wSDL:message>
1402 <wSDL:message name="CreateSequenceResponse">
1403   <wSDL:part name="createResponse"
1404   element="rm:CreateSequenceResponse"/>
1405 </wSDL:message>
1406 <wSDL:message name="CloseSequence">
1407   <wSDL:part name="close" element="rm:CloseSequence"/>
1408 </wSDL:message>
1409 <wSDL:message name="CloseSequenceResponse">
1410   <wSDL:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1411 </wSDL:message>
1412 <wSDL:message name="TerminateSequence">
1413   <wSDL:part name="terminate" element="rm:TerminateSequence"/>
1414 </wSDL:message>
1415 <wSDL:portType name="SequenceAbstractPortType">
1416   <wSDL:operation name="CreateSequence">
```

```
1417     <wsdl:input message="tns:CreateSequence"
1418 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CreateSequence"/>
1419     <wsdl:output message="tns:CreateSequenceResponse"
1420 wsa:Action="http://docs.oasis-
1421 open.org/wsrn/200510/CreateSequenceResponse"/>
1422 </wsdl:operation>
1423 <wsdl:operation name="CloseSequence">
1424 <wsdl:input name="tns:CloseSequence"
1425 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CloseSequence"/>
1426 <wsdl:output name="tns:CloseSequenceResponse"
1427 wsa:Action="http://docs.oasis-
1428 open.org/wsrn/200510/CloseSequenceResponse"/>
1429 </wsdl:operation>
1430 <wsdl:operation name="TerminateSequence">
1431 <wsdl:input message="tns:TerminateSequence"
1432 wsa:Action="http://docs.oasis-
1433 open.org/wsrn/200510/CreateSequenceResponse"/>
1434 </wsdl:operation>
1435 </wsdl:portType>
1436 </wsdl:definitions>
```

1437 **Appendix D.Acknowledgments**

1438 This document is based on initial contribution to OASIS WS-RX Technical Committee by the
1439 following authors: Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft,
1440 Doug Davis, IBM, Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM,
1441 Mary Ann Hondo, IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David
1442 Langworthy, Microsoft (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft,
1443 Steve Lucco, Microsoft, Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham,
1444 BEA, David Orchard, BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John
1445 Shewchuk, Microsoft, Tony Storey, IBM

1446 The following individuals have provided invaluable input into the initial contribution:

1447 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,
1448 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,
1449 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,
1450 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin
1451 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,
1452 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger
1453 Wolter, Microsoft

1454 The following individuals were members of the committee during the development of this
1455 specification:

1456 TBD

1457 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 (PTOS) 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 'optional'/'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)

1458 **Appendix F. Notices**

1459 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1460 that might be claimed to pertain to the implementation or use of the technology described in this
1461 document or the extent to which any license under such rights might or might not be available;
1462 neither does it represent that it has made any effort to identify any such rights. Information on
1463 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1464 website. Copies of claims of rights made available for publication and any assurances of licenses
1465 to be made available, or the result of an attempt made to obtain a general license or permission
1466 for the use of such proprietary rights by implementors or users of this specification, can be
1467 obtained from the OASIS Executive Director.

1468 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1469 applications, or other proprietary rights which may cover technology that may be required to
1470 implement this specification. Please address the information to the OASIS Executive Director.

1471 Copyright (C) OASIS Open (2005). All Rights Reserved.

1472 This document and translations of it may be copied and furnished to others, and derivative works
1473 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1474 published and distributed, in whole or in part, without restriction of any kind, provided that the
1475 above copyright notice and this paragraph are included on all such copies and derivative works.
1476 However, this document itself may not be modified in any way, such as by removing the copyright
1477 notice or references to OASIS, except as needed for the purpose of developing OASIS
1478 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1479 Property Rights document must be followed, or as required to translate it into languages other
1480 than English.

1481 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1482 successors or assigns.

1483 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1484 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1485 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
1486 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1487 PARTICULAR PURPOSE.