



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 exactly two parties, a source and a~~
81 ~~destination~~. It also defines a SOAP binding that is required for interoperability.
82 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 make no restriction on the scope of the RM
147 Source and RM Destination entities.

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

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

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

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

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

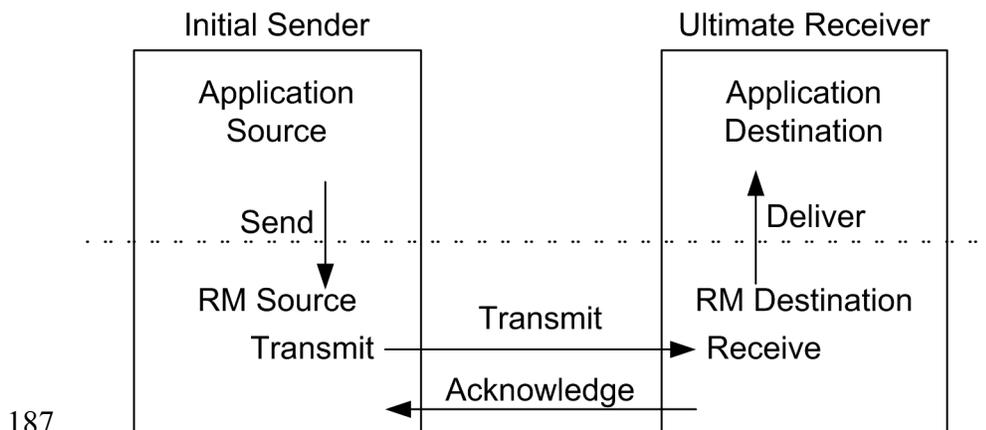
165 **AtMostOnce** Messages will be delivered at most once without duplication or an error
166 will be raised on at least one endpoint. It is possible that some messages in a
167 sequence may not be delivered.

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

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

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

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



188 Figure 1: Reliable Messaging Model

189 2.1 Glossary

190 The following definitions are used throughout this specification:

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

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

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

209 **2.2 Protocol Preconditions**

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

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

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

219 **2.3 Protocol Invariants**

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

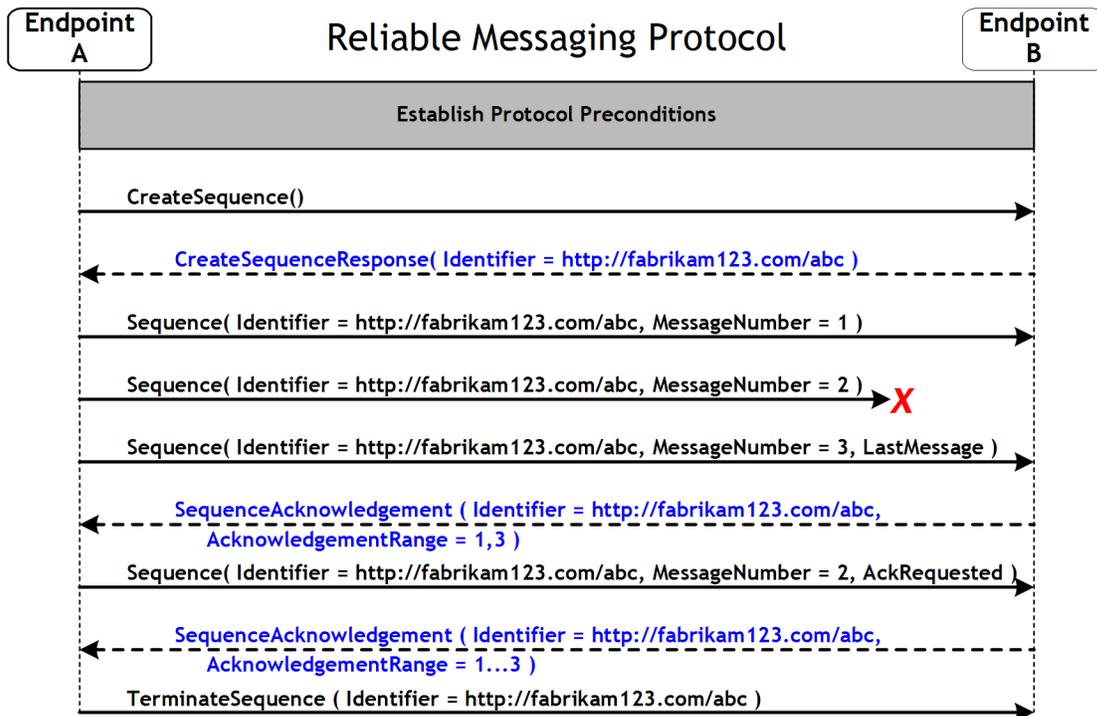
- 221 • The RM Source MUST assign each reliable message a sequence number (defined below)
222 beginning at 1 and increasing by exactly 1 for each subsequent reliable message.

223 Every acknowledgement issued by the RM Destination MUST include within an
224 acknowledgement range or ranges the sequence number of every message

225 successfully received by the RM Destination and MUST exclude sequence numbers of
 226 any messages not yet received.

227 2.4 Example Message Exchange

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



230 Figure 2: The WS-ReliableMessaging Protocol

- 231 1. The protocol preconditions are established. These include policy exchange,
 232 endpoint resolution, establishing trust.
- 233 2. The RM Source requests creation of a new Sequence.
- 234 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 235 4. The RM Source begins sending messages beginning with MessageNumber 1. In
 236 the figure the RM Source sends 3 messages.
- 237 5. Since the 3rd message is the last in this exchange, the RM Source includes a
 238 <wsrm:LastMessage> token.
- 239 6. The 2nd message is lost in transit.

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

259 **3 RM Protocol Elements**

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

264 **3.1 Sequences**

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

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

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

277 A following exemplar defines its syntax:

```
278 <wsrm:Sequence ...>  
279   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
280   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>  
281   <wsrm:LastMessage/>?  
282   ...  
283 </wsrm:Sequence>
```

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

285 `/wsrm:Sequence`

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

291 `/wsrm:Sequence/wsrm:Identifier`

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

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

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

297 /wsrm:Sequence/wsrm:MessageNumber

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

304 /wsrm:Sequence/wsrm:LastMessage

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

307 /wsrm:Sequence/{any}

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

310 /wsrm:Sequence/@{any}

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

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

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

324 in preference to the pattern defined in Section 1.2.

325 The following example illustrates a Sequence header block.

326 `<wsrm:Sequence>`

```

327     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
328     <wsrm:MessageNumber>10</wsrm:MessageNumber>
329     <wsrm:LastMessage/>
330 </wsrm:Sequence>

```

331 3.2 Sequence Acknowledgement

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

342 The following exemplar defines its syntax:

```

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

```

353 The following describes the content model of the <wsrm:SequenceAcknowledgement>
354 header block.

355 /wsrm:SequenceAcknowledgement

356 This element contains the Sequence acknowledgement information.

357 /wsrm:SequenceAcknowledgement/wsrm:Identifier

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

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

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

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

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

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

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

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

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

400 /wsm:SequenceAcknowledgement/wsm:None

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

405 /wsm:SequenceAcknowledgement/{any}

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

408 /wsm:SequenceAcknowledgement/@{any}

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

411 The following examples illustrate <wsm:SequenceAcknowledgement> elements:

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

```
413 <wsm:SequenceAcknowledgement>  
414   <wsm:Identifier>http://example.com/abc</wsm:Identifier>  
415   <wsm:AcknowledgementRange Upper="10" Lower="1"/>  
416 </wsm:SequenceAcknowledgement>
```

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

```
419 <wsm:SequenceAcknowledgement>  
420   <wsm:Identifier>http://example.com/abc</wsm:Identifier>  
421   <wsm:AcknowledgementRange Upper="2" Lower="1"/>  
422   <wsm:AcknowledgementRange Upper="6" Lower="4"/>  
423   <wsm:AcknowledgementRange Upper="10" Lower="8"/>  
424 </wsm:SequenceAcknowledgement>
```

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

```
426 <wsm:SequenceAcknowledgement>  
427   <wsm:Identifier>http://example.com/abc</wsm:Identifier>  
428   <wsm:Nack>3</wsm:Nack>  
429 </wsm:SequenceAcknowledgement>
```

430 3.3 Request Acknowledgement

431 The purpose of the <wsm:AckRequested> header block is to signal to the RM
432 Destination that the RM Source is requesting that a
433 <wsm:SequenceAcknowledgement> be returned.

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

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

443 The following exemplar defines its syntax:

```
444 <wsrm:AckRequested ...>  
445   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
446   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber> ?  
447   ...  
448 </wsrm:AckRequested>
```

449 `/wsrm:AckRequested`

450 This element requests an acknowledgement for the identified sequence.

451 `/wsrm:AckRequested/wsrm:Identifier`

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

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

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

457 `/wsrm:AckRequested/wsrm:MessageNumber`

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

462 `/wsrm:AckRequested/{any}`

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

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

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

468 3.4 Sequence Creation

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

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

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

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

490 `/wsrm:CreateSequence`

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

495 `/wsrm:CreateSequence/wsrm:AcksTo`

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

500 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

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

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

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

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

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

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

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

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

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

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

551 `/wsrm:CreateSequenceResponse`

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

556 `/wsrm:CreateSequenceResponse/wsrm:Identifier`

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

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

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

562 `/wsrm:CreateSequenceResponse/wsrm:Expires`

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

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

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

571 `/wsrm:CreateSequenceResponse/wsrm:Accept`

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

576 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

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

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

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

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/{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 additional attributes, based on schemas, to be added
592 to the element.

593 **3.5 Sequence Termination**

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

603 The following exemplar defines the TerminateSequence syntax:

```
604 <wsrm:TerminateSequence ...>  
605   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
```

606

607

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

608 /wsrm:TerminateSequence

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

613 /wsrm:TerminateSequence/wsrm:Identifier

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

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

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

619 /wsrm:TerminateSequence/{any}

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

622 /wsrm:TerminateSequence/@{any}

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

625 **3.6 Closing A Sequence**

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

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

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

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

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

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

650 The following exemplar defines the CloseSequence syntax:

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

652 /wsrm:CloseSequence

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

656 /wsrm:CloseSequence@Identifier

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

659 /wsrm:CloseSequence/{any}

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

662 /wsrm:CloseSequence@{any}

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

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

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

```
669 /wsrm:CloseSequenceResponse
```

670 /wsrm:CloseSequenceResponse

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

674 /wsrm:CloseSequenceResponse/{any}

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

677 /wsrm:CloseSequenceResponse@{any}

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

680 4 Faults

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

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

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

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

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

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

701 The definitions of faults use the following properties:

702 [Code] The fault code.

703 [Subcode] The fault subcode.

704 [Reason] The English language reason element.

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

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

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

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

issue10

10/9/2005

```

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

```

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

```

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

```

750 `</S11:Envelope>`

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

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

761 4.1 SequenceFault Element

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

767 The following exemplar defines its syntax:

```
768 <wsrm:SequenceFault ...>
769   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
770   ...
771 </wsrm:SequenceFault>
```

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

773 `/wsrm:SequenceFault`

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

775 `/wsrm:SequenceFault/wsrm:FaultCode`

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

778 `/wsrm:SequenceFault/{any}`

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

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

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

784 **4.2 Sequence Terminated**

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

790 Properties:

791 [Code] Sender or Receiver

792 [Subcode] wsrn:SequenceTerminated

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

794 [Detail]

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

796 **4.3 Unknown Sequence**

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

799 Properties:

800 [Code] Sender

801 [Subcode] wsrn:UnknownSequence

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

803 [Detail]

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

805 **4.4 Invalid Acknowledgement**

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

810 [Code] Sender

811 [Subcode] wsrn:InvalidAcknowledgement

812 [Reason] The SequenceAcknowledgement violates the cumulative acknowledgement
813 invariant.

814 [Detail]

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

816 **4.5 Message Number Rollover**

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

819 Properties:

820 [Code] Sender

821 [Subcode] wsrn:MessageNumberRollover

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

823 [Detail]

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

825 **4.6 Last Message Number Exceeded**

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

830 Properties:

831 [Code] Sender

832 [Subcode] wsrn:LastMessageNumberExceeded

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

835 [Detail]

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

837 **4.7 Create Sequence Refused**

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

839 Properties:

840 [Code] Sender

841 [Subcode] wsrn:CreateSequenceRefused

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

843 [Detail] empty

844 **4.8 Sequence Closed**

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

848 Properties:

849 [Code] Sender

850 [Subcode] wsrn:SequenceClosed

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

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

853 **5 Security Considerations**

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

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

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

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

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

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

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

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

920 **6 References**

921 **6.1 Normative**

922 **[KEYWORDS]**

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

925 **[SOAP]**

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

927 **[URI]**

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

930 **[XML-ns]**

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

932 **[XML-Schema1]**

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

934 **[XML-Schema2]**

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

936 **[WSSecurity]**

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

940 **[Tanenbaum]**

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

942 **[WSDL]**

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

944 **[WS-Addressing]**

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

946 **6.2 Non-Normative**

947 **[WS-Policy]**

issue10

10/9/2005

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

957 Appendix A.Schema

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

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

960 The following copy is provided for reference.

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

```

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

```

1037
1038
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078

```

    <xs:documentation>
This type is for elements whose [children] is an anyURI and can have
arbitrary attributes.
    </xs:documentation>
  </xs:annotation>
  <xs:simpleContent>
    <xs:extension base="xs:anyURI">
      <xs:anyAttribute namespace="##other" processContents="lax"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
</xs:element>
<!-- Fault Container and Codes -->
<xs:simpleType name="FaultCodes">
  <xs:restriction base="xs:QName">
    <xs:enumeration value="wsrm:UnknownSequence"/>
    <xs:enumeration value="wsrm:SequenceTerminated"/>
    <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
    <xs:enumeration value="wsrm:MessageNumberRollover"/>
    <xs:enumeration value="wsrm:CreateSequenceRefused"/>
    <xs:enumeration value="wsrm:LastMessageNumberExceeded"/>
  </xs:restriction>
</xs:simpleType>
<xs:complexType name="SequenceFaultType">
  <xs:sequence>
    <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
    <xs:any namespace="##any" processContents="lax" minOccurs="0"
maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##any" processContents="lax"/>
</xs:complexType>
<xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
<xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
<xs:element name="CreateSequenceResponse"
type="wsrm:CreateSequenceResponseType"/>
<xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
<xs:element name="CloseSequenceResponse"
type="wsrm:CloseSequenceResponseType"/>
<xs:element name="TerminateSequence"
type="wsrm:TerminateSequenceType"/>
<xs:complexType name="CreateSequenceType">
  <xs:sequence>
```

```

1079     <xs:element ref="wsrm:AcksTo"/>
1080     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1081     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1082     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1083 maxOccurs="unbounded">
1084         <xs:annotation>
1085             <xs:documentation>
1086 It is the authors intent that this extensibility be used to transfer a
1087 Security Token Reference as defined in WS-Security.
1088 </xs:documentation>
1089         </xs:annotation>
1090     </xs:any>
1091 </xs:sequence>
1092 <xs:anyAttribute namespace="##other" processContents="lax"/>
1093 </xs:complexType>
1094 <xs:complexType name="CreateSequenceResponseType">
1095     <xs:sequence>
1096         <xs:element ref="wsrm:Identifier"/>
1097         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1098         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1099         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1100 maxOccurs="unbounded"/>
1101     </xs:sequence>
1102     <xs:anyAttribute namespace="##other" processContents="lax"/>
1103 </xs:complexType>
1104 <xs:complexType name="CloseSequenceType">
1105     <xs:sequence>
1106         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1107 maxOccurs="unbounded"/>
1108     </xs:sequence>
1109     <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1110     <xs:anyAttribute namespace="##other" processContents="lax"/>
1111 </xs:complexType>
1112 <xs:complexType name="CloseSequenceResponseType">
1113     <xs:sequence>
1114         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1115 maxOccurs="unbounded"/>
1116     </xs:sequence>
1117     <xs:anyAttribute namespace="##other" processContents="lax"/>
1118 </xs:complexType>
1119 <xs:complexType name="TerminateSequenceType">
1120     <xs:sequence>

```

```

1121     <xs:element ref="wsrm:Identifier"/>
1122     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1123 maxOccurs="unbounded"/>
1124   </xs:sequence>
1125   <xs:anyAttribute namespace="##other" processContents="lax"/>
1126 </xs:complexType>
1127 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1128 <xs:complexType name="OfferType">
1129   <xs:sequence>
1130     <xs:element ref="wsrm:Identifier"/>
1131     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1132     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1133 maxOccurs="unbounded"/>
1134   </xs:sequence>
1135   <xs:anyAttribute namespace="##other" processContents="lax"/>
1136 </xs:complexType>
1137 <xs:complexType name="AcceptType">
1138   <xs:sequence>
1139     <xs:element ref="wsrm:AcksTo"/>
1140     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1141 maxOccurs="unbounded"/>
1142   </xs:sequence>
1143   <xs:anyAttribute namespace="##other" processContents="lax"/>
1144 </xs:complexType>
1145 <xs:element name="Expires">
1146   <xs:complexType>
1147     <xs:simpleContent>
1148       <xs:extension base="xs:duration">
1149         <xs:anyAttribute namespace="##other" processContents="lax"/>
1150       </xs:extension>
1151     </xs:simpleContent>
1152   </xs:complexType>
1153 </xs:element>
1154 </xs:schema>

```

1155 **Appendix B.Message Examples**

1156 B.1.Create Sequence

1157 Create Sequence

```
1158 <?xml version="1.0" encoding="UTF-8"?>
1159 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1160 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1161 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1162   <S:Header>
1163     <wsa:MessageID>
1164       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1165     </wsa:MessageID>
1166     <wsa:To>http://example.com/serviceB/123</wsa:To>
1167     <wsa:Action>http://docs.oasis-
1168 open.org/wsmr/200510/CreateSequence</wsa:Action>
1169     <wsa:ReplyTo>
1170       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1171     </wsa:ReplyTo>
1172   </S:Header>
1173   <S:Body>
1174     <wsmr:CreateSequence>
1175       <wsmr:AcksTo>
1176         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1177       </wsmr:AcksTo>
1178     </wsmr:CreateSequence>
1179   </S:Body>
1180 </S:Envelope>
```

1181 Create Sequence Response

```
1182 <?xml version="1.0" encoding="UTF-8"?>
1183 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1184 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1185 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1186   <S:Header>
1187     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1188     <wsa:RelatesTo>
1189       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1190     </wsa:RelatesTo>
1191     <wsa:Action>
1192       http://docs.oasis-open.org/wsmr/200510/CreateSequenceResponse
```

```
1193     </wsa:Action>
1194 </S:Header>
1195 <S:Body>
1196     <wsrm:CreateSequenceResponse>
1197         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1198     </wsrm:CreateSequenceResponse>
1199 </S:Body>
1200 </S:Envelope>
```

1201 B.2. Initial Transmission

1202 The following example WS-ReliableMessaging headers illustrate the message
1203 exchange in the above figure. The three messages have the following headers; the
1204 third message is identified as the last message in the sequence:

1205 Message 1

```
1206 <?xml version="1.0" encoding="UTF-8"?>
1207 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1208 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1209 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1210   <S:Header>
1211     <wsa:MessageID>
1212       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1213     </wsa:MessageID>
1214     <wsa:To>http://example.com/serviceB/123</wsa:To>
1215     <wsa:From>
1216       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1217     </wsa:From>
1218     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1219     <wsm:Sequence>
1220       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1221       <wsm:MessageNumber>1</wsm:MessageNumber>
1222     </wsm:Sequence>
1223   </S:Header>
1224   <S:Body>
1225     <!-- Some Application Data -->
1226   </S:Body>
1227 </S:Envelope>
```

1228 Message 2

```
1229 <?xml version="1.0" encoding="UTF-8"?>
1230 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1231 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1232 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1233   <S:Header>
1234     <wsa:MessageID>
1235       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1236     </wsa:MessageID>
1237     <wsa:To>http://example.com/serviceB/123</wsa:To>
```

```
1238     <wsa:From>
1239         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1240     </wsa:From>
1241     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1242     <wsrm:Sequence>
1243         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1244         <wsrm:MessageNumber>2</wsrm:MessageNumber>
1245     </wsrm:Sequence>
1246 </S:Header>
1247 <S:Body>
1248     <!-- Some Application Data -->
1249 </S:Body>
1250 </S:Envelope>
```

1251 Message 3

```
1252 <?xml version="1.0" encoding="UTF-8"?>
1253 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1254 xmlns:wsrm="http://docs.oasis-open.org/wsrn/200510/"
1255 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1256     <S:Header>
1257         <wsa:MessageID>
1258             http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1259         </wsa:MessageID>
1260         <wsa:To>http://example.com/serviceB/123</wsa:To>
1261         <wsa:From>
1262             <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1263         </wsa:From>
1264         <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1265         <wsrm:Sequence>
1266             <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1267             <wsrm:MessageNumber>3</wsrm:MessageNumber>
1268             <wsrm:LastMessage/>
1269         </wsrm:Sequence>
1270     </S:Header>
1271     <S:Body>
1272         <!-- Some Application Data -->
1273     </S:Body>
1274 </S:Envelope>
```

1275 B.3.First Acknowledgement

1276 Message number 2 has not been received by the RM Destination due to some
1277 transmission error so it responds with an acknowledgement for messages 1 and 3:

```
1278 <?xml version="1.0" encoding="UTF-8"?>
1279 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1280 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1281 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1282   <S:Header>
1283     <wsa:MessageID>
1284       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1285     </wsa:MessageID>
1286     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1287     <wsa:From>
1288       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1289     </wsa:From>
1290     <wsa:Action>
1291       http://docs.oasis-open.org/wsmr/200510/SequenceAcknowledgement
1292     </wsa:Action>
1293     <wsmr:SequenceAcknowledgement>
1294       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1295       <wsmr:AcknowledgementRange Upper="1" Lower="1"/>
1296       <wsmr:AcknowledgementRange Upper="3" Lower="3"/>
1297     </wsmr:SequenceAcknowledgement>
1298   </S:Header>
1299   <S:Body/>
1300 </S:Envelope>
```

1301 B.4.Retransmission

1302 The sending endpoint discovers that message number 2 was not received so it
1303 resends the message and requests an acknowledgement:

```
1304 <?xml version="1.0" encoding="UTF-8"?>
1305 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1306 xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
1307 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1308   <S:Header>
1309     <wsa:MessageID>
1310       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1311     </wsa:MessageID>
1312     <wsa:To>http://example.com/serviceB/123</wsa:To>
1313     <wsa:From>
1314       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1315     </wsa:From>
1316     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1317     <wsmr:Sequence>
1318       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1319       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1320     </wsmr:Sequence>
1321     <wsmr:AckRequested>
1322       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1323     </wsmr:AckRequested>
1324   </S:Header>
1325   <S:Body>
1326     <!-- Some Application Data -->
1327   </S:Body>
1328 </S:Envelope>
```

1329 B.5.Termination

1330 The RM Destination now responds with an acknowledgement for the complete
1331 sequence which can then be terminated:

```
1332 <?xml version="1.0" encoding="UTF-8"?>
1333 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1334 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1335 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1336   <S:Header>
1337     <wsa:MessageID>
1338       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1339     </wsa:MessageID>
1340     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1341     <wsa:From>
1342       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1343     </wsa:From>
1344     <wsa:Action>
1345       http://docs.oasis-open.org/wsm/200510/SequenceAcknowledgement
1346     </wsa:Action>
1347     <wsm:SequenceAcknowledgement>
1348       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1349       <wsm:AcknowledgementRange Upper="3" Lower="1"/>
1350     </wsm:SequenceAcknowledgement>
1351   </S:Header>
1352   <S:Body/>
1353 </S:Envelope>
```

1354 Terminate Sequence

```
1355 <?xml version="1.0" encoding="UTF-8"?>
1356 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1357 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1358 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1359   <S:Header>
1360     <wsa:MessageID>
1361       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1362     </wsa:MessageID>
1363     <wsa:To>http://example.com/serviceB/123</wsa:To>
1364     <wsa:Action>
1365       http://docs.oasis-open.org/wsm/200510/TerminateSequence
1366     </wsa:Action>
```

```
1367     <wsa:From>
1368     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1369     </wsa:From>
1370 </S:Header>
1371 <S:Body>
1372     <wsrm:TerminateSequence>
1373     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1374     </wsrm:TerminateSequence>
1375 </S:Body>
1376 </S:Envelope>
```

1377 Appendix C.WSDL

1378 The non-normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1379 <http://docs.oasis-open.org/wsrn/200510/wsd/wsrn.wsd>

1380 The following non-normative copy is provided for reference.

```
1381 <wSDL:definitions xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
1382 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1383 xmlns:wSA="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1384 xmlns:rm="http://docs.oasis-open.org/wsrn/200510/"
1385 xmlns:tNS="http://docs.oasis-open.org/wsrn/200510/wsd"
1386 targetNamespace="http://docs.oasis-open.org/wsrn/200510/wsd">
1387 <wSDL:types>
1388   <xs:schema>
1389     <xs:import namespace="http://docs.oasis-open.org/wsrn/200510/"
1390 schemaLocation="http://docs.oasis-open.org/wsrn/200510/wsrn.xsd"/>
1391     <xs:import
1392 namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1393 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1394   </xs:schema>
1395 </wSDL:types>
1396 <wSDL:message name="CreateSequence">
1397   <wSDL:part name="create" element="rm:CreateSequence"/>
1398 </wSDL:message>
1399 <wSDL:message name="CreateSequenceResponse">
1400   <wSDL:part name="createResponse"
1401 element="rm:CreateSequenceResponse"/>
1402 </wSDL:message>
1403 <wSDL:message name="CloseSequence">
1404   <wSDL:part name="close" element="rm:CloseSequence"/>
1405 </wSDL:message>
1406 <wSDL:message name="CloseSequenceResponse">
1407   <wSDL:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1408 </wSDL:message>
1409 <wSDL:message name="TerminateSequence">
1410   <wSDL:part name="terminate" element="rm:TerminateSequence"/>
1411 </wSDL:message>
1412 <wSDL:portType name="SequenceAbstractPortType">
1413   <wSDL:operation name="CreateSequence">
```

```
1414     <wsdl:input message="tns:CreateSequence"
1415 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CreateSequence"/>
1416     <wsdl:output message="tns:CreateSequenceResponse"
1417 wsa:Action="http://docs.oasis-
1418 open.org/wsrn/200510/CreateSequenceResponse"/>
1419     </wsdl:operation>
1420     <wsdl:operation name="CloseSequence">
1421     <wsdl:input name="tns:CloseSequence"
1422 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CloseSequence"/>
1423     <wsdl:output name="tns:CloseSequenceResponse"
1424 wsa:Action="http://docs.oasis-
1425 open.org/wsrn/200510/CloseSequenceResponse"/>
1426     </wsdl:operation>
1427     <wsdl:operation name="TerminateSequence">
1428     <wsdl:input message="tns:TerminateSequence"
1429 wsa:Action="http://docs.oasis-
1430 open.org/wsrn/200510/CreateSequenceResponse"/>
1431     </wsdl:operation>
1432     </wsdl:portType>
1433 </wsdl:definitions>
```

1434 **Appendix D.Acknowledgments**

1435 This document is based on initial contribution to OASIS WS-RX Technical Committee by the
1436 following authors: Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft,
1437 Doug Davis, IBM, Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM,
1438 Mary Ann Hondo, IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David
1439 Langworthy, Microsoft (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft,
1440 Steve Lucco, Microsoft, Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham,
1441 BEA, David Orchard, BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John
1442 Shewchuk, Microsoft, Tony Storey, IBM

1443 The following individuals have provided invaluable input into the initial contribution:

1444 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,
1445 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,
1446 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,
1447 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin
1448 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,
1449 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger
1450 Wolter, Microsoft

1451 The following individuals were members of the committee during the development of this
1452 specification:

1453 TBD

1454 **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 '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)

1455 **Appendix F.Notices**

1456 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1457 that might be claimed to pertain to the implementation or use of the technology described in this
1458 document or the extent to which any license under such rights might or might not be available;
1459 neither does it represent that it has made any effort to identify any such rights. Information on
1460 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1461 website. Copies of claims of rights made available for publication and any assurances of licenses
1462 to be made available, or the result of an attempt made to obtain a general license or permission
1463 for the use of such proprietary rights by implementors or users of this specification, can be
1464 obtained from the OASIS Executive Director.

1465 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1466 applications, or other proprietary rights which may cover technology that may be required to
1467 implement this specification. Please address the information to the OASIS Executive Director.

1468 Copyright (C) OASIS Open (2005). All Rights Reserved.

1469 This document and translations of it may be copied and furnished to others, and derivative works
1470 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1471 published and distributed, in whole or in part, without restriction of any kind, provided that the
1472 above copyright notice and this paragraph are included on all such copies and derivative works.
1473 However, this document itself may not be modified in any way, such as by removing the copyright
1474 notice or references to OASIS, except as needed for the purpose of developing OASIS
1475 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1476 Property Rights document must be followed, or as required to translate it into languages other
1477 than English.

1478 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1479 successors or assigns.

1480 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1481 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1482 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
1483 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1484 PARTICULAR PURPOSE.