



1 Web Services ReliableMessaging 2 (WS-Reliable Messaging)

3 Working Draft 07, December 13, 2005

4 **Document identifier:**

5 wsrn-1.1-spec-wd-07

6 **Location:**

7 **Editors:**

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

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

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

11 [Steve Winkler, SAP <steve.winkler@sap.com>](mailto:steve.winkler@sap.com)

12 TBD

13 **Contributors:**

14 TBD

15 **Abstract:**

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

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

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

30 **Status:**

31 This document is a work in progress and will be updated to reflect issues as they are resolved by the
32 Web Services Reliable Exchange (WS-RX) Technical Committee.

33 Table of Contents

34	1 Introduction.....	4
35	1.1 Goals and Requirements.....	4
36	1.1.1 Requirements.....	4
37	1.2 Notational Conventions.....	4
38	1.3 Namespace.....	4
39	1.4 Compliance.....	5
40	2 Reliable Messaging Model.....	6
41	2.1 Glossary.....	7
42	2.2 Protocol Preconditions.....	7
43	2.3 Protocol Invariants.....	8
44	2.4 Example Message Exchange.....	8
45	3 RM Protocol Elements.....	10
46	3.1 Sequence Creation.....	10
47	3.2 Closing A Sequence.....	13
48	3.3 Sequence Termination.....	14
49	3.4 Sequences.....	15
50	3.5 Request Acknowledgement.....	16
51	3.6 Sequence Acknowledgement.....	16
52	4 Faults.....	20
53	4.1 SequenceFault Element.....	21
54	4.2 Sequence Terminated.....	22
55	4.3 Unknown Sequence.....	22
56	4.4 Invalid Acknowledgement.....	22
57	4.5 Message Number Rollover.....	23
58	4.6 Create Sequence Refused.....	23
59	4.7 Sequence Closed.....	23
60	5 Security Considerations.....	24
61	6 References.....	26
62	6.1 Normative.....	26
63	6.2 Non-Normative.....	26
64	A. Schema.....	27
65	B. Message Examples.....	31
66	B.1 Create Sequence.....	31
67	B.2 Initial Transmission.....	31
68	B.3 First Acknowledgement.....	33
69	B.4 Retransmission.....	33
70	B.5 Termination.....	34

71	C. WSDL.....	36
72	D. Acknowledgments.....	37
73	E. Revision History.....	38
74	F. Notices.....	40

75 1 Introduction

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

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

84 1.1 Goals and Requirements

85 1.1.1 Requirements

86 1.2 Notational Conventions

87 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD
88 NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described
89 in RFC 2119 [KEYWORDS].

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

- 91 • The syntax appears as an XML instance, but values in italics indicate data types instead of values.
- 92 • Characters are appended to elements and attributes to indicate cardinality:
 - 93 ○ "?" (0 or 1)
 - 94 ○ "*" (0 or more)
 - 95 ○ "+" (1 or more)
- 96 • The character "|" is used to indicate a choice between alternatives.
- 97 • The characters "[" and "]" are used to indicate that contained items are to be treated as a group
98 with respect to cardinality or choice.
- 99 • An ellipsis (i.e. "...") indicates a point of extensibility that allows *child or attribute content not
100 specified in this document. Additional children elements and/or attributes MAY be added at the
101 indicated extension points but they other child, or attribute, content. Additional children elements
102 and/or attributes MAY be added at the indicated extension points but* MUST NOT contradict the
103 semantics of the parent and/or owner, respectively. If an extension is not recognized it SHOULD
104 be ignored.
- 105 • XML namespace prefixes (See Section [Namespace](#)) are used to indicate the namespace of the
106 element being defined.

107 1.3 Namespace

108 The XML namespace [XML-ns] URI that MUST be used by implementations of this specification is:

109 `http://docs.oasis-open.org/ws-rx/wsrn/200510`

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

112 The following namespaces are used in this document:

113 *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/ws-rx/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

114 The normative schema for WS-ReliableMessaging can be found at:

115 <http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1.xsd>

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

117 If an action IRI is used, and one is not already defined per the rules of the WS-Addressing specification
118 [WS-Addressing], then the action IRI MUST consist of the reliable messaging namespace URI
119 concatenated with a '/', followed by the message element name. For example:

120 `http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement`

121 **1.4 Compliance**

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

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

128 2 Reliable Messaging Model

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

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

138 In addition, the protocol allows the RM Source and RM Destination to provide their respective Application
139 Source and Application Destination a guarantee that a message that is sent by an Application Source will
140 be delivered to the Application Destination.

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

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

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

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

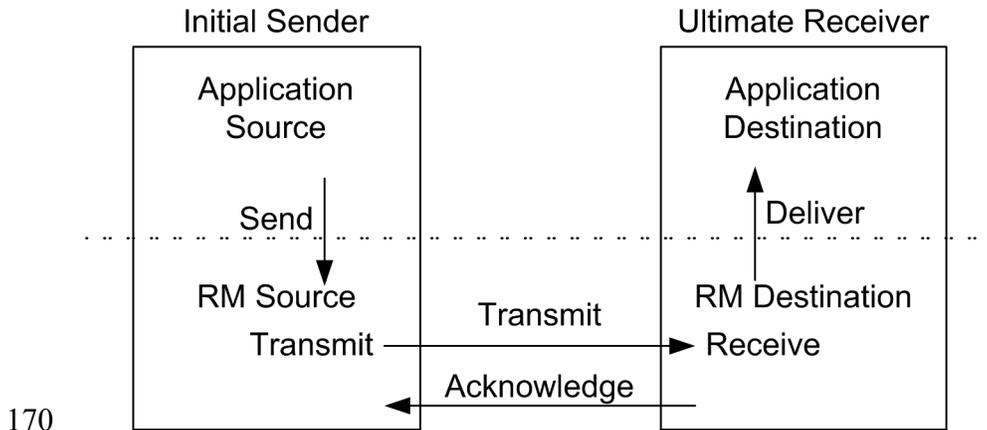
152 **AtMostOnce** Every message will be delivered at most once without duplication or an error will
153 be raised on at least one endpoint. It is possible that some messages in a sequence may not be
154 delivered.

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

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

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

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



171 Figure 1: Reliable Messaging Model

172 2.1 Glossary

173 The following definitions are used throughout this specification:

174 **Acknowledgement:** The communication from the RM Destination to the RM Source indicating the
175 successful receipt of a message.

176 **Application Destination:** The endpoint to which a message is Delivered.

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

178 **Deliver:** The act of transferring a message from the RM Destination to the Application Destination. The
179 reliability guarantee is fulfilled at this point.

180 **Delivery Assurance:** The guarantee that the messaging infrastructure provides on the delivery of a
181 message.

182 **Endpoint:** As defined in the WS-Addressing specification [WS-Addressing]; a Web service endpoint is a
183 (referenceable) entity, processor, or resource to which Web service messages can be addressed.
184 Endpoint references convey the information needed to address a Web service endpoint.

185 **Receive:** The act of reading a message from a network connection and qualifying it as relevant to RM
186 Destination functions.

187 **RM Destination:** For any one reliable sent message the endpoint that receives the message.

188 **Send:** The act of submitting a message to the RM Source for reliable delivery. The reliability guarantee
189 begins at this point.

190 **Transmit:** The act of writing a message to a network connection.

191 2.2 Protocol Preconditions

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

- 194 • For any single message exchange the RM Source **MUST** have an endpoint reference that uniquely
195 identifies the RM Destination endpoint.

- 196 • The RM Source MUST have knowledge of the destination's policies, if any, and the RM Source
 197 MUST be capable of formulating messages that adhere to this policy.
- 198 If a secure exchange of messages is required, then the RM Source and RM Destination MUST have a
 199 security context.

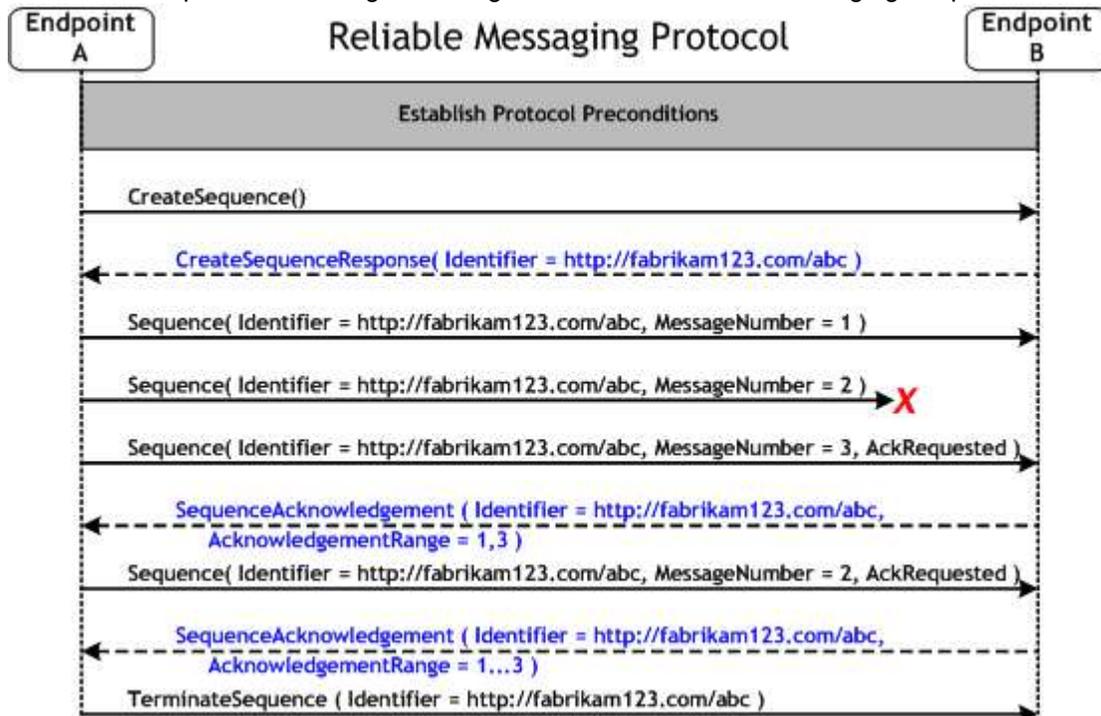
200 2.3 Protocol Invariants

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

- 202 • The RM Source MUST assign each reliable message a sequence number (defined below) beginning
 203 at 1 and increasing by exactly 1 for each subsequent reliable message.
- 204 • Every acknowledgement issued by the RM Destination MUST include within an acknowledgement
 205 range or ranges the sequence number of every message successfully received by the RM
 206 Destination and MUST exclude sequence numbers of any messages not yet received.

207 2.4 Example Message Exchange

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



209 Figure 2: The WS-ReliableMessaging Protocol

- 210 1. The protocol preconditions are established. These include policy exchange, endpoint resolution,
 211 establishing trust.
- 212 2. The RM Source requests creation of a new Sequence.
- 213 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 214 4. The RM Source begins sending messages beginning with MessageNumber 1. In the figure [above](#),
 215 the RM Source sends 3 messages.
- 216 5. Since the 3rd message is the last in this exchange, the RM Source includes a
 217 <wsrm:AckRequested> Header.

- 218 6. The 2nd message is lost in transit.
- 219 7. The RM Destination acknowledges receipt of message numbers 1 and 3 as a result of receiving the
220 RM Source's `<wsrm:AckRequested>` Header.
- 221 8. The RM Source retransmits the 2nd message. This is a new message on the underlying transport,
222 but **since** it has the same sequence identifier and message number so the RM Destination can
223 recognize it as equivalent to the earlier message, in case both are received.
- 224 9. The RM Source includes an `<wsrm:AckRequested>` element so the RM Destination will expedite
225 an acknowledgement.
- 226 10. The RM Destination receives the second transmission of the message with MessageNumber 2 and
227 acknowledges receipt of message numbers 1, 2, and 3.
- 228 11. The RM Source receives this acknowledgement and sends a TerminateSequence message to the
229 RM Destination indicating that the sequence is completed and reclaims any resources associated
230 with the Sequence.
- 231 12. The RM Destination receives the TerminateSequence message indicating that the RM Source will
232 not be sending any more messages, and reclaims any resources associated with the Sequence.
- 233 The RM Source will expect to receive acknowledgements from the RM Destination during the course of a
234 message exchange at occasions described in Section 3 below. Should the acknowledgement not be
235 received **in a timely fashion**~~timely~~, the RM Source MUST re-transmit the request since either the request
236 or the associated acknowledgement may have been lost. Since the nature and dynamic characteristics of
237 the underlying transport and potential intermediaries are unknown in the general case, the timing of re-
238 transmissions cannot be specified. Additionally, over-aggressive re-transmissions have been
239 demonstrated to cause transport or intermediary flooding which are counterproductive to the intention of
240 providing a reliable message exchange. Consequently, implementers are encouraged to utilize adaptive
241 mechanisms that dynamically adjust re-transmission time and the back-off intervals that are appropriate to
242 the nature of the transports and intermediaries envisioned. For the case of TCP/IP transports, a
243 mechanism similar to that described as RTTM in RFC 1323 [RTTM] should be considered.
- 244 Now that the basic model has been outlined, the details of the elements used in this protocol are now
245 provided in Section 3.

246 **3 RM Protocol Elements**

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

251 **3.1 Sequence Creation**

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

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

```
258 <wsrm:CreateSequence ...>  
259   <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>  
260   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
261   <wsrm:Offer ...>  
262     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
263     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
264     ...  
265   </wsrm:Offer> ?  
266   ...  
267 </wsrm:CreateSequence>
```

268 `/wsrm:CreateSequence`

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

273 `/wsrm:CreateSequence/wsrm:AcksTo`

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

277 Implementations MUST NOT use an endpoint reference in the AcksTo element that would prevent the
278 sending of Sequence Acknowledgements back to the RM Source. For example, using the WS-Addressing
279 "none" IRI would make it impossible for the RM Destination to ever send Sequence Acknowledgements.

280 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

288 /wsmr:CreateSequence/wsmr:Offer

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

291 /wsmr:CreateSequence/wsmr:Offer/wsmr:Identifier

292 This REQUIRED element MUST contain an absolute URI conformant with RFC3986 that uniquely
 293 identifies the offered Sequence.

294 /wsmr:CreateSequence/wsmr:Offer/wsmr:Identifier/@{any}

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

297 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

298 This element, if present, of type *xs:duration* specifies the duration for the Sequence. A value of 'PT0S'
 299 indicates that the Sequence will never expire. Absence of the element indicates an implied value of
 300 'PT0S'.

301 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires/@{any}

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

304 /wsmr:CreateSequence/wsmr:Offer/{any}

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

307 /wsmr:CreateSequence/wsmr:Offer/@{any}

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

310 /wsmr:CreateSequence/{any}

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

313 /wsmr:CreateSequence/@{any}

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

316 A <wsmr:CreateSequenceResponse> is sent in the body of a response message by an RM
 317 Destination in response to receipt of a <wsmr:CreateSequence> request message. It carries the
 318 <wsmr:Identifier> of the created Sequence and indicates that the RM Source may begin sending
 319 messages in the context of the identified Sequence.

320 The following exemplar defines the <wsmr:CreateSequenceResponse> syntax:

```

321 <wsmr:CreateSequenceResponse ...>
322   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>
323   <wsmr:Expires> xs:duration </wsmr:Expires> ?
324   <wsmr:Accept ...>
325     <wsmr:AcksTo ...> wsa:EndpointReferenceType </wsmr:AcksTo>
326     ...
327   </wsmr:Accept> ?
328   ...

```

329

`</wsrm:CreateSequenceResponse>`

330 `/wsrm:CreateSequenceResponse`

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

334 `/wsrm:CreateSequenceResponse/wsrm:Identifier`

335 This REQUIRED element MUST contain an absolute URI conformant with RFC3986 of the Sequence that
336 has been created by the RM Destination.

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

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

340 `/wsrm:CreateSequenceResponse/wsrm:Expires`

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

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

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

348 `/wsrm:CreateSequenceResponse/wsrm:Accept`

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

351 **Note:** If a `<wsrm:CreateSequenceResponse>` is returned without a child `<wsrm:Accept>` in response
352 to a `<wsrm:CreateSequence>` that did contain a child `<wsrm:Offer>`, then the RM Source MAY
353 immediately reclaim any resources associated with the unused offered Sequence.

354 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

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

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

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

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

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

364 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

370 **3.2 Closing A Sequence**

371 There may be times during the use of an RM Sequence that the RM Source or RM Destination will wish to
372 discontinue using a Sequence. Simply terminating the Sequence discards the state managed by the RM
373 Destination, leaving the RM Source unaware of the final ranges of messages that were successfully
374 delivered to the RM Destination. To ensure that the Sequence ends with a known final state both the RM
375 Source and RM Destination may choose to 'close' the Sequence before terminating it.

376 If the RM Source wishes to close the Sequence then it sends a `<wsrm:CloseSequence>` element, in the
377 body of a message, to the RM Destination. This message indicates that RM Destination MUST NOT
378 receive any new messages for the specified sequence, other than those already received at the time the
379 `<wsrm:CloseSequence>` element is interpreted by the RMD. Upon receipt of this message the RM
380 Destination MUST include a SequenceAcknowledgement header block in the CloseSequenceResponse
381 message. Note, this SequenceAcknowledgement MUST include the `<wsrm:Final>` element.

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

386 In the case where the RM Destination wishes to discontinue use of a sequence it may 'close' the
387 sequence itself. Please see `<wsrm:Final>` above and the SequenceClosed fault below. Note, the
388 SequenceClosed Fault SHOULD be used in place of the SequenceTerminated Fault, whenever possible,
389 to allow the RM Source to still receive Acknowledgements.

390 The following exemplar defines the CloseSequence syntax:

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

392 `/wsrm:CloseSequence`

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

396 `/wsrm:CloseSequence@Identifier`

397 This REQUIRED attribute contains an absolute URI conformant with RFC3986 that uniquely identifies the
398 sequence.

399 `/wsrm:CloseSequence/{any}`

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

402 `/wsrm:CloseSequence@{any}`

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

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

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

409

```
</wsrm:CloseSequenceResponse
```

410 /wsrm:CloseSequenceResponse

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

414 /wsrm:CloseSequenceResponse/{any}

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

417 /wsrm:CloseSequenceResponse@{any}

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

420 3.3 Sequence Termination

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

428 The following exemplar defines the TerminateSequence syntax:

```
429 <wsrm:TerminateSequence ...>  
430   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
431   ...  
432 </wsrm:TerminateSequence>
```

433 /wsrm:TerminateSequence

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

438 /wsrm:TerminateSequence/wsrm:Identifier

439 This REQUIRED element MUST contain an absolute URI conformant with RFC3986 of the Sequence that
440 is being terminated.

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

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

444 /wsrm:TerminateSequence/{any}

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

447 /wsrm:TerminateSequence/@{any}

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

450 3.4 Sequences

451 The RM protocol uses a `<wsrm:Sequence>` header block to track and manage the reliable delivery of
452 messages. Messages for which the delivery assurance applies MUST contain a `<wsrm:Sequence>`
453 header block. Each Sequence MUST have a unique `<wsrm:Identifier>` element and each message
454 within a Sequence MUST have a `<wsrm:MessageNumber>` element that increments by 1 from an initial
455 value of 1. These values are contained within a `<wsrm:Sequence>` header block accompanying each
456 message being delivered in the context of a Sequence.

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

458 A following exemplar defines its syntax:

```
459 <wsrm:Sequence ...>  
460   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
461   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>  
462   ...  
463 </wsrm:Sequence>
```

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

465 `/wsrm:Sequence`

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

470 `/wsrm:Sequence/wsrm:Identifier`

471 This REQUIRED element MUST contain an absolute URI conformant with RFC3986 that uniquely
472 identifies the Sequence.

473 `/wsrm:Sequence/wsrm:Identifier/@{any}`

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

476 `/wsrm:Sequence/wsrm:MessageNumber`

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

482 `/wsrm:Sequence/{any}`

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

485 `/wsrm:Sequence/@{any}`

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

488 The following example illustrates a Sequence header block.

```
489 <wsrm:Sequence>  
490   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
491   <wsrm:MessageNumber>10</wsrm:MessageNumber>  
492 </wsrm:Sequence>
```

493 3.5 Request Acknowledgement

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

496 The RM Source may request an acknowledgement message from the RM Destination at any time by by
497 including an `<wsrm:AckRequested>` header block in the message. An RM Destination that receives a
498 message that contains an `<wsrm:AckRequested>` header block MUST respond with a message
499 containing a `<wsrm:SequenceAcknowledgement>` header block. If a non-mustUnderstand fault occurs
500 when processing an RM Header that was piggy-backed on another message, a fault MUST be generated,
501 but the processing of the original message MUST NOT be affectedAt any time, the RM Source may
502 request an acknowledgement message from the RM Destination using an `<wsrm:AckRequested>`
503 header block.

504 ~~The RM Source requests this acknowledgement by including an `<wsrm:AckRequested>` header block~~
505 ~~in the message. An RM Destination that receives a message that contains an `<wsrm:AckRequested>`~~
506 ~~header block MUST respond with a message containing a `<wsrm:SequenceAcknowledgement>`~~
507 ~~header block. If a non-mustUnderstand fault occurs when processing an RM Header that was piggy-~~
508 ~~backed on another message, a fault MUST be generated, but the processing of the original message~~
509 ~~MUST NOT be affected.~~

510 The following exemplar defines its syntax:

```
511 <wsrm:AckRequested ...>  
512   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
  
513   ...  
514 </wsrm:AckRequested>
```

515 /wsrm:AckRequested

516 This element requests an acknowledgement for the identified sequence.

517 /wsrm:AckRequested/wsrm:Identifier

518 This REQUIRED element MUST contain an absolute URI, conformant with RFC3986, that uniquely
519 identifies the Sequence to which the request applies.

520 /wsrm:AckRequested/wsrm:Identifier/@{any}

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

523 /wsrm:AckRequested/{any}

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

526 /wsrm:AckRequested/@{any}

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

529 3.6 Sequence Acknowledgement

530 The RM Destination informs the RM Source of successful message receipt using a
531 `<wsrm:SequenceAcknowledgement>` header block. The `<wsrm:SequenceAcknowledgement>`
532 header block MAY be transmitted independently or included on return messages. The RM Destination
533 MAY send a `<wsrm:SequenceAcknowledgement>` header block at any point during which the
534 sequence is valid. The timing of acknowledgements can be advertised using policy and
535 acknowledgements can be explicitly requested using the `<wsrm:AckRequested>` directive (see Section
536 [Request Acknowledgement](#)). If a non-mustUnderstand fault occurs when processing an RM Header that
537 was piggy-backed on another message, a fault MUST be generated, but the processing of the original
538 message MUST NOT be affected.

539 The following exemplar defines its syntax:

```
540 <wsrm:SequenceAcknowledgement ...>
541   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
542   [ [ <wsrm:AcknowledgementRange ...
543       Upper="xs:unsignedLong"
544       Lower="xs:unsignedLong"/> +
545
546       | <wsrm:None/> ]
547       <wsrm:Final/> ?
548       | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> + ]
549   ...
550 </wsrm:SequenceAcknowledgement>
```

551 The following describes the content model of the `<wsrm:SequenceAcknowledgement>` header block.

552 `/wsrm:SequenceAcknowledgement`

553 This element contains the Sequence acknowledgement information.

554 `/wsrm:SequenceAcknowledgement/wsrm:Identifier`

555 This REQUIRED element MUST contain an absolute URI conformant with RFC3986 that uniquely
556 identifies the Sequence. A message MUST NOT contain multiple `<SequenceAcknowledgement>` header
557 blocks that share the same value for `<Identifier>`.

558 `/wsrm:SequenceAcknowledgement/wsrm:Identifier/@{any}`

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

561 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange`

562 This OPTIONAL element, if present, can occur 1 or more times. It contains a range of message Sequence
563 MessageNumbers successfully received by the RM Destination. The ranges SHOULD NOT overlap. This
564 element MUST NOT be present if a sibling `<wsrm:Nack>` or `<wsrm:None>` elements are also present as
565 a child of `<wsrm:SequenceAcknowledgement>`.

566 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper`

567 This REQUIRED attribute contains an `xs:unsignedLong` representing the `<wsrm:MessageNumber>` of
568 the highest contiguous message in a Sequence range received by the RM Destination.

569 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower`

570 This REQUIRED attribute contains an `xs:unsignedLong` representing the `<wsrm:MessageNumber>` of
571 the lowest contiguous message in a Sequence range received by the RM Destination.

572 /wsmr:SequenceAcknowledgement/wsmr:AcknowledgementRange/@{any}

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

575 /wsmr:SequenceAcknowledgement/wsmr:Final

576 This OPTIONAL element, if present, indicates that the RM Destination is not receiving new messages for
577 the specified Sequence. The RM Source can be assured that the ranges of messages acknowledged by
578 this SequenceAcknowledgement header block will not change in the future. This element MUST be
579 present when the Sequence is no longer receiving new message for the specified sequence. Note: this
580 element MUST NOT be used when sending a Nack, it can only be used when sending
581 AcknowledgementRanges or <wsmr:None>.

582 /wsmr:SequenceAcknowledgement/wsmr:Nack

583 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the
584 <wsmr:MessageNumber> of an unreceived message in a Sequence. This element permits the gap
585 analysis of the <wsmr:AcknowledgementRange> elements to be performed at the RM Destination
586 rather than at the RM Source, which may yield performance benefits in certain environments. The
587 <wsmr:Nack> element MUST NOT be present if a sibling <wsmr:AcknowledgementRange> or
588 <wsmr:None> elements are also present as a child of <wsmr:SequenceAcknowledgement>. Upon the
589 receipt of a Nack, an RM Source SHOULD retransmit the message identified by the Nack. The RM
590 Destination MUST NOT issue a <wsmr:SequenceAcknowledgement> containing a <wsmr:Nack> for
591 a message that it has previously acknowledged within a <wsmr:AcknowledgementRange>. The RM
592 Source SHOULD ignore a <wsmr:SequenceAcknowledgement> containing a <wsmr:Nack> for a
593 message that has previously been acknowledged within a <wsmr:AcknowledgementRange>.

594 /wsmr:SequenceAcknowledgement/wsmr:None

595 This OPTIONAL element, if present, MUST be used when the RM Destination has not received any
596 messages for the specified sequence. The <wsmr:None> element MUST NOT be present if a sibling
597 <wsmr:AcknowledgementRange> or <wsmr:Nack> elements are also present as a child of the
598 <wsmr:SequenceAcknowledgement>.

599 /wsmr:SequenceAcknowledgement/{any}

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

602 /wsmr:SequenceAcknowledgement/@{any}

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

605 The following examples illustrate <wsmr:SequenceAcknowledgement> elements:

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

```
607 <wsmr:SequenceAcknowledgement>  
608     <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>  
609     <wsmr:AcknowledgementRange Upper="10" Lower="1"/>  
610 </wsmr:SequenceAcknowledgement>
```

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

```
613 <wsmr:SequenceAcknowledgement>  
614     <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>
```

```
615     <wsrm:AcknowledgementRange Upper="2" Lower="1"/>
616     <wsrm:AcknowledgementRange Upper="6" Lower="4"/>
617     <wsrm:AcknowledgementRange Upper="10" Lower="8"/>
618 </wsrm:SequenceAcknowledgement>
```

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

```
620 <wsrm:SequenceAcknowledgement>
621     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
622     <wsrm:Nack>3</wsrm:Nack>
623 </wsrm:SequenceAcknowledgement>
```

624 4 Faults

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

628 Sequence creation uses a CreateSequence, CreateSequenceResponse request-response pattern. Faults
629 for this operation are treated as defined in WS-Addressing. CreateSequenceRefused is a possible fault
630 reply for this operation. UnknownSequence is a fault generated by endpoints when messages carrying RM
631 header blocks targeted at unrecognized sequences are detected; these faults are also treated as defined
632 in WS-Addressing. All other faults in this section relate to the processing of RM header blocks targeted at
633 known sequences and are collectively referred to as sequence faults. Sequence faults SHOULD be sent
634 to the same [destination] as `<wsrm:SequenceAcknowledgement>` messages. These faults are
635 correlated using the Sequence identifier carried in the detail.

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

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

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

642 The definitions of faults use the following properties:

643 [Code] The fault code.

644 [Subcode] The fault subcode.

645 [Reason] The English language reason element.

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

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

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

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

```
650 <S:Envelope>  
651   <S:Header>  
652     <wsa:Action>  
653       http://schemas.xmlsoap.org/ws/2004/08/addressing/fault  
654     </wsa:Action>  
655     <!-- Headers elided for clarity. -->  
656   </S:Header>  
657   <S:Body>  
658     <S:Fault>  
659       <S:Code>  
660         <S:Value> [Code] </S:Value>  
661         <S:Subcode>  
662           <S:Value> [Subcode] </S:Value>  
663         </S:Subcode>  
664       </S:Code>  
665       <S:Reason>
```

```

666     <S:Text xml:lang="en"> [Reason] </S:Text>
667   </S:Reason>
668   <S:Detail>
669     [Detail]
670     ...
671   </S:Detail>
672 </S:Fault>
673 </S:Body>
674 </S:Envelope>

```

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

```

677 <S11:Envelope>
678   <S11:Header>
679     <wsrm:SequenceFault>
680       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
681       ...
682     </wsrm:SequenceFault>
683     <!-- Headers elided for clarity. -->
684   </S11:Header>
685   <S11:Body>
686     <S11:Fault>
687       <faultcode> [Code] </faultcode>
688       <faultstring> [Reason] </faultstring>
689     </S11:Fault>
690   </S11:Body>
691 </S11:Envelope>

```

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

```

694 <S11:Envelope>
695   <S11:Body>
696     <S11:Fault>
697       <faultcode> [Subcode] </faultcode>
698       <faultstring xml:lang="en"> [Reason] </faultstring>
699     </S11:Fault>
700   </S11:Body>
701 </S11:Envelope>

```

702 4.1 SequenceFault Element

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

707 The following exemplar defines its syntax:

```

708 <wsrm:SequenceFault ...>
709   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
710   ...
711 </wsrm:SequenceFault>

```

712 The following describes the content model of the SequenceFault element.

713 /wsrm:SequenceFault

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

715 /wsm:SequenceFault/wsm:FaultCode

716 This element, if present, MUST contain a qualified name from the set of fault [Subcodes] defined below.

717 /wsm:SequenceFault/{any}

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

720 /wsm:SequenceFault/@{any}

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

723 4.2 Sequence Terminated

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

728 Properties:

729 [Code] Sender or Receiver

730 [Subcode] wsm:SequenceTerminated

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

732 [Detail]

```
733 <wsm:Identifier ...> xs:anyURI </wsm:Identifier>
```

734 4.3 Unknown Sequence

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

737 Properties:

738 [Code] Sender

739 [Subcode] wsm:UnknownSequence

740 [Reason] The value of wsm:Identifier is not a known Sequence identifier.

741 [Detail]

```
742 <wsm:Identifier ...> xs:anyURI </wsm:Identifier>
```

743 4.4 Invalid Acknowledgement

744 This fault is sent by the RM Source in response to a <wsm:SequenceAcknowledgement> that violates
745 the cumulative acknowledgement invariant. An example of such a violation would be a
746 SequenceAcknowledgement covering messages that have not been sent.

747 [Code] Sender

748 [Subcode] wsm:InvalidAcknowledgement

749 [Reason] The SequenceAcknowledgement violates the cumulative acknowledgement invariant.

750 [Detail]

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

752 **4.5 Message Number Rollover**

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

754 Properties:

755 [Code] Sender

756 [Subcode] wsrm:MessageNumberRollover

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

758 [Detail]

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

760 **4.6 Create Sequence Refused**

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

762 Properties:

763 [Code] Sender

764 [Subcode] wsrm:CreateSequenceRefused

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

766 [Detail]

```
767 xs:any
```

768 **4.7 Sequence Closed**

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

772 Properties:

773 [Code] Sender

774 [Subcode] wsrm:SequenceClosed

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

776 [Detail]

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

778 5 Security Considerations

779 It is strongly recommended that the communication between services be secured using the mechanisms
780 described in WS-Security. In order to properly secure messages, the body and all relevant headers need
781 to be included in the signature. Specifically, the <wsrm:Sequence> header needs to be signed with the
782 body in order to "bind" the two together. In order to properly secure messages, the body and all relevant
783 headers need to be included in the signature. Specifically, the <wsrm:Sequence> header needs to be
784 signed with the body in order to "bind" the two together. The <wsrm:SequenceAcknowledgement>
785 header may be signed independently because a reply independent of the message is not a security
786 concern.

787 Because Sequences are expected to exchange a number of messages, it is recommended that a security
788 context be established using the mechanisms described in WS-Trust and WS-SecureConversation
789 [SecureConversation]. If a Sequence is bound to a specific destination, then the security context needs to
790 be established or shared with the destination servicing the Sequence. While the context can be
791 established at any time, it is critical that the messages establishing the Sequence be secured even if they
792 precede security context establishment. However, it is recommended that the security context be
793 established first. Security contexts are independent of reliable messaging Sequences. Consequently,
794 security contexts can come and go independent of the lifetime of the Sequence. If a Sequence is bound to
795 a specific destination, then the security context needs to be established or shared with the destination
796 servicing the Sequence. While the context can be established at any time, it is critical that the messages
797 establishing the Sequence be secured even if they precede security context establishment. However, it is
798 recommended that the security context be established first. Security contexts are independent of reliable
799 messaging Sequences. Consequently, security contexts can come and go independent of the lifetime of
800 the Sequence. In fact, it is recommended that the lifetime of a security context be less than the lifetime of
801 the Sequence unless the Sequence is very short-lived.

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

- 808 • Closing and re-establishing a security context
- 809 • Exchanging new secrets between the parties
- 810 • Using a derived key sequence and switch "generations"
- 811 • Attaching a nonce to each message and using it in a derived key function with the shared secret

812 The security context may be re-established using the mechanisms described in WS-Trust and WS-
813 SecureConversation. Similarly, secrets can be exchanged using the mechanisms described in WS-
814 Trust. Note, however, that the current shared secret should not be used to encrypt the new shared secret.
815 Similarly, secrets can be exchanged using the mechanisms described in WS-Trust. Note, however, that
816 the current shared secret should not be used to encrypt the new shared secret. Derived keys, the preferred
817 solution from this list, can be specified using the mechanisms described in WS-SecureConversation.

818 There is a core tension between security and reliable messaging that can be problematic if not considered
819 in implementations. That is, one aspect of security is to prevent message replay and the core tenet of
820 reliable messaging is to replay messages until they are acknowledged. Consequently, if the security sub-
821 system processes a message but a failure occurs before the reliable messaging sub-system records the
822 message (or the message is considered "processed"), then it is possible (and likely) that the security sub-

823 ~~system will treat subsequent copies as replays and discard them. At the same time, the reliable~~
824 ~~messaging sub-system will likely continue to expect and even solicit the missing message(s). That is, one~~
825 ~~aspect of security is to prevent message replay and the core tenet of reliable messaging is to replay~~
826 ~~messages until they are acknowledged. Consequently, if the security sub-system processes a message~~
827 ~~but a failure occurs before the reliable messaging sub-system records the message (or the message is~~
828 ~~considered "processed"), then it is possible (and likely) that the security sub-system will treat subsequent~~
829 ~~copies as replays and discard them. At the same time, the reliable messaging sub-system will likely~~
830 ~~continue to expect and even solicit the missing message(s).~~ Care should be taken to avoid and prevent
831 this rare condition.

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

- 834 • **Message alteration** – Alteration is prevented by including signatures of the message information
835 using WS-Security.
- 836 • **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-Security.
- 837 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by comparing
838 secured policies – see WS-Policy and WS-SecurityPolicy).
- 839 • **Authentication** – Authentication is established using the mechanisms described in WS-Security
840 and WS-Trust. Each message is authenticated using the mechanisms described in WS-Security.
- 841 • **Accountability** – Accountability is a function of the type of and string of the key and algorithms
842 being used. ~~In many cases, a strong symmetric key provides sufficient accountability. In many~~
843 ~~cases, a strong symmetric key provides sufficient accountability.~~ However, in some environments,
844 strong PKI signatures are required.
- 845 • **Availability** – All reliable messaging services are subject to a variety of availability attacks. ~~Replay~~
846 ~~detection is a common attack and it is recommended that this be addressed by the mechanisms~~
847 ~~described in WS-Security (Note that because of legitimate message replays, detection should~~
848 ~~include a differentiator besides message id such as a timestamp). Other attacks, such as network-~~
849 ~~level denial of service attacks are harder to avoid and are outside the scope of this specification.~~
850 Replay detection is a common attack and it is recommended that this be addressed by the
851 mechanisms described in WS-Security. (Note that because of legitimate message replays, detection
852 should include a differentiator besides message id such as a timestamp). Other attacks, such as
853 network-level denial of service attacks are harder to avoid and are outside the scope of this
854 specification. That said, care should be taken to ensure that minimal state is saved prior to any
855 authenticating sequences.

856 **6 References**

857 **6.1 Normative**

858 **[KEYWORDS]**

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

861 **[SOAP]**

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

863 **[URI]**

864 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#)," RFC 3986,
865 MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.

866 **[XML-ns]**

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

868 **[XML-Schema1]**

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

870 **[XML-Schema2]**

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

872 **[WSSecurity]**

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

875 **[RTTM]**

876 V. Jacobson, R. Braden, D. Borman, "[TCP Extensions for High Performance](#)", RFC 1323, May 1992.

877 **[WSDL]**

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

879 **[WS-Addressing]**

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

881 **6.2 Non-Normative**

882 **[WS-Policy]**

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

884 **[WS-PolicyAttachment]**

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

886 **[SecurityPolicy]**

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

888 **[SecureConversation]**

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

890 A. Schema

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

892 <http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1.xsd>

893 The following copy is provided for reference.

```
894 <xs:schema targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"
895 xmlns:xs="http://www.w3.org/2001/XMLSchema"
896 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
897 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
898 elementFormDefault="qualified" attributeFormDefault="unqualified">
899   <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
900   schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
901   <!-- Protocol Elements -->
902   <xs:complexType name="SequenceType">
903     <xs:sequence>
904       <xs:element ref="wsrm:Identifier"/>
905       <xs:element name="MessageNumber" type="xs:unsignedLong"/>
906       <xs:any namespace="##other" processContents="lax" minOccurs="0"
907 maxOccurs="unbounded"/>
908     </xs:sequence>
909     <xs:anyAttribute namespace="##other" processContents="lax"/>
910   </xs:complexType>
911   <xs:element name="Sequence" type="wsrm:SequenceType"/>
912   <xs:element name="SequenceAcknowledgement">
913     <xs:complexType>
914       <xs:sequence>
915         <xs:element ref="wsrm:Identifier"/>
916         <xs:choice>
917           <s:sequence>
918             <xs:choice>
919               <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
920                 <xs:complexType>
921                   <xs:sequence/>
922                   <xs:attribute name="Upper" type="xs:unsignedLong"
923 use="required"/>
924                   <xs:attribute name="Lower" type="xs:unsignedLong"
925 use="required"/>
926                   <xs:anyAttribute namespace="##other" processContents="lax"/>
927                 </xs:complexType>
928               </xs:element>
929               <xs:element name="None" minOccurs="0">
930                 <xs:complexType>
931                   <xs:sequence/>
932                 </xs:complexType>
933               </xs:element>
934             </xs:choice>
935             <xs:element name="Final" minOccurs="0">
936               <xs:complexType>
937                 <xs:sequence/>
938               </xs:complexType>
939             </xs:element>
940           </s:sequence>
941           <xs:element name="Nack" type="xs:unsignedLong"
942 maxOccurs="unbounded"/>
943         </xs:choice>
944     </xs:complexType>
```

```

945         <xs:any namespace="##other" processContents="lax" minOccurs="0"
946 maxOccurs="unbounded"/>
947     </xs:sequence>
948     <xs:anyAttribute namespace="##other" processContents="lax"/>
949 </xs:complexType>
950 </xs:element>
951 <xs:complexType name="AckRequestedType">
952     <xs:sequence>
953         <xs:element ref="wsrm:Identifier"/>
954
955         <xs:any namespace="##other" processContents="lax" minOccurs="0"
956 maxOccurs="unbounded"/>
957     </xs:sequence>
958     <xs:anyAttribute namespace="##other" processContents="lax"/>
959 </xs:complexType>
960 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
961 <xs:element name="Identifier">
962     <xs:complexType>
963         <xs:annotation>
964             <xs:documentation>
965 This type is for elements whose [children] is an anyURI and can have arbitrary
966 attributes.
967             </xs:documentation>
968         </xs:annotation>
969         <xs:simpleContent>
970             <xs:extension base="xs:anyURI">
971                 <xs:anyAttribute namespace="##other" processContents="lax"/>
972             </xs:extension>
973         </xs:simpleContent>
974     </xs:complexType>
975 </xs:element>
976 <!-- Fault Container and Codes -->
977 <xs:simpleType name="FaultCodes">
978     <xs:restriction base="xs:QName">
979         <xs:enumeration value="wsrm:UnknownSequence"/>
980         <xs:enumeration value="wsrm:SequenceTerminated"/>
981         <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
982         <xs:enumeration value="wsrm:MessageNumberRollover"/>
983         <xs:enumeration value="wsrm:CreateSequenceRefused"/>
984     </xs:restriction>
985 </xs:simpleType>
986 <xs:complexType name="SequenceFaultType">
987     <xs:sequence>
988         <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
989         <xs:any namespace="##any" processContents="lax" minOccurs="0"
990 maxOccurs="unbounded"/>
991     </xs:sequence>
992     <xs:anyAttribute namespace="##any" processContents="lax"/>
993 </xs:complexType>
994 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
995 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
996 <xs:element name="CreateSequenceResponse"
997 type="wsrm:CreateSequenceResponseType"/>
998 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
999 <xs:element name="CloseSequenceResponse"
1000 type="wsrm:CloseSequenceResponseType"/>
1001 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1002 <xs:complexType name="CreateSequenceType">
1003     <xs:sequence>
1004         <xs:element ref="wsrm:AcksTo"/>
1005         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1006         <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>

```

```

1007     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1008 maxOccurs="unbounded">
1009         <xs:annotation>
1010             <xs:documentation>
1011 It is the authors intent that this extensibility be used to transfer a
1012 Security Token Reference as defined in WS-Security.
1013 </xs:documentation>
1014         </xs:annotation>
1015     </xs:any>
1016 </xs:sequence>
1017 <xs:anyAttribute namespace="##other" processContents="lax"/>
1018 </xs:complexType>
1019 <xs:complexType name="CreateSequenceResponseType">
1020     <xs:sequence>
1021         <xs:element ref="wsrm:Identifier"/>
1022         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1023         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1024         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1025 maxOccurs="unbounded"/>
1026     </xs:sequence>
1027     <xs:anyAttribute namespace="##other" processContents="lax"/>
1028 </xs:complexType>
1029 <xs:complexType name="CloseSequenceType">
1030     <xs:sequence>
1031         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1032 maxOccurs="unbounded"/>
1033     </xs:sequence>
1034     <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1035     <xs:anyAttribute namespace="##other" processContents="lax"/>
1036 </xs:complexType>
1037 <xs:complexType name="CloseSequenceResponseType">
1038     <xs:sequence>
1039         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1040 maxOccurs="unbounded"/>
1041     </xs:sequence>
1042     <xs:anyAttribute namespace="##other" processContents="lax"/>
1043 </xs:complexType>
1044 <xs:complexType name="TerminateSequenceType">
1045     <xs:sequence>
1046         <xs:element ref="wsrm:Identifier"/>
1047         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1048 maxOccurs="unbounded"/>
1049     </xs:sequence>
1050     <xs:anyAttribute namespace="##other" processContents="lax"/>
1051 </xs:complexType>
1052 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1053 <xs:complexType name="OfferType">
1054     <xs:sequence>
1055         <xs:element ref="wsrm:Identifier"/>
1056         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1057         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1058 maxOccurs="unbounded"/>
1059     </xs:sequence>
1060     <xs:anyAttribute namespace="##other" processContents="lax"/>
1061 </xs:complexType>
1062 <xs:complexType name="AcceptType">
1063     <xs:sequence>
1064         <xs:element ref="wsrm:AcksTo"/>
1065         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1066 maxOccurs="unbounded"/>
1067     </xs:sequence>
1068     <xs:anyAttribute namespace="##other" processContents="lax"/>
1069 </xs:complexType>

```

```
1070 <xs:element name="Expires">
1071   <xs:complexType>
1072     <xs:simpleContent>
1073       <xs:extension base="xs:duration">
1074         <xs:anyAttribute namespace="##other" processContents="lax"/>
1075       </xs:extension>
1076     </xs:simpleContent>
1077   </xs:complexType>
1078 </xs:element>
1079 </xs:schema>
```

1080 B. Message Examples

1081 B.1 Create Sequence

1082 Create Sequence

```
1083 <?xml version="1.0" encoding="UTF-8"?>
1084 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1085 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1086 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1087   <S:Header>
1088     <wsa:MessageID>
1089       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1090     </wsa:MessageID>
1091     <wsa:To>http://example.com/serviceB/123</wsa:To>
1092     <wsa:Action>http://docs.oasis-open.org/ws-
1093 rx/wsmr/200510/CreateSequence</wsa:Action>
1094     <wsa:ReplyTo>
1095       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1096     </wsa:ReplyTo>
1097   </S:Header>
1098   <S:Body>
1099     <wsmr:CreateSequence>
1100       <wsmr:AcksTo>
1101         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1102       </wsmr:AcksTo>
1103     </wsmr:CreateSequence>
1104   </S:Body>
1105 </S:Envelope>
```

1106 Create Sequence Response

```
1107 <?xml version="1.0" encoding="UTF-8"?>
1108 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1109 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1110 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1111   <S:Header>
1112     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1113     <wsa:RelatesTo>
1114       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1115     </wsa:RelatesTo>
1116     <wsa:Action>
1117       http://docs.oasis-open.org/ws-rx/wsmr/200510/CreateSequenceResponse
1118     </wsa:Action>
1119   </S:Header>
1120   <S:Body>
1121     <wsmr:CreateSequenceResponse>
1122       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1123     </wsmr:CreateSequenceResponse>
1124   </S:Body>
1125 </S:Envelope>
```

1126 B.2 Initial Transmission

1127 The following example WS-ReliableMessaging headers illustrate the message exchange in the above
1128 figure. The three messages have the following headers; the third message is identified as the last
1129 message in the sequence:

1130 Message 1

```
1131 <?xml version="1.0" encoding="UTF-8"?>
1132 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1133 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1134 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1135   <S:Header>
1136     <wsa:MessageID>
1137       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1138     </wsa:MessageID>
1139     <wsa:To>http://example.com/serviceB/123</wsa:To>
1140     <wsa:From>
1141       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1142     </wsa:From>
1143     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1144     <wsmr:Sequence>
1145       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1146       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1147     </wsmr:Sequence>
1148   </S:Header>
1149   <S:Body>
1150     <!-- Some Application Data -->
1151   </S:Body>
1152 </S:Envelope>
```

1153 Message 2

```
1154 <?xml version="1.0" encoding="UTF-8"?>
1155 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1156 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1157 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1158   <S:Header>
1159     <wsa:MessageID>
1160       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1161     </wsa:MessageID>
1162     <wsa:To>http://example.com/serviceB/123</wsa:To>
1163     <wsa:From>
1164       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1165     </wsa:From>
1166     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1167     <wsmr:Sequence>
1168       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1169       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1170     </wsmr:Sequence>
1171   </S:Header>
1172   <S:Body>
1173     <!-- Some Application Data -->
1174   </S:Body>
1175 </S:Envelope>
```

1176 Message 3

```
1177 <?xml version="1.0" encoding="UTF-8"?>
1178 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1179 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1180 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1181   <S:Header>
1182     <wsa:MessageID>
1183       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1184     </wsa:MessageID>
1185     <wsa:To>http://example.com/serviceB/123</wsa:To>
1186     <wsa:From>
1187       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1188 </wsa:From>
1189 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1190 <wsrm:Sequence>
1191 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1192 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1193 </wsrm:Sequence>
1194 <wsrm:AckRequested>
1195 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1196 </wsrm:AckRequested>
1197 </S:Header>
1198 <S:Body>
1199 <!-- Some Application Data -->
1200 </S:Body>
1201 </S:Envelope>

```

1202 B.3 First Acknowledgement

1203 Message number 2 has not been received by the RM Destination due to some transmission error so it
1204 responds with an acknowledgement for messages 1 and 3:

```

1205 <?xml version="1.0" encoding="UTF-8"?>
1206 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1207 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1208 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1209 <S:Header>
1210 <wsa:MessageID>
1211 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1212 </wsa:MessageID>
1213 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1214 <wsa:From>
1215 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1216 </wsa:From>
1217 <wsa:Action>
1218 http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement
1219 </wsa:Action>
1220 <wsrm:SequenceAcknowledgement>
1221 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1222 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1223 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1224 </wsrm:SequenceAcknowledgement>
1225 </S:Header>
1226 <S:Body/>
1227 </S:Envelope>

```

1228 B.4 Retransmission

1229 The RM Sourcediscovers that message number 2 was not received so it resends the message and
1230 requests an acknowledgement:

```

1231 <?xml version="1.0" encoding="UTF-8"?>
1232 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1233 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1234 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1235 <S:Header>
1236 <wsa:MessageID>
1237 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1238 </wsa:MessageID>
1239 <wsa:To>http://example.com/serviceB/123</wsa:To>
1240 <wsa:From>
1241 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1242 </wsa:From>

```

```

1243 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1244 <wsrm:Sequence>
1245   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1246   <wsrm:MessageNumber>2</wsrm:MessageNumber>
1247 </wsrm:Sequence>
1248 <wsrm:AckRequested>
1249   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1250 </wsrm:AckRequested>
1251 </S:Header>
1252 <S:Body>
1253   <!-- Some Application Data -->
1254 </S:Body>
1255 </S:Envelope>

```

1256 B.5 Termination

1257 The RM Destination now responds with an acknowledgement for the complete sequence which can then
1258 be terminated:

```

1259 <?xml version="1.0" encoding="UTF-8"?>
1260 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1261 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1262 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1263   <S:Header>
1264     <wsa:MessageID>
1265       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1266     </wsa:MessageID>
1267     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1268     <wsa:From>
1269       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1270     </wsa:From>
1271     <wsa:Action>
1272       http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement
1273     </wsa:Action>
1274     <wsrm:SequenceAcknowledgement>
1275       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1276       <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1277     </wsrm:SequenceAcknowledgement>
1278   </S:Header>
1279   <S:Body/>
1280 </S:Envelope>

```

1281 Terminate Sequence

```

1282 <?xml version="1.0" encoding="UTF-8"?>
1283 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1284 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1285 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1286   <S:Header>
1287     <wsa:MessageID>
1288       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1289     </wsa:MessageID>
1290     <wsa:To>http://example.com/serviceB/123</wsa:To>
1291     <wsa:Action>
1292       http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence
1293     </wsa:Action>
1294     <wsa:From>
1295       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1296     </wsa:From>
1297   </S:Header>
1298   <S:Body>
1299     <wsrm:TerminateSequence>

```

```
1300     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1301     </wsrm:TerminateSequence>
1302     </S:Body>
1303 </S:Envelope>
```

1304 C. WSDL

1305 The non-normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1306 <http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd/wsrn-1.1.wsd>

1307 The following non-normative copy is provided for reference.

```
1308 <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1309 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1310 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1311 xmlns:rm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1312 xmlns:tns="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd"
1313 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd">
1314 <wsdl:types>
1315   <xs:schema>
1316     <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1317     schemaLocation="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1.xsd"/>
1318   </xs:schema>
1319 </wsdl:types>
1320 <wsdl:message name="CreateSequence">
1321   <wsdl:part name="create" element="rm:CreateSequence"/>
1322 </wsdl:message>
1323 <wsdl:message name="CreateSequenceResponse">
1324   <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1325 </wsdl:message>
1326 <wsdl:message name="CloseSequence">
1327   <wsdl:part name="close" element="rm:CloseSequence"/>
1328 </wsdl:message>
1329 <wsdl:message name="CloseSequenceResponse">
1330   <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1331 </wsdl:message>
1332 <wsdl:message name="TerminateSequence">
1333   <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1334 </wsdl:message>
1335 <wsdl:portType name="SequenceAbstractPortType">
1336   <wsdl:operation name="CreateSequence">
1337     <wsdl:input message="tns:CreateSequence" wsa:Action="http://docs.oasis-
1338 open.org/ws-rx/wsrn/200510/CreateSequence"/>
1339     <wsdl:output message="tns:CreateSequenceResponse"
1340 wsa:Action="http://docs.oasis-open.org/ws-
1341 rx/wsrn/200510/CreateSequenceResponse"/>
1342   </wsdl:operation>
1343   <wsdl:operation name="CloseSequence">
1344     <wsdl:input name="tns:CloseSequence" wsa:Action="http://docs.oasis-
1345 open.org/ws-rx/wsrn/200510/CloseSequence"/>
1346     <wsdl:output name="tns:CloseSequenceResponse"
1347 wsa:Action="http://docs.oasis-open.org/ws-
1348 rx/wsrn/200510/CloseSequenceResponse"/>
1349   </wsdl:operation>
1350   <wsdl:operation name="TerminateSequence">
1351     <wsdl:input message="tns:TerminateSequence"
1352 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence"/>
1353   </wsdl:operation>
1354 </wsdl:portType>
1355 </wsdl:definitions>
```

1356 **D. Acknowledgments**

1357 This document is based on initial contribution to OASIS WS-RX Technical Committee by the following
1358 authors:

1359 Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft, Doug Davis, IBM,
1360 Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM, Mary Ann Hondo,
1361 IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David Langworthy, Microsoft
1362 (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft, Steve Lucco, Microsoft,
1363 Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham, BEA, David Orchard,
1364 BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John Shewchuk, Microsoft, Tony
1365 Storey, IBM.

1366 The following individuals have provided invaluable input into the initial contribution:

1367 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,
1368 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,
1369 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,
1370 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin
1371 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,
1372 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger
1373 Wolter, Microsoft.

1374 The following individuals were members of the committee during the development of this specification:

1375 *TBD*

1376 **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 'optiona'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrn/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrn:FaultCode element)
wd-06	2005-11-02	Gilbert Pilz	Start wd-06 by changing title page from cd-01.
wd-06	2005-11-03	Gilbert Pilz	i047 (Reorder spec sections)
wd-07	2005-11-17	Gilbert Pilz	Start wd-07
wd-07	2005-11-28	Doug Davis	i071 – except for period in Appendix headings
wd-07	2005-11-28	Doug Davis	i10
wd-07	2005-11-28	Doug Davis	i030
wd-07	2005-11-28	Doug Davis	i037
wd-07	2005-11-28	Doug Davis	i038
wd-07	2005-11-28	Doug Davis	i041
wd-07	2005-11-28	Doug Davis	i043
wd-07	2005-11-28	Doug Davis	i044

<u>Rev</u>	<u>Date</u>	<u>By Whom</u>	<u>What</u>
wd-07	2005-11-28	Doug Davis	i048
wd-07	2005-11-28	Doug Davis	i051
wd-07	2005-11-28	Doug Davis	i053
wd-07	2005-11-28	Doug Davis	i059
wd-07	2005-11-28	Doug Davis	i062
wd-07	2005-11-28	Doug Davis	i063
wd-07	2005-11-28	Doug Davis	i065
wd-07	2005-11-28	Doug Davis	i067
wd-07	2005-11-28	Doug Davis	i068
wd-07	2005-11-28	Doug Davis	i069
wd-07	2005-11-28	Doug Davis	Fix bulleted list (#2) in section 2.3
wd-07	2005-11-29	Gilbert Pilz	i074 (Use of [tcShortName] in artifact locations namespaces, etc)
wd-07	2005-11-29	Gilbert Pilz	i071 – Fixed styles and formatting for TOC. Fixed styles of the appendix headings.
wd-07	2005-11-30	Doug Davis	Removed dup definition of "Receive"
wd-07	2005-11-30	Gilbert Pilz	Fixed lost formatting from heading for Namespace section. Fixed style of text body elements to match OASIS example documents. Fixed tables to match OASIS example documents.
wd-07	2005-12-01	Gilbert Pilz	Updated fix for i074 to eliminate trailing '/'. Added corresponding text around action IRI composition.
wd-07	2005-12-01	Gilbert Pilz	Use non-fixed fields for date values on both title page and body footers.
wd-07	2005-12-01	Doug Davis	Alphabetize the glossary
wd-07	2005-12-02	Doug Davis	i064
wd-07	2005-12-02	Doug Davis	i066
wd-07	2005-12-13	Steve Winkler	Editorial fixups
<u>Rev</u>	<u>Date</u>	<u>By Whom</u>	<u>What</u>
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

Rev	Date	By Whom	What
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)
wd-06	2005-11-02	Gilbert Pilz	Start wd-06 by changing title page from cd-01.
wd-06	2005-11-03	Gilbert Pilz	i047 (Reorder spec sections)
wd-07	2005-11-17	Gilbert Pilz	Start wd-07
wd-07	2005-11-28	Doug Davis	i071—except for period in Appendix headings
wd-07	2005-11-28	Doug Davis	i10
wd-07	2005-11-28	Doug Davis	i030
wd-07	2005-11-28	Doug Davis	i037
wd-07	2005-11-28	Doug Davis	i038
wd-07	2005-11-28	Doug Davis	i041
wd-07	2005-11-28	Doug Davis	i043
wd-07	2005-11-28	Doug Davis	i044
wd-07	2005-11-28	Doug Davis	i048
wd-07	2005-11-28	Doug Davis	i051
wd-07	2005-11-28	Doug Davis	i053
wd-07	2005-11-28	Doug Davis	i059
wd-07	2005-11-28	Doug Davis	i062
wd-07	2005-11-28	Doug Davis	i063
wd-07	2005-11-28	Doug Davis	i065
wd-07	2005-11-28	Doug Davis	i067
wd-07	2005-11-28	Doug Davis	i068
wd-07	2005-11-28	Doug Davis	i069
wd-07	2005-11-28	Doug Davis	Fix bulleted list (#2) in section 2.3
wd-07	2005-11-29	Gilbert Pilz	i074 (Use of [tcShortName] in artifact locations namespaces, etc)
wd-07	2005-11-29	Gilbert Pilz	i071—Fixed styles and formatting for TOC. Fixed styles of the appendix headings.
wd-07	2005-11-30	Doug Davis	Removed dup definition of "Receive"

Rev	Date	By Whom	What
wd-07	2005-11-30	Gilbert Pilz	Fixed lost formatting from heading for Namespace section. Fixed style of text body elements to match OASIS example documents. Fixed tables to match OASIS example documents.
wd-07	2005-12-01	Gilbert Pilz	Updated fix for i074 to eliminate trailing '/'. Added corresponding text around action IRI composition.
wd-07	2005-12-01	Gilbert Pilz	Use non-fixed fields for date values on both title page and body footers.
wd-07	2005-12-01	Doug Davis	Alphabetize the glossary
wd-07	2005-12-02	Doug Davis	i064
wd-07	2005-12-02	Doug Davis	i066

1377 **F. Notices**

1378 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
1379 might be claimed to pertain to the implementation or use of the technology described in this document or
1380 the extent to which any license under such rights might or might not be available; neither does it represent
1381 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to
1382 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made
1383 available for publication and any assurances of licenses to be made available, or the result of an attempt
1384 made to obtain a general license or permission for the use of such proprietary rights by implementors or
1385 users of this specification, can be obtained from the OASIS Executive Director.

1386 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or
1387 other proprietary rights which may cover technology that may be required to implement this specification.
1388 Please address the information to the OASIS Executive Director.

1389 Copyright (C) OASIS Open (2005). All Rights Reserved.

1390 This document and translations of it may be copied and furnished to others, and derivative works that
1391 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and
1392 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and
1393 this paragraph are included on all such copies and derivative works. However, this document itself may
1394 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as
1395 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights
1396 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it
1397 into languages other than English.

1398 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
1399 or assigns.

1400 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1401 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
1402 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR
1403 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.