



1 Web Services Reliable Messaging 2 (WS-ReliableMessaging)

3 Working Draft 16, November 20, 2006

4 Document identifier:

5 wsrn-1.1-spec-wd-16

6 Location:

7 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-spec-wd-16.pdf>

8 Editors:

9 Doug Davis, IBM <dug@us.ibm.com>
10 Anish Karmarkar, Oracle <Anish.Karmarkar@oracle.com>
11 Gilbert Pilz, BEA <gpilz@bea.com>
12 Steve Winkler, SAP <steve.winkler@sap.com>
13 Ümit Yalçınalp, SAP <umit.yalcinalp@sap.com>

14 Contributors:

15 See the Acknowledgments (Appendix E).

16 Abstract:

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

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

25 By using the XML [XML], SOAP [SOAP 1.1], [SOAP 1.2] and WSDL [WSDL 1.1] extensibility model,
26 SOAP-based and WSDL-based specifications are designed to be composed with each other to define a
27 rich Web services environment. As such, WS-ReliableMessaging by itself does not define all the features
28 required for a complete messaging solution. WS-ReliableMessaging is a building block that is used in
29 conjunction with other specifications and application-specific protocols to accommodate a wide variety of
30 requirements and scenarios related to the operation of distributed Web services.

31 Status:

32 This document was last revised or approved by the WS-RX on the above date. The level of approval is
33 also listed above. Check the current location noted above for possible later revisions of this document.
34 This document is updated periodically on no particular schedule. Technical Committee members should
35 send comments on this specification to the Technical Committee's email list. Others should send
36 comments to the Technical Committee by using the "Send A Comment" button on the Technical
37 Committee's web page at <http://www.oasis-open.org/committees/ws-rx>. For information on whether any
38 patents have been disclosed that may be essential to implementing this specification, and any offers of
39 patent licensing terms, please refer to the Intellectual Property Rights section of the Technical
40 Committee web page (<http://www.oasis-open.org/committees/ws-rx/ipr.php>). The non-normative errata
41 page for this specification is located at <http://www.oasis-open.org/committees/ws-rx>.

42 Table of Contents

43	1 Introduction.....	4
44	1.1 Notational Conventions.....	4
45	1.2 Namespace.....	5
46	1.3 Conformance.....	5
47	2 Reliable Messaging Model.....	6
48	2.1 Glossary.....	6
49	2.2 Protocol Preconditions.....	7
50	2.3 Protocol Invariants.....	8
51	2.4 Example Message Exchange.....	8
52	3 RM Protocol Elements.....	10
53	3.1 Considerations on the Use of Extensibility Points.....	10
54	3.2 Considerations on the Use of "Piggy-Backing".....	10
55	3.3 Composition with WS-Addressing.....	10
56	3.4 Sequence Creation.....	10
57	3.5 Closing A Sequence.....	15
58	3.6 Sequence Termination.....	16
59	3.7 Sequences.....	18
60	3.8 Request Acknowledgement.....	19
61	3.9 Sequence Acknowledgement.....	20
62	4 Faults.....	23
63	4.1 SequenceFault Element.....	24
64	4.2 Sequence Terminated.....	25
65	4.3 Unknown Sequence.....	25
66	4.4 Invalid Acknowledgement.....	26
67	4.5 Message Number Rollover.....	26
68	4.6 Create Sequence Refused.....	27
69	4.7 Sequence Closed.....	27
70	4.8 WSRM Required.....	28
71	5 Security Threats and Countermeasures.....	29
72	5.1 Threats and Countermeasures.....	29
73	5.1.1 Integrity Threats.....	29
74	5.1.1.1 Countermeasures.....	29
75	5.1.2 Resource Consumption Threats.....	30
76	5.1.2.1 Countermeasures.....	30
77	5.1.3 Sequence Spoofing Threats.....	30
78	5.1.3.1 Sequence Hijacking.....	30
79	5.1.3.2 Countermeasures.....	30

80	5.2 Security Solutions and Technologies.....	31
81	5.2.1 Transport Layer Security.....	31
82	5.2.1.1 Model.....	31
83	5.2.1.2 Countermeasure Implementation.....	32
84	5.2.2 SOAP Message Security.....	33
85	5.2.2.1 Model.....	33
86	5.2.2.2 Countermeasure Implementation.....	33
87	6 Securing Sequences.....	35
88	6.1 Securing Sequences Using WS-Security.....	35
89	6.2 Securing Sequences Using SSL/TLS.....	36
90	7 References.....	38
91	7.1 Normative.....	38
92	7.2 Non-Normative.....	39
93	Appendix A. Schema.....	41
94	Appendix B. WSDL.....	46
95	Appendix C. Message Examples.....	48
96	Appendix C.1 Create Sequence.....	48
97	Appendix C.2 Initial Transmission.....	48
98	Appendix C.3 First Acknowledgement.....	50
99	Appendix C.4 Retransmission.....	50
100	Appendix C.5 Termination.....	51
101	Appendix D. State Tables.....	53
102	Appendix E. Acknowledgments.....	57
103	Appendix F. Revision History.....	58
104	Appendix G. Notices.....	64

105 **1 Introduction**

106 It is often a requirement for two Web services that wish to communicate to do so reliably in the presence
107 of software component, system, or network failures. The primary goal of this specification is to create a
108 modular mechanism for reliable transfer of messages. It defines a messaging protocol to identify, track,
109 and manage the reliable transfer of messages between a source and a destination. It also defines a
110 SOAP binding that is required for interoperability. Additional bindings can be defined.

111 This mechanism is extensible allowing additional functionality, such as security, to be tightly integrated.
112 This specification integrates with and complements the WS-Security [[WS-Security](#)], WS-Policy [[WS-](#)
113 [Policy](#)], and other Web services specifications. Combined, these allow for a broad range of reliable,
114 secure messaging options.

115 **1.1 Notational Conventions**

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

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

- 120 • The syntax appears as an XML instance, but values in italics indicate data types instead of values.
- 121 • Characters are appended to elements and attributes to indicate cardinality:
 - 122 ○ "?" (0 or 1)
 - 123 ○ "*" (0 or more)
 - 124 ○ "+" (1 or more)
- 125 • The character "|" is used to indicate a choice between alternatives.
- 126 • The characters "[" and "]" are used to indicate that contained items are to be treated as a group
127 with respect to cardinality or choice.
- 128 • An ellipsis (i.e. "...") indicates a point of extensibility that allows other child or attribute content
129 specified in this document. Additional children elements and/or attributes MAY be added at the
130 indicated extension points but they MUST NOT contradict the semantics of the parent and/or
131 owner, respectively. If an extension is not recognized it SHOULD be ignored.
- 132 • XML namespace prefixes (See Section [1.2](#)) are used to indicate the namespace of the element
133 being defined.

134 Elements and Attributes defined by this specification are referred to in the text of this document using
135 XPath 1.0 [[XPATH 1.0](#)] expressions. Extensibility points are referred to using an extended version of this
136 syntax:

- 137 • An element extensibility point is referred to using {any} in place of the element name. This
138 indicates that any element name can be used, from any namespace other than the wsrn:
139 namespace.
- 140 • An attribute extensibility point is referred to using @{any} in place of the attribute name. This
141 indicates that any attribute name can be used, from any namespace other than the wsrn:
142 namespace.

143 1.2 Namespace

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

145 <http://docs.oasis-open.org/ws-rx/wsrn/200608>

146 Dereferencing the above URI will produce the Resource Directory Description Language [RDDL 2.0]
147 document that describes this namespace.

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

150 Table 1

Prefix	Namespace
S	(Either SOAP 1.1 or 1.2)
S11	http://schemas.xmlsoap.org/soap/envelope/
S12	http://www.w3.org/2003/05/soap-envelope
wsrn	http://docs.oasis-open.org/ws-rx/wsrn/200608
wsa	http://www.w3.org/2005/08/addressing
wsaw	http://www.w3.org/2006/05/addressing/wsdl
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
xs	http://www.w3.org/2001/XMLSchema

151 The normative schema for WS-ReliableMessaging can be found linked from the namespace document
152 that is located at the namespace URI specified above.

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

154 1.3 Conformance

155 An implementation is not conformant with this specification if it fails to satisfy one or more of the MUST or
156 REQUIRED level requirements defined herein. A SOAP Node MUST NOT use the XML namespace
157 identifier for this specification (listed in Section 1.2) within SOAP Envelopes unless it is conformant with
158 this specification.

159 Normative text within this specification takes precedence over normative outlines, which in turn take
160 precedence over the XML Schema [XML Schema Part 1, Part 2] descriptions.

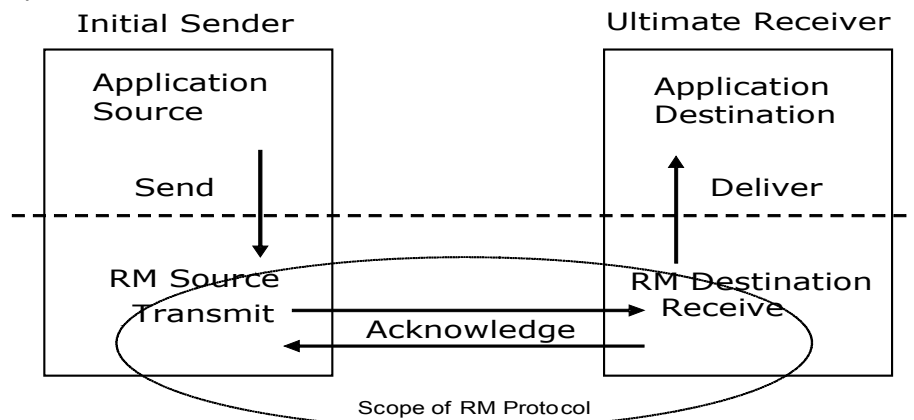
161 2 Reliable Messaging Model

162 Many errors can interrupt a conversation. Messages can be lost, duplicated or reordered. Further the host
163 systems can experience failures and lose volatile state.

162 The WS-ReliableMessaging specification defines an interoperable protocol that enables a Reliable
163 Messaging (RM) Source to accurately determine the disposition of each message it Transmits as
164 perceived by the RM Destination, so as to allow it to resolve any in-doubt status regarding receipt of the
165 message Transmitted. The protocol also enables an RM Destination to efficiently determine which of
166 those messages it Receives have been previously Received, enabling it to filter out duplicate message
167 transmissions caused by the retransmission, by the RM Source, of an unacknowledged message. It also
168 enables an RM Destination to Deliver the messages it Receives to the Application Destination in the order
169 in which they were sent by an Application Source, in the event that they are Received out of order. Note
170 that this specification places no restriction on the scope of the RM Source or RM Destination entities. For
171 example, either can span multiple WSDL Ports or Endpoints.

172 The protocol enables the implementation of a broad range of reliability features which include ordered
173 Delivery, duplicate elimination, and guaranteed receipt. The protocol can also be implemented with a
174 range of robustness characteristics ranging from in-memory persistence that is scoped to a single process
175 lifetime, to replicated durable storage that is recoverable in all but the most extreme circumstances. It is
176 expected that the Endpoints will implement as many or as few of these reliability characteristics as
177 necessary for the correct operation of the application using the protocol. Regardless of which of the
178 reliability features is enabled, the wire protocol does not change.

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



185 Figure 1: Reliable Messaging Model

186 2.1 Glossary

187 The following definitions are used throughout this specification:

188 **Accept:** The act of qualifying a message by the RM Destination such that it becomes eligible for Delivery
189 and acknowledgement.

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

192 **Acknowledgement Message:** A message containing a `SequenceAcknowledgement` header block.
193 Acknowledgement Messages may or may not contain a SOAP body.

194 **Acknowledgement Request:** A message containing an `AckRequested` header. Acknowledgement
195 Requests may or may not contain a SOAP body.

196 **Application Destination:** The Endpoint to which a message is Delivered.

197 **Application Source:** The Endpoint that Sends a message.

198 **Back-channel:** When the underlying transport provides a mechanism to return a transport-protocol
199 specific response, capable of carrying a SOAP message, without initiating a new connection, this
200 specification refers to this mechanism as a back-channel.

201 **Deliver:** The act of transferring a message from the RM Destination to the Application Destination.

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

205 **Receive:** The act of reading a message from a network connection and accepting it.

206 **RM Destination:** The Endpoint that Receives messages Transmitted reliably from an RM Source.

207 **RM Protocol Header Block:** One of `Sequence`, `SequenceAcknowledgement`, or `AckRequested`.

208 **RM Source:** The Endpoint that Transmits messages reliably to an RM Destination.

209 **Send:** The act of transferring a message from the Application Source to the RM Source for reliable
210 transfer.

211 **Sequence Lifecycle Message:** A message that contains one of: `CreateSequence`,
212 `CreateSequenceResponse`, `CloseSequence`, `CloseSequenceResponse`, `TerminateSequence`,
213 `TerminateSequenceResponse` as the child element of the SOAP body element.

214 **Sequence Traffic Message:** A message containing a `Sequence` header block.

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

216 **2.2 Protocol Preconditions**

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

- 219 • For any single message exchange the RM Source **MUST** have an endpoint reference that uniquely
220 identifies the RM Destination Endpoint.
- 221 • The RM Source **MUST** have successfully created a `Sequence` with the RM Destination.
- 222 • The RM Source **MUST** be capable of formulating messages that adhere to the RM Destination's
223 policies.
- 224 • If a secure exchange of messages is **REQUIRED**, then the RM Source and RM Destination **MUST**
225 have a security context.

226 2.3 Protocol Invariants

227 During the lifetime of a Sequence, two invariants are REQUIRED for correctness:

- 228 • The RM Source MUST assign each message within a Sequence a message number (defined
229 below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers
230 MUST be assigned in the same order in which messages are sent by the Application Source.
- 231 • Within every Acknowledgement Message it issues, the RM Destination MUST include one or more
232 AcknowledgementRange child elements that contain, in their collective ranges, the message
233 number of every message accepted by the RM Destination. The RM Destination MUST exclude, in
234 the AcknowledgementRange elements, the message numbers of any messages it has not
235 accepted. If no messages have been received the RM Destination MUST return None instead of an
236 AcknowledgementRange (s). The RM Destination MAY transmit a Nack for a specific message
237 or messages in stead of an AcknowledgementRange (s).

238 2.4 Example Message Exchange

239 Figure 2 illustrates a possible message exchange between two reliable messaging Endpoints A and B.

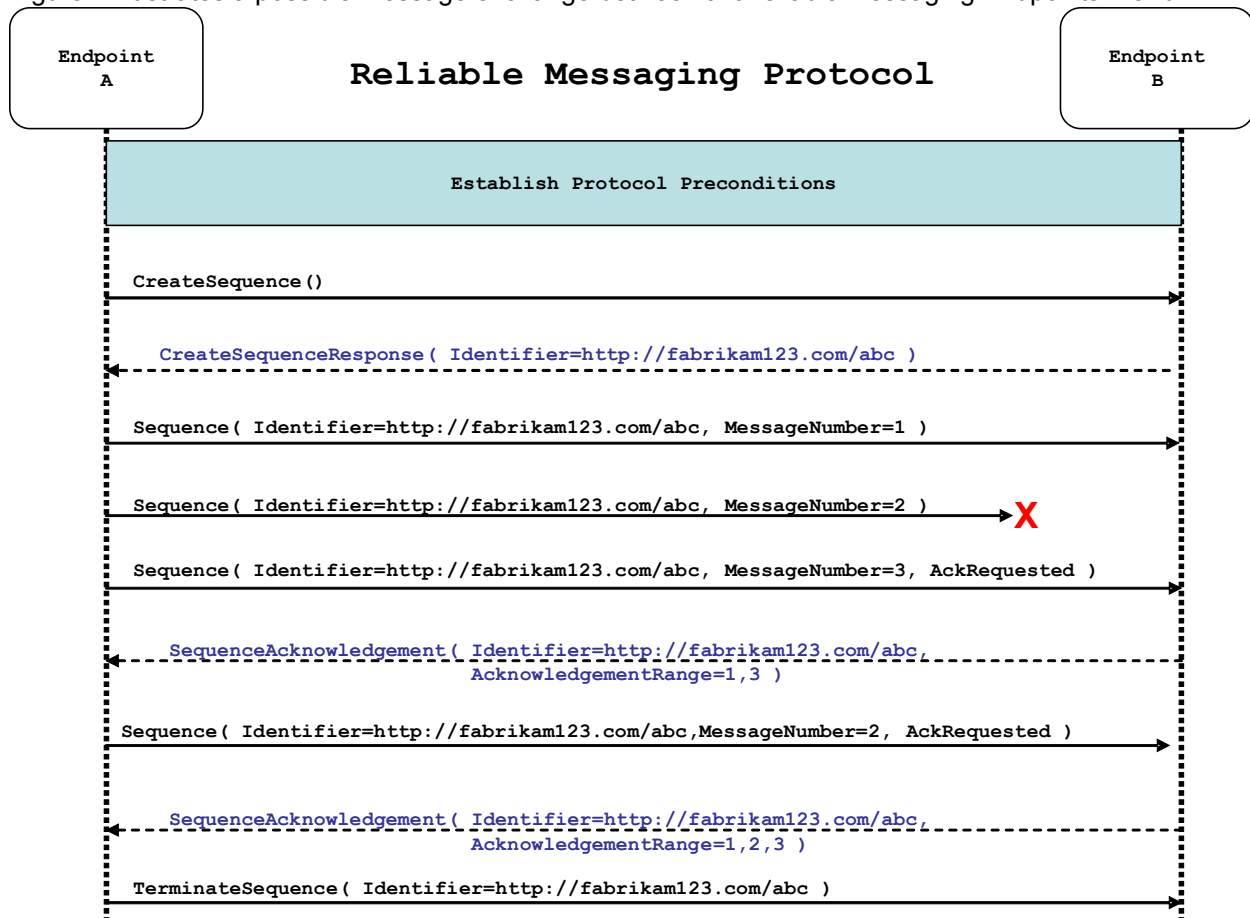


Figure 2: The WS-ReliableMessaging Protocol

- 240 1. The protocol preconditions are established. These include policy exchange, endpoint resolution,
241 and establishing trust.

- 242 2. The RM Source requests creation of a new Sequence.
- 243 3. The RM Destination creates a new Sequence and returns its unique identifier.
- 244 4. The RM Source begins Transmitting messages in the Sequence beginning with MessageNumber 1.
245 In the figure above, the RM Source sends 3 messages in the Sequence.
- 246 5. The 2nd message in the Sequence is lost in transit.
- 247 6. The 3rd message is the last in this Sequence and the RM Source includes an `AckRequested`
248 header to ensure that it gets a timely `SequenceAcknowledgement` for the Sequence.
- 249 7. The RM Destination acknowledges receipt of message numbers 1 and 3 as a result of receiving the
250 RM Source's `AckRequested` header.
- 251 8. The RM Source retransmits the unacknowledged message with MessageNumber 2. This is a new
252 message from the perspective of the underlying transport, but it has the same Sequence Identifier
253 and MessageNumber so the RM Destination can recognize it as a duplicate of the earlier message,
254 in case the original and retransmitted messages are both Received. The RM Source includes an
255 `AckRequested` header in the retransmitted message so the RM Destination will expedite an
256 acknowledgement.
- 257 9. The RM Destination Receives the second transmission of the message with MessageNumber 2
258 and acknowledges receipt of message numbers 1, 2, and 3.
- 259 10. The RM Source Receives this Acknowledgement and sends a `TerminateSequence` message to the
260 RM Destination indicating that the Sequence is completed and reclaims any resources associated
261 with the Sequence.
- 262 11. The RM Destination Receives the `TerminateSequence` message indicating that the RM Source will
263 not be sending any more messages. The RM Destination sends a `TerminateSequenceResponse`
264 message to the RM Source and reclaims any resources associated with the Sequence.

265 The RM Source will expect to Receive Acknowledgements from the RM Destination during the course of a
266 message exchange at occasions described in Section 3 below. Should an Acknowledgement not be
267 Received in a timely fashion, the RM Source MUST re-transmit the message since either the message or
268 the associated Acknowledgement might have been lost. Since the nature and dynamic characteristics of
269 the underlying transport and potential intermediaries are unknown in the general case, the timing of re-
270 transmissions cannot be specified. Additionally, over-aggressive re-transmissions have been
271 demonstrated to cause transport or intermediary flooding which are counterproductive to the intention of
272 providing a reliable exchange of messages. Consequently, implementers are encouraged to utilize
273 adaptive mechanisms that dynamically adjust re-transmission time and the back-off intervals that are
274 appropriate to the nature of the transports and intermediaries envisioned. For the case of TCP/IP
275 transports, a mechanism similar to that described as RTTM in RFC 1323 [[RTTM](#)] SHOULD be
276 considered.

277 Now that the basic model has been outlined, the details of the elements used in this protocol are now
278 provided in Section 3.

279 **3 RM Protocol Elements**

280 The following sub-sections define the various RM protocol elements, and prescribe their usage by a
281 conformant implementations.

282 **3.1 Considerations on the Use of Extensibility Points**

283 The following protocol elements define extensibility points at various places. Implementations MAY add
284 child elements and/or attributes at the indicated extension points but MUST NOT contradict the semantics
285 of the parent and/or owner, respectively. If a receiver does not recognize an extension, the receiver
286 SHOULD ignore the extension.

287 **3.2 Considerations on the Use of "Piggy-Backing"**

288 Some RM header blocks may be added to messages that are targeted to the same Endpoint to which
289 those headers are to be sent (a concept often referred to as "piggy-backing"), thus saving the overhead of
290 an additional message exchange. Reference parameters MUST be considered when determining whether
291 two EPRs are targeted to the same Endpoint. See the sections that define each RM header block to know
292 which ones may be considered for piggy-backing.

293 **3.3 Composition with WS-Addressing**

294 When the RM protocol, defined in this specification, is composed with the WS-Addressing specification,
295 the following rules prescribe the constraints on the value of the `wsa:Action` header:

- 296 1. When an Endpoint generates a message that carries an RM protocol element, that is defined in
297 section 3 below, in the body of a SOAP envelope that Endpoint MUST include in that envelope a
298 `wsa:Action` SOAP header block whose value is an IRI that is a concatenation of the WS-RM
299 namespace URI, followed by a "/", followed by the value of the local name of the child element of
300 the SOAP body. For example, for a Sequence creation request message as described in section
301 3.4 below, the value of the `wsa:Action` IRI would be:

```
302 http://docs.oasis-open.org/ws-rx/wsrn/200608/CreateSequence
```

- 303 2. When an Endpoint generates an Acknowledgement Message that has no element content in the
304 SOAP body, then the value of the `wsa:Action` IRI MUST be:

```
305 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
```

- 306 3. When an Endpoint generates an Acknowledgement Request that has no element content in the
307 SOAP body, then the value of the `wsa:Action` IRI MUST be:

```
308 http://docs.oasis-open.org/ws-rx/wsrn/200608/AckRequested
```

- 309 4. When an Endpoint generates an RM fault as defined in section 4 below, the value of the
310 `wsa:Action` IRI MUST be as defined in section 4 below.

311 **3.4 Sequence Creation**

312 The RM Source MUST request creation of an outbound Sequence by sending a `CreateSequence`
313 element in the body of a message to the RM Destination which in turn responds either with a message
314 containing `CreateSequenceResponse` or a `CreateSequenceRefused` fault. The RM Source MAY
315 include an offer to create an inbound Sequence within the `CreateSequence` message. This offer is
316 either accepted or rejected by the RM Destination in the `CreateSequenceResponse` message.

317 The SOAP version used for the `CreateSequence` message SHOULD be used for all subsequent
318 messages in or for that Sequence, sent by either the RM Source or the RM Destination.

319 The following exemplar defines the `CreateSequence` syntax:

```
320 <wsrm:CreateSequence ...>
321   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>
322   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
323   <wsrm:Offer ...>
324     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
325     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>
326     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
327     <wsrm:IncompleteSequenceBehavior>
328       wsrml:IncompleteSequenceBehaviorType
329     </wsrm:IncompleteSequenceBehavior> ?
330     ...
331   </wsrm:Offer> ?
332   ...
333 </wsrm:CreateSequence>
```

334 The following describes the content model of the `CreateSequence` element.

335 `/wsrm:CreateSequence`

336 This element requests creation of a new Sequence between the RM Source that sends it, and the RM
337 Destination to which it is sent. The RM Source MUST NOT send this element as a header block. The RM
338 Destination MUST respond either with a `CreateSequenceResponse` response message or a
339 `CreateSequenceRefused` fault.

340 `/wsrm:CreateSequence/wsrm:AcksTo`

341 The RM Source MUST include this element in any `CreateSequence` message it sends. This element is of
342 type `wsa:EndpointReferenceType` (as specified by WS-Addressing). It specifies the endpoint
343 reference to which messages containing `SequenceAcknowledgement` header blocks and faults related
344 to the created Sequence are to be sent, unless otherwise noted in this specification (for example, see
345 Section 3.5).

346 Implementations MUST NOT use an endpoint reference in the `AcksTo` element that would prevent the
347 sending of Sequence Acknowledgements back to the RM Source. For example, using the WS-Addressing
348 "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever
349 send Sequence Acknowledgements.

350 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

358 `/wsrm:CreateSequence/wsrm:Offer`

359 This element, if present, enables an RM Source to offer a corresponding Sequence for the reliable
360 exchange of messages Transmitted from RM Destination to RM Source.

361 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier`

362 The RM Source MUST set the value of this element to an absolute URI (conformant with RFC3986 [URI])
363 that uniquely identifies the offered Sequence.

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

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

367 /wsmr:CreateSequence/wsmr:Offer/wsmr:Endpoint

368 An RM Source MUST include this element, of type `wsa:EndpointReferenceType` (as specified by
369 WS-Addressing). This element specifies the endpoint reference to which Sequence Lifecycle Messages,
370 Sequence Traffic Messages, Acknowledgement Requests, and fault messages related to the offered
371 Sequence are to be sent.

372 Implementations MUST NOT use an endpoint reference in the Endpoint element that would prevent the
373 sending of Sequence Lifecycle Message, Sequence Traffic Message, etc. For example, using the WS-
374 Addressing "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM
375 Destination to ever send Sequence Lifecycle Messages (e.g. `TerminateSequence`) to the RM Source
376 for the Offered Sequence. Implementations MAY use the WS-MakeConnection anonymous URI template
377 and doing so implies that messages will be retrieved using a mechanism such as the `MakeConnection`
378 message.

379 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

380 This element, if present, of type `xs:duration` specifies the duration for the offered Sequence. A value of
381 "PT0S" indicates that the offered Sequence will never expire. Absence of the element indicates an implied
382 value of "PT0S".

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

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

386 /wsmr:CreateSequence/wsmr:Offer/wsmr:IncompleteSequenceBehavior

387 This element, if present, specifies the behavior that the destination will exhibit upon the closure or
388 termination of an incomplete Sequence. For the purposes of defining the values used, the term "discard"
389 refers to behavior equivalent to the Application Destination never processing a particular message.

390 A value of "DiscardEntireSequence" indicates that the entire Sequence MUST be discarded if the
391 Sequence is closed, or terminated, when there are one or more gaps in the final
392 `SequenceAcknowledgement`.

393 A value of "DiscardFollowingFirstGap" indicates that messages in the Sequence beyond the first gap
394 MUST be discarded when there are one or more gaps in the final `SequenceAcknowledgement`.

395 The default value of "NoDiscard" indicates that no acknowledged messages in the Sequence will be
396 discarded.

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

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

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

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

403 /wsmr:CreateSequence/{any}

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

406 /wsmr:CreateSequence/@{any}

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

409 A `CreateSequenceResponse` is sent in the body of a response message by an RM Destination in
410 response to receipt of a `CreateSequence` request message. It carries the `Identifier` of the created
411 Sequence and indicates that the RM Source can begin sending messages in the context of the identified
412 Sequence.

413 The following exemplar defines the `CreateSequenceResponse` syntax:

```
414 <wsmr:CreateSequenceResponse ...>  
415   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
416   <wsmr:Expires ...> xs:duration </wsmr:Expires> ?  
417   <wsmr:IncompleteSequenceBehavior>  
418     wsmr:IncompleteSequenceBehaviorType  
419   </wsmr:IncompleteSequenceBehavior> ?  
420   <wsmr:Accept ...>  
421     <wsmr:AcksTo wsa:EndpointReferenceType </wsmr:AcksTo>  
422     ...  
423   </wsmr:Accept> ?  
424   ...  
425 </wsmr:CreateSequenceResponse>
```

426 The following describes the content model of the `CreateSequenceResponse` element.

427 /wsmr:CreateSequenceResponse

428 This element is sent in the body of the response message in response to a `CreateSequence` request
429 message. It indicates that the RM Destination has created a new Sequence at the request of the RM
430 Source. The RM Destination MUST NOT send this element as a header block.

431 /wsmr:CreateSequenceResponse/wsmr:Identifier

432 The RM Destination MUST include this element within any `CreateSequenceResponse` message it sends.
433 The RM Destination MUST set the value of this element to the absolute URI (conformant with RFC3986)
434 that uniquely identifies the Sequence that has been created by the RM Destination.

435 /wsmr:CreateSequenceResponse/wsmr:Identifier/@{any}

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

438 /wsmr:CreateSequenceResponse/wsmr:Expires

439 This element, if present, of type `xs:duration` accepts or refines the RM Source's requested duration for
440 the Sequence. It specifies the amount of time after which any resources associated with the Sequence
441 SHOULD be reclaimed thus causing the Sequence to be silently terminated. At the RM Destination this
442 duration is measured from a point proximate to Sequence creation and at the RM Source this duration is
443 measured from a point approximate to the successful processing of the `CreateSequenceResponse`. A
444 value of "PT0S" indicates that the Sequence will never expire. Absence of the element indicates an
445 implied value of "PT0S". The RM Destination MUST set the value of this element to be equal to or less
446 than the value requested by the RM Source in the corresponding `CreateSequence` message.

447 /wsmr:CreateSequenceResponse/wsmr:Expires/@{any}

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

450 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`

451 This element, if present, specifies the behavior that the destination will exhibit upon the closure or
452 termination of an incomplete Sequence. For the purposes of defining the values used, the term "discard"
453 refers to behavior equivalent to the Application Destination never processing a particular message.

454 A value of "DiscardEntireSequence" indicates that the entire Sequence MUST be discarded if the
455 Sequence is closed, or terminated, when there are one or more gaps in the final
456 `SequenceAcknowledgement`.

457 A value of "DiscardFollowingFirstGap" indicates that messages in the Sequence beyond the first gap
458 MUST be discarded when there are one or more gaps in the final `SequenceAcknowledgement`.

459 The default value of "NoDiscard" indicates that no acknowledged messages in the Sequence will be
460 discarded.

461 `/wsrm:CreateSequenceResponse/wsrm:Accept`

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

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

467 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

468 The RM Destination MUST include this element, of type `wsa:EndpointReferenceType` (as specified
469 by WS-Addressing). It specifies the endpoint reference to which messages containing
470 `SequenceAcknowledgement` header blocks and faults related to the created Sequence are to be sent,
471 unless otherwise noted in this specification (for example, see Section 3.5).

472 Implementations MUST NOT use an endpoint reference in the `AcksTo` element that would prevent the
473 sending of Sequence Acknowledgements back to the RM Source. For example, using the WS-Addressing
474 "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever
475 send Sequence Acknowledgements.

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

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

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

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

482 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

488 3.5 Closing A Sequence

489 There are times during the use of an RM Sequence that the RM Source or RM Destination will wish to
490 discontinue using a Sequence. Simply terminating the Sequence discards the state managed by the RM
491 Destination, leaving the RM Source unaware of the final ranges of messages that were successfully
492 transferred to the RM Destination. To ensure that the Sequence ends with a known final state either the
493 RM Source or RM Destination MAY choose to close the Sequence before terminating it.

494 If the RM Source wishes to close the Sequence, then it sends a `CloseSequence` element, in the body of
495 a message, to the RM Destination. This message indicates that the RM Destination MUST NOT accept
496 any new messages for the specified Sequence, other than those already accepted at the time the
497 `CloseSequence` element is interpreted by the RM Destination. Upon receipt of this message, or
498 subsequent to the RM Destination closing the Sequence of its own volition, the RM Destination MUST
499 include a final `SequenceAcknowledgement` (within which the RM Destination MUST include the `Final`
500 element) header block on any messages associated with the Sequence destined to the RM Source,
501 including the `CloseSequenceResponse` message or on any Sequence fault Transmitted to the RM
502 Source.

503 If the RM Destination decides to close a Sequence of its own volition, it MAY inform the RM Source of this
504 event by sending a `CloseSequence` element, in the body of a message, to the `AcksTo` EPR of that
505 Sequence. The RM Destination MUST include a final `SequenceAcknowledgement` (within which the RM
506 Destination MUST include the `Final` element) header block in this message and any subsequent
507 messages associated with the Sequence destined to the RM Source.

508 While the RM Destination MUST NOT accept any new messages for the specified Sequence it MUST still
509 process Sequence Lifecycle Messages and Acknowledgement Requests. For example, it MUST respond to
510 `AckRequested`, `TerminateSequence` as well as `CloseSequence` messages. Note, subsequent
511 `CloseSequence` messages have no effect on the state of the Sequence.

512 In the case where the RM Destination wishes to discontinue use of a Sequence it is RECOMMENDED
513 that it close the Sequence. Please see `Final` and the `SequenceClosed` fault. Whenever possible the
514 `SequenceClosed` fault SHOULD be used in place of the `SequenceTerminated` fault to allow the RM
515 Source to still Receive Acknowledgements.

516 The following exemplar defines the `CloseSequence` syntax:

```
517 <wsmr:CloseSequence ...>  
518   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
519   ...  
520 </wsmr:CloseSequence>
```

521 The following describes the content model of the `CloseSequence` element.

522 `/wsmr:CloseSequence`

523 This element isMAY be sent by an RM Source to indicate that the RM Destination MUST NOT accept any
524 new messages for this Sequence. This element MAY also be sent by an RM Destination to indicate that it
525 will not accept any new messages for this Sequence.

526 `/wsmr:CloseSequence/wsmr:Identifier`

527 The RM Source or RM Destination MUST include this element in any `CloseSequence` messages it sends.
528 The RM Source or RM Destination MUST set the value of this element to the absolute URI (conformant
529 with RFC3986) of the Sequence that is being closed.

530 `/wsmr:CloseSequence/wsmr:Identifier/@{any}`

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

533 /wsmr:CloseSequence/{any}

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

536 /wsmr:CloseSequence@{any}

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

539 A `CloseSequenceResponse` is sent in the body of a `response`-message ~~by an RM Destination~~ in
540 response to receipt of a `CloseSequence` request message. It indicates that the ~~responding partyRM-~~
541 ~~Destination~~ has closed the Sequence.

542 The following exemplar defines the `CloseSequenceResponse` syntax:

```
543 <wsmr:CloseSequenceResponse ...>  
544   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
545   ...  
546 </wsmr:CloseSequenceResponse>
```

547 The following describes the content model of the `CloseSequenceResponse` element.

548 /wsmr:CloseSequenceResponse

549 This element is sent in the body of a `response`-message ~~by an RM Destination~~ in response to receipt of a
550 `CloseSequence` request message. It indicates that the ~~responding partyRM-Destination~~ has closed the
551 Sequence.

552 /wsmr:CloseSequenceResponse/wsmr:Identifier

553 The ~~responding party (RMS or RMD)RM-Destination~~ MUST include this element in any
554 `CloseSequenceResponse` message it sends. The ~~responding partyRM-Destination~~ MUST set the value
555 of this element to the absolute URI (conformant with RFC3986) of the Sequence that is being closed.

556 /wsmr:CloseSequenceResponse/wsmr:Identifier/@{any}

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

559 /wsmr:CloseSequenceResponse/{any}

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

562 /wsmr:CloseSequenceResponse@{any}

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

565 3.6 Sequence Termination

566 When the RM Source has completed its use of the Sequence it sends a `TerminateSequence` element,
567 in the body of a message, to the RM Destination to indicate that the Sequence is complete and that it will
568 not be sending any further messages related to the Sequence. The RM Destination can safely reclaim any
569 resources associated with the Sequence upon receipt of the `TerminateSequence` message. Under
570 normal usage the RM Source will complete its use of the Sequence when all of the messages in the

571 Sequence have been acknowledged. However, the RM Source is free to Terminate or Close a Sequence
572 at any time regardless of the acknowledgement state of the messages.

573 If the RM Destination decides to terminate a Sequence of its own volition, it MAY inform the RM Source of
574 this event by sending a TerminateSequence element, in the body of a message, to the AcksTo EPR for
575 that Sequence. The RM Destination MUST include a final SequenceAcknowledgement (within which
576 the RM Destination MUST include the Final element) header block in this message.

577 The following exemplar defines the TerminateSequence syntax:

```
578 <wsrm:TerminateSequence ...>  
579   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
580   ...  
581 </wsrm:TerminateSequence>
```

582 The following describes the content model of the TerminateSequence element.

583 /wsrm:TerminateSequence

584 This element **MAY** be sent by an RM Source to indicate it has completed its use of the Sequence. It
585 indicates that the RM Destination can safely reclaim any resources related to the identified Sequence. The
586 RM Source **MUST NOT** send this element as a header block. The RM Source **MAY** retransmit this
587 element. Once this element is sent, other than this element, the RM Source **MUST NOT** send any
588 additional message to the RM Destination referencing this Sequence.

589 This element MAY also be sent by the RM Destination to indicate that it has unilaterally terminated the
590 Sequence. Upon sending this message the RM Destination MUST NOT accept any additional messages
591 (with the exception of the corresponding TerminateSequenceResponse) for this Sequence. Upon
592 receipt of a TerminateSequence the RM Source MUST NOT send any additional messages (with the
593 exception of the corresponding TerminateSequenceResponse) for this Sequence.

594 /wsrm:TerminateSequence/wsrm:Identifier

595 The RM Source **or RM Destination** **MUST** include this element in any TerminateSequence message it
596 sends. The RM Source **or RM Destination** **MUST** set the value of this element to the absolute URI
597 (conformant with RFC3986) of the Sequence that is being terminated.

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

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

601 /wsrm:TerminateSequence/{any}

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

604 /wsrm:TerminateSequence/@{any}

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

607 A TerminateSequenceResponse is sent in the body of a **response**-message **by an RM Destination** in
608 response to receipt of a TerminateSequence request message. It indicates that the **responding**
609 **party RM Destination** has terminated the Sequence.

610 The following exemplar defines the TerminateSequenceResponse syntax:

```
611 <wsrm:TerminateSequenceResponse ...>
```

```

612     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
613     ...
614 </wsrm:TerminateSequenceResponse>

```

615 The following describes the content model of the `TerminateSequence` element.

616 `/wsrm:TerminateSequenceResponse`

617 This element is sent in the body of a **response**-message **by an RM-Destination** in response to receipt of a
618 `TerminateSequence` request message. It indicates that the **responding partyRM-Destination** has
619 terminated the Sequence. The **responding partyRM-Destination** MUST NOT send this element as a
620 header block.

621 `/wsrm:TerminateSequenceResponse/wsrm:Identifier`

622 The **responding party (RMS or RMD)RM-Destination** MUST include this element in any
623 `TerminateSequenceResponse` message it sends. The **responding partyRM-Destination** MUST set the
624 value of this element to the absolute URI (conformant with RFC3986) of the Sequence that is being
625 terminated.

626 `/wsrm:TerminateSequenceResponse/wsrm:Identifier/@{any}`

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

629 `/wsrm:TerminateSequenceResponse/{any}`

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

632 `/wsrm:TerminateSequenceResponse/@{any}`

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

635 On receipt of a `TerminateSequence` message **the receiving party (RMS or RMD)an RM-Destination-**
636 **MUST** respond with a corresponding `TerminateSequenceResponse` message or generate a fault
637 `UnknownSequenceFault` if the Sequence is not known.

638 3.7 Sequences

639 The RM protocol uses a Sequence header block to track and manage the reliable transfer of messages.
640 The RM Source MUST include a `Sequence` header block in all messages for which reliable transfer is
641 REQUIRED. The RM Source MUST identify Sequences with unique Identifier elements and the RM
642 Source MUST assign each message within a Sequence a `MessageNumber` element that increments by 1
643 from an initial value of 1. These values are contained within a `Sequence` header block accompanying
644 each message being transferred in the context of a Sequence.

645 The RM Source MUST NOT include more than one `Sequence` header block in any message.

646 A following exemplar defines its syntax:

```

647 <wsrm:Sequence ...>
648   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
649   <wsrm:MessageNumber> wsrm:MessageNumberType </wsrm:MessageNumber>
650   ...
651 </wsrm:Sequence>

```

652 The following describes the content model of the `Sequence` header block.

653 /wsmr:Sequence

654 This protocol element associates the message in which it is contained with a previously established RM
655 Sequence. It contains the Sequence's unique identifier and the containing message's ordinal position
656 within that Sequence. The RM Destination MUST understand the Sequence header block. The RM
657 Source MUST assign a `mustUnderstand` attribute with a value 1/true (from the namespace
658 corresponding to the version of SOAP to which the Sequence SOAP header block is bound) to the
659 Sequence header block element.

660 /wsmr:Sequence/wsmr:Identifier

661 An RM Source that includes a Sequence header block in a SOAP envelope MUST include this element in
662 that header block. The RM Source MUST set the value of this element to the absolute URI (conformant
663 with RFC3986) that uniquely identifies the Sequence.

664 /wsmr:Sequence/wsmr:Identifier/@{any}

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

667 /wsmr:Sequence/wsmr:MessageNumber

668 The RM Source MUST include this element within any Sequence headers it creates. This element is of
669 type `MessageNumberType`. It represents the ordinal position of the message within a Sequence.
670 Sequence message numbers start at 1 and monotonically increase by 1 throughout the Sequence. See
671 Section 4.5 for Message Number Rollover fault.

672 /wsmr:Sequence/{any}

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

675 /wsmr:Sequence/@{any}

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

678 The following example illustrates a Sequence header block.

```
679 <wsmr:Sequence>  
680   <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>  
681   <wsmr:MessageNumber>10</wsmr:MessageNumber>  
682 </wsmr:Sequence>
```

683 3.8 Request Acknowledgement

684 The purpose of the `AckRequested` header block is to signal to the RM Destination that the RM Source is
685 requesting that a `SequenceAcknowledgement` be sent.

686 The RM Source MAY request an Acknowledgement Message from the RM Destination at any time by
687 transmitting an `AckRequested` header block independently or it MAY include an `AckRequested` header
688 block in any message targeted to the RM Destination. An RM Destination that Receives a message that
689 contains an `AckRequested` header block MUST send a message containing a
690 `SequenceAcknowledgement` header block to the `AcksTo` endpoint reference (see Section 3.4) for a
691 known Sequence or else generate an `UnknownSequence` fault. If a non-`mustUnderstand` fault occurs
692 when processing an RM header that was piggy-backed on another message, a fault MUST be generated,
693 but the processing of the original message MUST NOT be affected. It is RECOMMENDED that the RM

694 Destination return a `AcknowledgementRange` or `None` element instead of a `Nack` element (see Section
695 3.9).

696 The following exemplar defines its syntax:

```
697 <wsrm:AckRequested ...>  
698   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
699   ...  
700 </wsrm:AckRequested>
```

701 The following describes the content model of the `AckRequested` header block.

702 `/wsrm:AckRequested`

703 This element requests an Acknowledgement for the identified Sequence.

704 `/wsrm:AckRequested/wsrm:Identifier`

705 An RM Source that includes an `AckRequested` header block in a SOAP envelope MUST include this
706 element in that header block. The RM Source MUST set the value of this element to the absolute URI,
707 (conformant with RFC3986), that uniquely identifies the Sequence to which the request applies.

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

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

711 `/wsrm:AckRequested/{any}`

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

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

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

717 **3.9 Sequence Acknowledgement**

718 The RM Destination informs the RM Source of successful message receipt using a

719 `SequenceAcknowledgement` header block. The RM Destination MAY Transmit the

720 `SequenceAcknowledgement` header block independently or it MAY include the

721 `SequenceAcknowledgement` header block on any message targeted to the `AcksTo` EPR.

722 Acknowledgements can be explicitly requested using the `AckRequested` directive (see Section 3.8). If a

723 non-mustUnderstand fault occurs when processing an RM header that was piggy-backed on another

724 message, a fault MUST be generated, but the processing of the original message MUST NOT be

725 affected.

726 A RM Destination MAY include a `SequenceAcknowledgement` header block on any SOAP envelope

727 targeted to the endpoint referenced by the `AcksTo` EPR.

728 During creation of a Sequence the RM Source MAY specify the WS-Addressing anonymous IRI as the

729 address of the `AcksTo` EPR for that Sequence. When the RM Source specifies the WS-Addressing

730 anonymous IRI as the address of the `AcksTo` EPR, the RM Destination MUST Transmit any

731 `SequenceAcknowledgement` headers for the created Sequence in a SOAP envelope to be Transmitted

732 on the protocol binding-specific back-channel. Such a channel is provided by the context of a Received

733 message containing a SOAP envelope that contains a `Sequence` header block and/or an `AckRequested`

734 header block for that same Sequence identifier. When the RM Destination receives an `AckRequested`

735 header, and the `AckTo` EPR for that sequence is the WS-Addressing anonymous IRI, the RM Destination
736 SHOULD respond on the protocol binding-specific back-channel provided by the Received message
737 containing the `AckRequested` header block.

738 The following exemplar defines its syntax:

```
739 <wsrm:SequenceAcknowledgement ...>
740   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
741   [ [ [ <wsrm:AcknowledgementRange ...
742         Upper="wsrm:MessageNumberType"
743         Lower="wsrm:MessageNumberType" /> +
744         | <wsrm:None/> ]
745         <wsrm:Final/> ? ]
746         | <wsrm:Nack> wsrm:MessageNumberType </wsrm:Nack> + ]
747
748   ...
749 </wsrm:SequenceAcknowledgement>
```

750 The following describes the content model of the `SequenceAcknowledgement` header block.

751 `/wsrm:SequenceAcknowledgement`

752 This element contains the Sequence Acknowledgement information.

753 `/wsrm:SequenceAcknowledgement/wsrm:Identifier`

754 An RM Destination that includes a `SequenceAcknowledgement` header block in a SOAP envelope
755 MUST include this element in that header block. The RM Destination MUST set the value of this element
756 to the absolute URI (conformant with RFC3986) that uniquely identifies the Sequence. The RM
757 Destination MUST NOT include multiple `SequenceAcknowledgement` header blocks that share the
758 same value for `Identifier` within the same SOAP envelope.

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

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

762 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange`

763 The RM Destination MAY include one or more instances of this element within a
764 `SequenceAcknowledgement` header block. It contains a range of Sequence message numbers
765 successfully accepted by the RM Destination. The ranges MUST NOT overlap. The RM Destination
766 MUST NOT include this element if a sibling `Nack` or `None` element is also present as a child of
767 `SequenceAcknowledgement`.

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

769 The RM Destination MUST set the value of this attribute equal to the message number of the highest
770 contiguous message in a Sequence range accepted by the RM Destination.

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

772 The RM Destination MUST set the value of this attribute equal to the message number of the lowest
773 contiguous message in a Sequence range accepted by the RM Destination.

774 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@{any}`

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

777 `/wsrm:SequenceAcknowledgement/wsrm:None`

778 The RM Destination MUST include this element within a `SequenceAcknowledgement` header block if
779 the RM Destination has not accepted any messages for the specified Sequence. The RM Destination
780 MUST NOT include this element if a sibling `AcknowledgementRange` or `Nack` element is also present
781 as a child of the `SequenceAcknowledgement`.

782 `/wsrm:SequenceAcknowledgement/wsrm:Final`

783 The RM Destination MAY include this element within a `SequenceAcknowledgement` header block. This
784 element indicates that the RM Destination is not receiving new messages for the specified Sequence. The
785 RM Source can be assured that the ranges of messages acknowledged by this
786 `SequenceAcknowledgement` header block will not change in the future. The RM Destination MUST
787 include this element when the Sequence is closed. The RM Destination MUST NOT include this element
788 when sending a `Nack`; it can only be used when sending `AcknowledgementRange` elements or a `None`.

789 `/wsrm:SequenceAcknowledgement/wsrm:Nack`

790 The RM Destination MAY include this element within a `SequenceAcknowledgement` header block. If
791 used, the RM Destination MUST set the value of this element to a `MessageNumberType` representing
792 the `MessageNumber` of an unreceived message in a Sequence. The RM Destination MUST NOT include
793 a `Nack` element if a sibling `AcknowledgementRange` or `None` element is also present as a child of
794 `SequenceAcknowledgement`. Upon the receipt of a `Nack`, an RM Source SHOULD retransmit the
795 message identified by the `Nack`. The RM Destination MUST NOT issue a `SequenceAcknowledgement`
796 containing a `Nack` for a message that it has previously acknowledged within a
797 `AcknowledgementRange`. The RM Source SHOULD ignore a `SequenceAcknowledgement` containing
798 a `Nack` for a message that has previously been acknowledged within a `AcknowledgementRange`.

799 `/wsrm:SequenceAcknowledgement/{any}`

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

802 `/wsrm:SequenceAcknowledgement/@{any}`

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

805 The following examples illustrate `SequenceAcknowledgement` elements:

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

```
807 <wsrm:SequenceAcknowledgement>  
808   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
809   <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
810 </wsrm:SequenceAcknowledgement>
```

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

```
813 <wsrm:SequenceAcknowledgement>  
814   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
815   <wsrm:AcknowledgementRange Upper="2" Lower="1"/>  
816   <wsrm:AcknowledgementRange Upper="6" Lower="4"/>  
817   <wsrm:AcknowledgementRange Upper="10" Lower="8"/>  
818 </wsrm:SequenceAcknowledgement>
```

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

```
820 <wsrm:SequenceAcknowledgement>  
821   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
```

822
823

```
<wsrm:Nack>3</wsrm:Nack>  
</wsrm:SequenceAcknowledgement>
```

824 4 Faults

825 Faults for the `CreateSequence` message exchange are treated as defined in WS-Addressing. Create
826 Sequence Refused is a possible fault reply for this operation. Unknown Sequence is a fault generated by
827 Endpoints when messages carrying RM header blocks targeted at unrecognized or terminated Sequences
828 are detected. WSRM Required is a fault generated an RM Destination that requires the use of WS-RM on
829 a Received message that did not use the protocol. All other faults in this section relate to known
830 Sequences. Destinations that generate faults related to known sequences SHOULD transmit those faults.
831 If transmitted, such faults MUST be transmitted to the same [destination] as Acknowledgement
832 messages.

833 Entities that generate WS-ReliableMessaging faults MUST include as the [action] property the default fault
834 action IRI defined below. The value from the W3C Recommendation is below for informational purposes:

```
835 http://docs.oasis-open.org/ws-rx/wsrn/200608/fault
```

836 The faults defined in this section are generated if the condition stated in the preamble is met. Fault
837 handling rules are defined in section 6 of WS-Addressing SOAP Binding.

838 The definitions of faults use the following properties:

839 [Code] The fault code.

840 [Subcode] The fault subcode.

841 [Reason] The English language reason element.

842 [Detail] The detail element(s). If absent, no detail element is defined for the fault. If more than one detail
843 element is defined for a fault, implementations MUST include the elements in the order that they are
844 specified.

845 Entities that generate WS-ReliableMessaging faults MUST set the [Code] property to either "Sender" or
846 "Receiver". These properties are serialized into text XML as follows:

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

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

```
848 <S:Envelope>  
849   <S:Header>  
850     <wsa:Action>  
851       http://docs.oasis-open.org/ws-rx/wsrn/200608/fault  
852     </wsa:Action>  
853     <!-- Headers elided for brevity. -->  
854   </S:Header>  
855   <S:Body>  
856     <S:Fault>  
857       <S:Code>  
858         <S:Value> [Code] </S:Value>  
859         <S:Subcode>  
860           <S:Value> [Subcode] </S:Value>  
861         </S:Subcode>  
862       </S:Code>  
863       <S:Reason>  
864         <S:Text xml:lang="en"> [Reason] </S:Text>  
865       </S:Reason>  
866     <S:Detail>
```



```

867     [Detail]
868     ...
869     </S:Detail>
870     </S:Fault>
871     </S:Body>
872 </S:Envelope>

```

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

```

875 <S11:Envelope>
876   <S11:Header>
877     <wsrm:SequenceFault>
878       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
879       <wsrm:Detail> [Detail] </wsrm:Detail>
880       ...
881     </wsrm:SequenceFault>
882     <!-- Headers elided for brevity. -->
883   </S11:Header>
884   <S11:Body>
885     <S11:Fault>
886       <faultcode> [Code] </faultcode>
887       <faultstring> [Reason] </faultstring>
888     </S11:Fault>
889   </S11:Body>
890 </S11:Envelope>

```

891 The properties bind to a SOAP 1.1 fault as follows when the fault is generated as a result of processing a
892 `CreateSequence` request message:

```

893 <S11:Envelope>
894   <S11:Body>
895     <S11:Fault>
896       <faultcode> [Subcode] </faultcode>
897       <faultstring> [Reason] </faultstring>
898     </S11:Fault>
899   </S11:Body>
900 </S11:Envelope>

```

901 4.1 SequenceFault Element

902 The purpose of the `SequenceFault` element is to carry the specific details of a fault generated during
903 the reliable messaging specific processing of a message belonging to a Sequence. WS-
904 ReliableMessaging nodes MUST use the `SequenceFault` container only in conjunction with the SOAP
905 1.1 fault mechanism. WS-ReliableMessaging nodes MUST NOT use the `SequenceFault` container in
906 conjunction with the SOAP 1.2 binding.

907 The following exemplar defines its syntax:

```

908 <wsrm:SequenceFault ...>
909   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
910   <wsrm:Detail> ... </wsrm:Detail> ?
911   ...
912 </wsrm:SequenceFault>

```

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

914 `/wsrm:SequenceFault`

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

916 /wsm:SequenceFault/wsm:FaultCode
 917 WS-ReliableMessaging nodes that generate a `SequenceFault` MUST set the value of this element to a
 918 qualified name from the set of fault [Subcodes] defined below.

919 /wsm:SequenceFault/wsm:Detail
 920 This element, if present, carries application specific error information related to the fault being described.

921 /wsm:SequenceFault/wsm:Detail/{any}
 922 The application specific error information related to the fault being described.

923 /wsm:SequenceFault/wsm:Detail/@{any}
 924 The application specific error information related to the fault being described.

925 /wsm:SequenceFault/{any}
 926 This is an extensibility mechanism to allow different (extensible) types of information, based on a schema,
 927 to be passed.

928 /wsm:SequenceFault/@{any}
 929 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the
 930 element.

931 4.2 Sequence Terminated

932 The Endpoint that generates this fault SHOULD make every reasonable effort to notify the corresponding
 933 Endpoint of this decision.

934 Properties:

935 [Code] Sender or Receiver
 936 [Subcode] wsm:SequenceTerminated
 937 [Reason] The Sequence has been terminated due to an unrecoverable error.
 938 [Detail]

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

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Source or RM Destination.	Encountering an unrecoverable condition or detection of violation of the protocol.	Sequence termination.	MUST terminate the Sequence if not otherwise terminated.

940 4.3 Unknown Sequence

941 Properties:

942 [Code] Sender
 943 [Subcode] wsm:UnknownSequence

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

945 [Detail]

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

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Source or RM Destination.	In response to a message containing an unknown or terminated Sequence identifier.	None.	MUST terminate the Sequence if not otherwise terminated.

947 4.4 Invalid Acknowledgement

948 An example of when this fault is generated is when a message is Received by the RM Source containing
949 a SequenceAcknowledgement covering messages that have not been sent.

950 [Code] Sender

951 [Subcode] wsrn:InvalidAcknowledgement

952 [Reason] The SequenceAcknowledgement violates the cumulative Acknowledgement invariant.

953 [Detail]

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

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Source.	In response to a SequenceAcknowledgement that violate the invariants stated in 2.3 or any of the requirements in 3.9 about valid combinations of AckRange, Nack and None in a single SequenceAcknowledgement element or with respect to already Received such elements.	Unspecified.	Unspecified.

955 4.5 Message Number Rollover

956 If the condition listed below is reached, the RM Destination MUST generate this fault.

957 Properties:

958 [Code] Sender

959 [Subcode] wsrn:MessageNumberRollover

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

961 [Detail]

```
962 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
963 <wsrm:MaxMessageNumber> wsrm:MessageNumberType </wsrm:MaxMessageNumber>
```

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Destination.	Message number in /wsrm:Sequence/wsrm:MessageNumber of a Received message exceeds the internal limitations of an RM Destination or reaches the maximum value of 9,223,372,036,854,775,807.	RM Destination SHOULD continue to accept undelivered messages until the Sequence is closed or terminated.	RM Source SHOULD continue to retransmit undelivered messages until the Sequence is closed or terminated.

964 4.6 Create Sequence Refused

965 Properties:

966 [Code] Sender or Receiver

967 [Subcode] wsrm:CreateSequenceRefused

968 [Reason] The Create Sequence request has been refused by the RM Destination.

969 [Detail]

```
970 xs:any
```

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Destination.	In response to a CreateSequence message when the RM Destination does not wish to create a new Sequence.	Unspecified.	Sequence terminated.

971 4.7 Sequence Closed

972 This fault is generated by an RM Destination to indicate that the specified Sequence has been closed.

973 This fault MUST be generated when an RM Destination is asked to accept a message for a Sequence that
974 is closed.

975 Properties:

976 [Code] Sender

977 [Subcode] wsrm:SequenceClosed

978 [Reason] The Sequence is closed and can not accept new messages.

979 [Detail]

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

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Destination.	In response to a message that belongs to a Sequence that is already closed.	Unspecified.	Sequence closed.

981 **4.8 WSRM Required**

982 If an RM Destination requires the use of WS-RM, this fault is generated when it Receives an incoming
983 message that did not use this protocol.

984 Properties:

985 [Code] Sender

986 [Subcode] wsrm:WSRMRequired

987 [Reason] The RM Destination requires the use of WSRM.

988 [Detail]

989 `xs:any`

990 **5 Security Threats and Countermeasures**

991 This specification considers two sets of security requirements, those of the applications that use the WS-
992 RM protocol and those of the protocol itself.

993 This specification makes no assumptions about the security requirements of the applications that use WS-
994 RM. However, once those requirements have been satisfied within a given operational context, the
995 addition of WS-RM to this operational context should not undermine the fulfillment of those requirements;
996 the use of WS-RM should not create additional attack vectors within an otherwise secure system.

997 There are many other security concerns that one may need to consider when implementing or using this
998 protocol. The material below should not be considered as a "check list". Implementers and users of this
999 protocol are urged to perform a security analysis to determine their particular threat profile and the
1000 appropriate responses to those threats.

1001 Implementers are also advised that there is a core tension between security and reliable messaging that
1002 can be problematic if not addressed by implementations; one aspect of security is to prevent message
1003 replay but one of the invariants of this protocol is to resend messages until they are acknowledged.
1004 Consequently, if the security sub-system processes a message but a failure occurs before the reliable
1005 messaging sub-system Receives that message, then it is possible (and likely) that the security sub-system
1006 will treat subsequent copies as replays and discard them. At the same time, the reliable messaging sub-
1007 system will likely continue to expect and even solicit the missing message(s). Care should be taken to
1008 avoid and prevent this condition.

1009 **5.1 Threats and Countermeasures**

1010 The primary security requirement of this protocol is to protect the specified semantics and protocol
1011 invariants against various threats. The following sections describe several threats to the integrity and
1012 operation of this protocol and provide some general outlines of countermeasures to those threats.
1013 Implementers and users of this protocol should keep in mind that all threats are not necessarily applicable
1014 to all operational contexts.

1015 **5.1.1 Integrity Threats**

1016 In general, any mechanism which allows an attacker to alter the information in a Sequence Traffic
1017 Message, Sequence Lifecycle Message, Acknowledgement Messages, Acknowledgement Request, or
1018 Sequence-related fault, or which allows an attacker to alter the correlation of a RM Protocol Header Block
1019 to its intended message represents a threat to the WS-RM protocol.

1020 For example, if an attacker is able to swap `Sequence` headers on messages in transit between the RM
1021 Source and RM Destination then they have undermined the implementation's ability to guarantee the first
1022 invariant described in Section 2.3. The result is that there is no way of guaranteeing that messages will be
1023 Delivered to the Application Destination in the same order that they were sent by the Application Source.

1024 **5.1.1.1 Countermeasures**

1025 Integrity threats are generally countered via the use of digital signatures some level of the communication
1026 protocol stack. Note that, in order to counter header swapping attacks, the signature SHOULD include
1027 both the SOAP body and any relevant SOAP headers (e.g. `Sequence` header). Because some headers
1028 (`AckRequested`, `SequenceAcknowledgement`) are independent of the body of the SOAP message in which
1029 they occur, implementations MUST allow for signatures that cover only these headers.

1030 **5.1.2 Resource Consumption Threats**

1031 The creation of a Sequence with an RM Destination consumes various resources on the systems used to
1032 implement that RM Destination. These resources can include network connections, database tables,
1033 message queues, etc. This behavior can be exploited to conduct denial of service attacks against an RM
1034 Destination. For example, a simple attack is to repeatedly send `CreateSequence` messages to an RM
1035 Destination. Another attack is to create a Sequence for a service that is known to require in-order
1036 message Delivery and use this Sequence to send a stream of very large messages to that service,
1037 making sure to omit message number “1” from that stream.

1038 **5.1.2.1 Countermeasures**

1039 There are a number of countermeasures against the described resource consumption threats. The
1040 technique advocated by this specification is for the RM Destination to restrict the ability to create a
1041 Sequence to a specific set of entities/principals. This reduces the number of potential attackers and, in
1042 some cases, allows the identity of any attackers to be determined.

1043 The ability to restrict Sequence creation depends, in turn, upon the RM Destination's ability identify and
1044 authenticate the RM Source that issued the `CreateSequence` message.

1045 **5.1.3 Sequence Spoofing Threats**

1046 Sequence spoofing is a class of threats in which the attacker uses knowledge of the `Identifier` for a
1047 particular Sequence to forge Sequence Lifecycle or Traffic Messages. For example the attacker creates a
1048 fake `TerminateSequence` message that references the target Sequence and sends this message to the
1049 appropriate RM Destination. Some sequence spoofing attacks also require up-to-date knowledge of the
1050 current `MessageNumber` for their target Sequence.

1051 In general any Sequence Lifecycle Message, RM Protocol Header Block, or sequence-correlated SOAP
1052 fault (e.g. `InvalidAcknowledgement`) can be used by someone with knowledge of the Sequence identifier
1053 to attack the Sequence. These attacks are “two-way” in that an attacker may choose to target the RM
1054 Source by, for example, inserting a fake `SequenceAcknowledgement` header into a message that it sends
1055 to the `AcksTo` EPR of an RM Source.

1056 **5.1.3.1 Sequence Hijacking**

1057 Sequence hijacking is a specific case of a sequence spoofing attack. The attacker attempts to inject
1058 Sequence Traffic Messages into an existing Sequence by inserting fake `Sequence` headers into those
1059 messages.

1060 Note that “sequence hijacking” should not be equated with “security session hijacking”. Although a
1061 Sequence may be bound to some form of a security session in order to counter the threats described in
1062 this section, applications MUST NOT rely on WS-RM-related information to make determinations about
1063 the identity of the entity that created a message; applications SHOULD rely only upon information that is
1064 established by the security infrastructure to make such determinations. Failure to observe this rule
1065 creates, among other problems, a situation in which the absence of WS-RM may deprive an application of
1066 the ability to authenticate its peers even though the necessary security processing has taken place.

1067 **5.1.3.2 Countermeasures**

1068 There are a number of countermeasures against sequence spoofing threats. The technique advocated by
1069 this specification is to consider the Sequence to be a shared resource that is jointly owned by the RM

1070 Source that initiated its creation (i.e. that sent the `CreateSequence` message) and the RM Destination that
1071 serves as its terminus (i.e. that sent the `CreateSequenceResponse` message). To counter sequence
1072 spoofing attempts the RM Destination SHOULD ensure that every message or fault that it Receives that
1073 refers to a particular Sequence originated from the RM Source that jointly owns the referenced Sequence.
1074 For its part the RM Source SHOULD ensure that every message or fault that it Receives that refers to a
1075 particular Sequence originated from the RM Destination that jointly owns the referenced Sequence.

1076 For the RM Destination to be able to identify its sequence peer it MUST be able to identify and
1077 authenticate the entity that sent the `CreateSequence` message. Similarly for the RM Source to identify its
1078 sequence peer it MUST be able to identify and authenticate the entity that sent the
1079 `CreateSequenceResponse` message. For either the RM Destination or the RM Source to determine if a
1080 message was sent by its sequence peer it MUST be able to identify and authenticate the initiator of that
1081 message and, if necessary, correlate this identity with the sequence peer identity established at sequence
1082 creation time.

1083 **5.2 Security Solutions and Technologies**

1084 The security threats described in the previous sections are neither new nor unique. The solutions that
1085 have been developed to secure other SOAP-based protocols can be used to secure WS-RM as well. This
1086 section maps the facilities provided by common web services security solutions against countermeasures
1087 described in the previous sections.

1088 Before continuing this discussion, however, some examination of the underlying requirements of the
1089 previously described countermeasures is necessary. Specifically it should be noted that the technique
1090 described in Section 5.1.2.1 has two components. Firstly, the RM Destination identifies and authenticates
1091 the issuer of a `CreateSequence` message. Secondly, the RM Destination performs an authorization check
1092 against this authenticated identity and determines if the RM Source is permitted to create Sequences with
1093 the RM Destination. Since the facilities for performing this authorization check (runtime infrastructure,
1094 policy frameworks, etc.) lie completely within the domain of individual implementations, any discussion of
1095 such facilities is considered to be beyond the scope of this specification.

1096 **5.2.1 Transport Layer Security**

1097 This section describes how the facilities provided by SSL/TLS [RFC 4346] can be used to implement the
1098 countermeasures described in the previous sections. The use of SSL/TLS is subject to the constraints
1099 defined in Section 4 of the Basic Security Profile 1.0 [BSP 1.0].

1100 The description provided here is general in nature and is not intended to serve as a complete definition on
1101 the use of SSL/TLS to protect WS-RM. In order to interoperate implementations need to agree on the
1102 choice of features as well as the manner in which they will be used. The mechanisms described in the
1103 Web Services Security Policy Language [SecurityPolicy] MAY be used by services to describe the
1104 requirements and constraints of the use of SSL/TLS.

1105 **5.2.1.1 Model**

1106 The basic model for using SSL/TLS is as follows:

- 1107 1. The RM Source establishes an SSL/TLS session with the RM Destination.
- 1108 2. The RM Source uses this SSL/TLS session to send a `CreateSequence` message to the RM
1109 Destination.

- 1110 3. The RM Destination establishes an SSL/TLS session with the RM Source and sends an
1111 asynchronous `CreateSequenceResponse` using this session. Alternately it may respond with a
1112 synchronous `CreateSequenceResponse` using the session established in (1).
- 1113 4. For the lifetime of the Sequence the RM Source uses the SSL/TLS session from (1) to Transmit
1114 any and all messages or faults that refer to that Sequence.
- 1115 5. For the lifetime of the Sequence the RM Destination either uses the SSL/TLS session established
1116 in (3) to Transmit any and all messages or faults that refer to that Sequence or, for synchronous
1117 exchanges, the RM Destination uses the SSL/TLS session established in (1).

1118 5.2.1.2 Countermeasure Implementation

1119 Used in its simplest fashion (without relying upon any authentication mechanisms), SSL/TLS provides the
1120 necessary integrity qualities to counter the threats described in Section 5.1.1. Note, however, that the
1121 nature of SSL/TLS limits the scope of this integrity protection to a single transport level session. If
1122 SSL/TLS is the only mechanism used to provide integrity, any intermediaries between the RM Source and
1123 the RM Destination MUST be trusted to preserve the integrity of the messages that flow through them.

1124 As noted, the technique described in Sections 5.1.2.1 involves the use of authentication. This specification
1125 advocates either of two mechanisms for authenticating entities using SSL/TLS. In both of these methods
1126 the SSL/TLS server (the party accepting the SSL/TLS connection) authenticates itself to the SSL/TLS
1127 client using an X.509 certificate that is exchanged during the SSL/TLS handshake.

- 1128 • **HTTP Basic Authentication:** This method of authentication presupposes that a SOAP/HTTP
1129 binding is being used as part of the protocol stack beneath WS-RM. Subsequent to the
1130 establishment of the SSL/TLS session, the sending party authenticates itself to the receiving party
1131 using HTTP Basic Authentication [RFC 2617]. For example, a RM Source might authenticate itself
1132 to a RM Destination (e.g. when transmitting a Sequence Traffic Message) using BasicAuth.
1133 Similarly the RM Destination might authenticate itself to the RM Source (e.g. when sending an
1134 Acknowledgement) using BasicAuth.
- 1135 • **SSL/TLS Client Authentication:** In this method of authentication, the party initiating the
1136 connection authenticates itself to the party accepting the connection using an X.509 certificate
1137 that is exchanged during the SSL/TLS handshake.

1138 To implement the countermeasures described in section 5.1.2.1 the RM Source must authenticate itself
1139 using one the above mechanisms. The authenticated identity can then be used to determine if the RM
1140 Source is authorized to create a Sequence with the RM Destination.

1141 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring
1142 an RM node's Sequence peer to be equivalent to their SSL/TLS session peer. This allows the
1143 authorization decisions described in section 5.1.3.2 to be based on SSL/TLS session identity rather than
1144 on authentication information. For example, an RM Destination can determine that a Sequence Traffic
1145 Message rightfully belongs to its referenced Sequence if that message arrived over the same SSL/TLS
1146 session that was used to carry the `CreateSequence` message for that Sequence. Note that requiring a
1147 one-to-one relationship between SSL/TLS session peer and Sequence peer constrains the lifetime of a
1148 SSL/TLS-protected Sequence to be less than or equal to the lifetime of the SSL/TLS session that is used
1149 to protect that Sequence.

1150 This specification does not preclude the use of other methods of using SSL/TLS to implement the
1151 countermeasures (such as associating specific authentication information with a Sequence) although such
1152 methods are not covered by this document.

1153 Issues specific to the life-cycle management of SSL/TLS sessions (such as the resumption of a SSL/TLS
1154 session) are outside the scope of this specification.

1155 **5.2.2 SOAP Message Security**

1156 The mechanisms described in WS-Security may be used in various ways to implement the
1157 countermeasures described in the previous sections. This specification advocates using the protocol
1158 described by WS-SecureConversation [[SecureConversation](#)] (optionally in conjunction with WS-Trust
1159 [[Trust](#)]) as a mechanism for protecting Sequences. The use of WS-Security (as an underlying component
1160 of WS-SecureConversation) is subject to the constraints defined in the Basic Security Profile 1.0.

1161 The description provided here is general in nature and is not intended to serve as a complete definition on
1162 the use of WS-SecureConversation/WS-Trust to protect WS-RM. In order to interoperate implementations
1163 need to agree on the choice of features as well as the manner in which they will be used. The
1164 mechanisms described in the Web Services Security Policy Language MAY be used by services to
1165 describe the requirements and constraints of the use of WS-SecureConversation.

1166 **5.2.2.1 Model**

1167 The basic model for using WS-SecureConversation is as follows:

- 1168 1. The RM Source and the RM Destination create a WS-SecureConversation security context. This
1169 may involve the participation of third parties such as a security token service. The tokens
1170 exchanged may contain authentication claims (e.g. X.509 certificates or Kerberos service tickets).
- 1171 2. During the `CreateSequence` exchange, the RM Source SHOULD explicitly identify the security
1172 context that will be used to protect the Sequence. This is done so that, in cases where the
1173 `CreateSequence` message is signed by more than one security context, the RM Source can
1174 indicate which security context should be used to protect the newly created Sequence.
- 1175 3. For the lifetime of the Sequence the RM Source and the RM Destination use the session key(s)
1176 associated with the security context to sign (as defined by WS-Security) at least the body and any
1177 relevant WS-RM-defined headers of any and all messages or faults that refer to that Sequence.

1178 **5.2.2.2 Countermeasure Implementation**

1179 Without relying upon any authentication information, the per-message signatures provide the necessary
1180 integrity qualities to counter the threats described in Section 5.1.1.

1181 To implement the countermeasures described in section 5.1.2.1 some mutually agreed upon form of
1182 authentication claims must be provided by the RM Source to the RM Destination during the establishment
1183 of the Security Context. These claims can then be used to determine if the RM Source is authorized to
1184 create a Sequence with the RM Destination.

1185 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring
1186 an RM node's Sequence peer to be equivalent to their security context session peer. This allows the
1187 authorization decisions described in section 5.1.3.2 to be based on the identity of the message's security
1188 context rather than on any authentication claims that may have been established during security context
1189 initiation. Note that other methods of using WS-SecureConversation to implement the countermeasures
1190 (such as associating specific authentication claims to a Sequence) are possible but not covered by this
1191 document.

1192 As with transport security, the requisite equivalence of a security context peer and with a Sequence peer
1193 limits the lifetime of a Sequence to the lifetime of the protecting security context. Unlike transport security,

1194 the association between a Sequence and its protecting security context cannot always be established
1195 implicitly at Sequence creation time. This is due to the fact that the `CreateSequence` and
1196 `CreateSequenceResponse` messages may be signed by more than one security context.

1197 Issues specific to the life-cycle management of WS-SecureConversation security contexts (such as
1198 amending or renewing contexts) are outside the scope of this specification.

1199 6 Securing Sequences

1200 As noted in Section 5, the RM Source and RM Destination should be able to protect their shared
1201 Sequences against the threat of Sequence Spoofing attacks. There are a number of OPTIONAL means of
1202 achieving this objective depending upon the underlying security infrastructure.

1203 6.1 Securing Sequences Using WS-Security

1204 One mechanism for protecting a Sequence is to include a security token using a
1205 `wsse:SecurityTokenReference` element from WS-Security (see section 9 in WS-
1206 SecureConversation) in the `CreateSequence` element. This establishes an association between the
1207 created (and, if present, offered) Sequence(s) and the referenced security token, such that the RM Source
1208 and Destination MUST use the security token as the basis for authorization of all subsequent interactions
1209 related to the Sequence(s). The `wsse:SecurityTokenReference` explicitly identifies the token as
1210 there may be more than one token on a `CreateSequence` message or inferred from the communication
1211 context (e.g. transport protection).

1212 It is RECOMMENDED that a message independent referencing mechanism be used to identify the token,
1213 if the token being referenced supports such mechanism.

1214 The following exemplar defines the `CreateSequence` syntax when extended to include a
1215 `wsse:SecurityTokenReference`:

```
1216 <wsrm:CreateSequence ...>  
1217   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>  
1218   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1219   <wsrm:Offer ...>  
1220     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
1221     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>  
1222     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1223     <wsrm:IncompleteSequenceBehavior>  
1224       wsrml:IncompleteSequenceBehaviorType  
1225     </wsrm:IncompleteSequenceBehavior> ?  
1226     ...  
1227   </wsrm:Offer> ?  
1228   ...  
1229   <wsse:SecurityTokenReference>  
1230     ...  
1231   </wsse:SecurityTokenReference> ?  
1232   ...  
1233 </wsrm:CreateSequence>
```

1234 The following describes the content model of the additional `CreateSequence` elements.

1235 `/wsrm:CreateSequence/wsse:SecurityTokenReference`

1236 This element uses the extensibility mechanism defined for the `CreateSequence` element (defined in
1237 section 3.4) to communicate an explicit reference to the security token, using a
1238 `wsse:SecurityTokenReference` as documented in WS-Security, that the RM Source and Destination
1239 MUST use to authorize messages for the created (and, if present, the offered) Sequence(s). All
1240 subsequent messages related to the created (and, if present, the offered) Sequence(s) MUST
1241 demonstrate proof-of-possession of the secret associated with the token (e.g., by using or deriving from a
1242 private or secret key).

1243 When a RM Source transmits a `CreateSequence` that has been extended to include a
1244 `wsse:SecurityTokenReference` it SHOULD ensure that the RM Destination both understands and
1245 will conform to the requirements listed above. In order to achieve this, the RM Source SHOULD include

1246 the `UsesSequenceSTR` element as a SOAP header block within the `CreateSequence` message. This
1247 element MUST include a `soap:mustUnderstand` attribute with a value of 'true'. Thus the RM Source
1248 can be assured that a RM Destination that responds with a `CreateSequenceResponse` understands
1249 and conforms with the requirements listed above. Note that an RM Destination understanding this header
1250 does not mean that it has processed and understood any WS-Security headers, the fault behavior defined
1251 in WS-Security still applies.

1252 The following exemplar defines the `UsesSequenceSTR` syntax:

```
1253 <wsm:UsesSequenceSTR ... />
```

1254 The following describes the content model of the `UsesSequenceSTR` header block.

1255 `/wsm:UsesSequenceSTR`

1256 This element SHOULD be included as a SOAP header block in `CreateSequence` messages that use the
1257 extensibility mechanism described above in this section. The `soap:mustUnderstand` attribute value
1258 MUST be 'true'. The receiving RM Destination MUST understand and correctly implement the extension
1259 described above or else generate a `soap:MustUnderstand` fault, thus aborting the requested
1260 Sequence creation.

1261 The following is an example of a `CreateSequence` message using the
1262 `wsse:SecurityTokenReference` extension and the `UsesSequenceSTR` header block:

```
1263 <soap:Envelope ...>  
1264   <soap:Header>  
1265     ...  
1266     <wsm:UsesSequenceSTR soap:mustUnderstand='true' />  
1267     ...  
1268   </soap:Header>  
1269   <soap:Body>  
1270     <wsm:CreateSequence>  
1271       <wsm:AcksTo>  
1272         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>  
1273       </wsm:AcksTo>  
1274       <wsse:SecurityTokenReference>  
1275         ...  
1276       </wsse:SecurityTokenReference>  
1277     </wsm:CreateSequence>  
1278   </soap:Body>  
1279 </soap:Envelope>
```

1280 6.2 Securing Sequences Using SSL/TLS

1281 One mechanism for protecting a Sequence is to bind the Sequence to the underlying SSL/TLS session(s).
1282 The RM Source indicates to the RM Destination that a Sequence is to be bound to the underlying
1283 SSL/TLS session(s) via the `UsesSequenceSSL` header block. If the RM Source wishes to bind a
1284 Sequence to the underlying SSL/TLS sessions(s) it MUST include the `UsesSequenceSSL` element as a
1285 SOAP header block within the `CreateSequence` message.

1286 The following exemplar defines the `UsesSequenceSSL` syntax:

```
1287 <wsm:UsesSequenceSSL soap:mustUnderstand="true" ... />
```

1288 The following describes the content model of the `UsesSequenceSSL` header block.

1289 `/wsm:UsesSequenceSSL`

1290 The RM Source MAY include this element as a SOAP header block of a `CreateSequence` message to
1291 indicate to the RM Destination that the resulting Sequence is to be bound to the SSL/TLS session that was

1292 used to carry the `CreateSequence` message. If included, the RM Source MUST mark this header with a
1293 `soap:mustUnderstand` attribute with a value of 'true'. The receiving RM Destination MUST understand
1294 and correctly implement the functionality described in Section 5.2.1 or else generate a
1295 `soap:MustUnderstand` fault, thus aborting the requested Sequence creation.

1296 Note that the use inclusion of the above header by the RM Source implies that all Sequence-related
1297 information (Sequence Lifecycle or Acknowledgment messages or Sequence-related faults) flowing from
1298 the RM Destination to the RM Source will be bound to the SSL/TLS session that is used to carry the
1299 `CreateSequenceResponse` message.

1300 **7 References**

1301 **7.1 Normative**

1302 **[KEYWORDS]**

1303 S. Bradner, "[Key words for use in RFCs to Indicate Requirement Levels](#)," RFC 2119, Harvard University,
1304 March 1997

1305 <http://www.ietf.org/rfc/rfc2119.txt>

1306 **[WS-RM Policy]**

1307 OASIS WS-RX Technical Committee Draft, "[Web Services ReliableMessaging Policy Assertion\(WS-RM
1308 Policy\)](#)" October 2006

1309 <http://docs.oasis-open.org/ws-rx/wsrmp/200608/wsrmp-1.1-spec-wd-11.pdf>

1310 **[SOAP 1.1]**

1311 W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.

1312 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

1313 **[SOAP 1.2]**

1314 W3C Recommendation, "[SOAP Version 1.2 Part 1: Messaging Framework](#)" June 2003.

1315 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

1316 **[URI]**

1317 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#)," RFC 3986,
1318 MIT/LCS, U.C. Irvine, Xerox Corporation, January 2005.

1319 <http://ietf.org/rfc/rfc3986>

1320 **[UUID]**

1321 P. Leach, M. Mealling, R. Salz, "[A Universally Unique IDentifier \(UUID\) URN Namespace](#)," RFC 4122,
1322 Microsoft, Refactored Networks - LLC, DataPower Technology Inc, July 2005

1323 <http://www.ietf.org/rfc/rfc4122.txt>

1324 **[XML]**

1325 W3C Recommendation, "[Extensible Markup Language \(XML\) 1.0 \(Fourth Edition\)](#)", September 2006.

1326 <http://www.w3.org/TR/REC-xml/>

1327 **[XML-ns]**

1328 W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.

1329 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>

1330 **[XML-Schema Part1]**

1331 W3C Recommendation, "[XML Schema Part 1: Structures](#)," October 2004.

1332 <http://www.w3.org/TR/xmlschema-1/>

1333 **[XML-Schema Part2]**

1334 W3C Recommendation, "XML Schema Part 2: Datatypes," October 2004.

1335 <http://www.w3.org/TR/xmlschema-2/>

1336 **[XPath 1.0]**

1337 W3C Recommendation, "XML Path Language (XPath) Version 1.0," 16 November 1999.

1338 <http://www.w3.org/TR/xpath>

1339 **[WSDL 1.1]**

1340 W3C Note, "Web Services Description Language (WSDL 1.1)," 15 March 2001.

1341 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

1342 **[WS-Addressing]**

1343 W3C Recommendation, "Web Services Addressing 1.0 - Core", May 2006.

1344 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>

1345 W3C Recommendation, "Web Services Addressing 1.0 – SOAP Binding", May 2006.

1346 <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>

1347 **7.2 Non-Normative**

1348 **[BSP 1.0]**

1349 WS-I Working Group Draft. "Basic Security Profile Version 1.0," August 2006

1350 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>

1351 **[RDDL 2.0]**

1352 Jonathan Borden, Tim Bray, eds. "Resource Directory Description Language (RDDL) 2.0," January 2004

1353 <http://www.openhealth.org/RDDL/20040118/rddl-20040118.html>

1354 **[RFC 2617]**

1355 J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Loutonen, L. Stewart, "HTTP
1356 Authentication: Basic and Digest Access Authentication," June 1999.

1357 <http://www.ietf.org/rfc/rfc2617.txt>

1358 **[RFC 4346]**

1359 T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1," April 2006.

1360 <http://www.ietf.org/rfc/rfc4346.txt>

1361 **[WS-Policy]**

1362 W3C Member Submission, "Web Services Policy Framework (WS-Policy)," April 2006.

1363 <http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/>

1364 **[WS-PolicyAttachment]**

1365 W3C Member Submission, "Web Services Policy Attachment (WS-PolicyAttachment)," April 2006.

1366 [http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-
1367 20060425/](http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-20060425/)

1368 **[WS-Security]**

1369 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:
1370 SOAP Message Security 1.0 (WS-Security 2004)", OASIS Standard 200401, March 2004.

1371 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

1372 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:
1373 SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard 200602, February 2006.

1374 <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

1375 **[RTTM]**

1376 V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, May
1377 1992.

1378 <http://www.rfc-editor.org/rfc/rfc1323.txt>

1379 **[SecurityPolicy]**

1380 G. Della-Libra, et. al. "Web Services Security Policy Language (WS-SecurityPolicy)", July 2005

1381 <http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf>

1382 **[SecureConversation]**

1383 S. Anderson, et al, "Web Services Secure Conversation Language (WS-SecureConversation)," February
1384 2005.

1385 <http://schemas.xmlsoap.org/ws/2004/04/sc/>

1386 **[Trust]**

1387 S. Anderson, et al, "Web Services Trust Language (WS-Trust)," February 2005.

1388 <http://schemas.xmlsoap.org/ws/2005/02/trust>

1389 Appendix A. Schema

1390 The normative schema that is defined for WS-ReliableMessaging using [XML-Schema Part1] and [XML-
1391 Schema Part2] is located at:

1392 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-200608.xsd>

1393 The following copy is provided for reference.

```
1394 <?xml version="1.0" encoding="UTF-8"?>
1395 <!--
1396 OASIS takes no position regarding the validity or scope of any intellectual
1397 property or other rights that might be claimed to pertain to the
1398 implementation or use of the technology described in this document or the
1399 extent to which any license under such rights might or might not be available;
1400 neither does it represent that it has made any effort to identify any such
1401 rights. Information on OASIS's procedures with respect to rights in OASIS
1402 specifications can be found at the OASIS website. Copies of claims of rights
1403 made available for publication and any assurances of licenses to be made
1404 available, or the result of an attempt made to obtain a general license or
1405 permission for the use of such proprietary rights by implementors or users of
1406 this specification, can be obtained from the OASIS Executive Director.
1407 OASIS invites any interested party to bring to its attention any copyrights,
1408 patents or patent applications, or other proprietary rights which may cover
1409 technology that may be required to implement this specification. Please
1410 address the information to the OASIS Executive Director.
1411 Copyright © OASIS Open 2002-2006. All Rights Reserved.
1412 This document and translations of it may be copied and furnished to others,
1413 and derivative works that comment on or otherwise explain it or assist in its
1414 implementation may be prepared, copied, published and distributed, in whole or
1415 in part, without restriction of any kind, provided that the above copyright
1416 notice and this paragraph are included on all such copies and derivative
1417 works. However, this document itself does not be modified in any way, such as
1418 by removing the copyright notice or references to OASIS, except as needed for
1419 the purpose of developing OASIS specifications, in which case the procedures
1420 for copyrights defined in the OASIS Intellectual Property Rights document must
1421 be followed, or as required to translate it into languages other than English.
1422 The limited permissions granted above are perpetual and will not be revoked by
1423 OASIS or its successors or assigns.
1424 This document and the information contained herein is provided on an "AS IS"
1425 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1426 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1427 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1428 FOR A PARTICULAR PURPOSE.
1429 -->
1430 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
1431 xmlns:wsa="http://www.w3.org/2005/08/addressing"
1432 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1433 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1434 elementFormDefault="qualified" attributeFormDefault="unqualified">
1435   <xs:import namespace="http://www.w3.org/2005/08/addressing"
1436   schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd"/>
1437   <!-- Protocol Elements -->
1438   <xs:complexType name="SequenceType">
1439     <xs:sequence>
1440       <xs:element ref="wsrm:Identifier"/>
1441       <xs:element name="MessageNumber" type="wsrm:MessageNumberType"/>
1442       <xs:any namespace="##other" processContents="lax" minOccurs="0"
1443   maxOccurs="unbounded"/>
1444     </xs:sequence>
```

```

1445     <xs:anyAttribute namespace="##other" processContents="lax"/>
1446 </xs:complexType>
1447 <xs:element name="Sequence" type="wsrm:SequenceType"/>
1448 <xs:element name="SequenceAcknowledgement">
1449   <xs:complexType>
1450     <xs:sequence>
1451       <xs:element ref="wsrm:Identifier"/>
1452       <xs:choice>
1453         <xs:sequence>
1454           <xs:choice>
1455             <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
1456               <xs:complexType>
1457                 <xs:sequence/>
1458                 <xs:attribute name="Upper" type="xs:unsignedLong"
1459 use="required"/>
1460                 <xs:attribute name="Lower" type="xs:unsignedLong"
1461 use="required"/>
1462               <xs:anyAttribute namespace="##other" processContents="lax"/>
1463             </xs:complexType>
1464           </xs:element>
1465           <xs:element name="None">
1466             <xs:complexType>
1467               <xs:sequence/>
1468             </xs:complexType>
1469           </xs:element>
1470         </xs:choice>
1471       <xs:element name="Final" minOccurs="0">
1472         <xs:complexType>
1473           <xs:sequence/>
1474         </xs:complexType>
1475       </xs:element>
1476     </xs:sequence>
1477     <xs:element name="Nack" type="xs:unsignedLong"
1478 maxOccurs="unbounded"/>
1479   </xs:choice>
1480   <xs:any namespace="##other" processContents="lax" minOccurs="0"
1481 maxOccurs="unbounded"/>
1482 </xs:sequence>
1483 <xs:anyAttribute namespace="##other" processContents="lax"/>
1484 </xs:complexType>
1485 </xs:element>
1486 <xs:complexType name="AckRequestedType">
1487   <xs:sequence>
1488     <xs:element ref="wsrm:Identifier"/>
1489     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1490 maxOccurs="unbounded"/>
1491   </xs:sequence>
1492   <xs:anyAttribute namespace="##other" processContents="lax"/>
1493 </xs:complexType>
1494 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1495 <xs:element name="Identifier">
1496   <xs:complexType>
1497     <xs:annotation>
1498       <xs:documentation>
1499         This type is for elements whose [children] is an anyURI and can have
1500 arbitrary attributes.
1501       </xs:documentation>
1502     </xs:annotation>
1503     <xs:simpleContent>
1504       <xs:extension base="xs:anyURI">
1505         <xs:anyAttribute namespace="##other" processContents="lax"/>
1506       </xs:extension>
1507     </xs:simpleContent>

```

```

1508     </xs:complexType>
1509 </xs:element>
1510 <xs:element name="Address">
1511   <xs:complexType>
1512     <xs:simpleContent>
1513       <xs:extension base="xs:anyURI">
1514         <xs:anyAttribute namespace="##other" processContents="lax"/>
1515       </xs:extension>
1516     </xs:simpleContent>
1517   </xs:complexType>
1518 </xs:element>
1519 <xs:simpleType name="MessageNumberType">
1520   <xs:restriction base="xs:unsignedLong">
1521     <xs:minInclusive value="1"/>
1522     <xs:maxInclusive value="9223372036854775807"/>
1523   </xs:restriction>
1524 </xs:simpleType>
1525 <!-- Fault Container and Codes -->
1526 <xs:simpleType name="FaultCodes">
1527   <xs:restriction base="xs:QName">
1528     <xs:enumeration value="wsrm:SequenceTerminated"/>
1529     <xs:enumeration value="wsrm:UnknownSequence"/>
1530     <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1531     <xs:enumeration value="wsrm:MessageNumberRollover"/>
1532     <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1533     <xs:enumeration value="wsrm:SequenceClosed"/>
1534     <xs:enumeration value="wsrm:WSRMRequired"/>
1535     <xs:enumeration value="wsrm:UnsupportedSelection"/>
1536   </xs:restriction>
1537 </xs:simpleType>
1538 <xs:complexType name="SequenceFaultType">
1539   <xs:sequence>
1540     <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1541     <xs:element name="Detail" type="wsrm:DetailType" minOccurs="0"/>
1542     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1543 maxOccurs="unbounded"/>
1544   </xs:sequence>
1545   <xs:anyAttribute namespace="##other" processContents="lax"/>
1546 </xs:complexType>
1547 <xs:complexType name="DetailType">
1548   <xs:sequence>
1549     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1550 maxOccurs="unbounded"/>
1551   </xs:sequence>
1552   <xs:anyAttribute namespace="##other" processContents="lax"/>
1553 </xs:complexType>
1554 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1555 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1556 <xs:element name="CreateSequenceResponse"
1557 type="wsrm:CreateSequenceResponseType"/>
1558 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1559 <xs:element name="CloseSequenceResponse"
1560 type="wsrm:CloseSequenceResponseType"/>
1561 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1562 <xs:element name="TerminateSequenceResponse"
1563 type="wsrm:TerminateSequenceResponseType"/>
1564 <xs:complexType name="CreateSequenceType">
1565   <xs:sequence>
1566     <xs:element ref="wsrm:AcksTo"/>
1567     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1568     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1569     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1570 maxOccurs="unbounded"/>

```

```

1571     <xs:annotation>
1572         <xs:documentation>
1573             It is the authors intent that this extensibility be used to
1574 transfer a Security Token Reference as defined in WS-Security.
1575         </xs:documentation>
1576     </xs:annotation>
1577 </xs:any>
1578 </xs:sequence>
1579 <xs:anyAttribute namespace="##other" processContents="lax"/>
1580 </xs:complexType>
1581 <xs:complexType name="CreateSequenceResponseType">
1582     <xs:sequence>
1583         <xs:element ref="wsrm:Identifier"/>
1584         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1585         <xs:element name="IncompleteSequenceBehavior"
1586 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1587         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1588         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1589 maxOccurs="unbounded"/>
1590     </xs:sequence>
1591     <xs:anyAttribute namespace="##other" processContents="lax"/>
1592 </xs:complexType>
1593 <xs:complexType name="CloseSequenceType">
1594     <xs:sequence>
1595         <xs:element ref="wsrm:Identifier"/>
1596         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1597 maxOccurs="unbounded"/>
1598     </xs:sequence>
1599     <xs:anyAttribute namespace="##other" processContents="lax"/>
1600 </xs:complexType>
1601 <xs:complexType name="CloseSequenceResponseType">
1602     <xs:sequence>
1603         <xs:element ref="wsrm:Identifier"/>
1604         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1605 maxOccurs="unbounded"/>
1606     </xs:sequence>
1607     <xs:anyAttribute namespace="##other" processContents="lax"/>
1608 </xs:complexType>
1609 <xs:complexType name="TerminateSequenceType">
1610     <xs:sequence>
1611         <xs:element ref="wsrm:Identifier"/>
1612         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1613 maxOccurs="unbounded"/>
1614     </xs:sequence>
1615     <xs:anyAttribute namespace="##other" processContents="lax"/>
1616 </xs:complexType>
1617 <xs:complexType name="TerminateSequenceResponseType">
1618     <xs:sequence>
1619         <xs:element ref="wsrm:Identifier"/>
1620         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1621 maxOccurs="unbounded"/>
1622     </xs:sequence>
1623     <xs:anyAttribute namespace="##other" processContents="lax"/>
1624 </xs:complexType>
1625 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1626 <xs:complexType name="OfferType">
1627     <xs:sequence>
1628         <xs:element ref="wsrm:Identifier"/>
1629         <xs:element name="Endpoint" type="wsa:EndpointReferenceType"/>
1630         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1631         <xs:element name="IncompleteSequenceBehavior"
1632 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1633         <xs:any namespace="##other" processContents="lax" minOccurs="0"

```

```

1634 maxOccurs="unbounded"/>
1635     </xs:sequence>
1636     <xs:anyAttribute namespace="##other" processContents="lax"/>
1637 </xs:complexType>
1638 <xs:complexType name="AcceptType">
1639     <xs:sequence>
1640         <xs:element ref="wsrm:AcksTo"/>
1641         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1642 maxOccurs="unbounded"/>
1643     </xs:sequence>
1644     <xs:anyAttribute namespace="##other" processContents="lax"/>
1645 </xs:complexType>
1646 <xs:element name="Expires">
1647     <xs:complexType>
1648         <xs:simpleContent>
1649             <xs:extension base="xs:duration">
1650                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1651             </xs:extension>
1652         </xs:simpleContent>
1653     </xs:complexType>
1654 </xs:element>
1655 <xs:simpleType name="IncompleteSequenceBehaviorType">
1656     <xs:restriction base="xs:string">
1657         <xs:enumeration value="DiscardEntireSequence"/>
1658         <xs:enumeration value="DiscardFollowingFirstGap"/>
1659         <xs:enumeration value="NoDiscard"/>
1660     </xs:restriction>
1661 </xs:simpleType>
1662 <xs:element name="UsesSequenceSTR">
1663     <xs:complexType>
1664         <xs:sequence/>
1665         <xs:anyAttribute namespace="##other" processContents="lax"/>
1666     </xs:complexType>
1667 </xs:element>
1668 <xs:element name="UsesSequenceSSL">
1669     <xs:complexType>
1670         <xs:sequence/>
1671         <xs:anyAttribute namespace="##other" processContents="lax"/>
1672     </xs:complexType>
1673 </xs:element>
1674 <xs:element name="UnsupportedElement">
1675     <xs:simpleType>
1676         <xs:restriction base="xs:QName"/>
1677     </xs:simpleType>
1678 </xs:element>
1679 </xs:schema>

```

1680 Appendix B. WSDL

1681 This WSDL describes the WS-RM protocol from the point of view of an RM Destination. In the case where
1682 an endpoint acts both as an RM Destination and an RM Source, note that additional messages may be
1683 present in exchanges with that endpoint.

1684 Also note that this WSDL is intended to describe the internal structure of the WS-RM protocol, and will not
1685 generally appear in a description of a WS-RM-capable Web service. See WS-RM Policy [WS-RM Policy]
1686 for a higher-level mechanism to indicate that WS-RM is engaged.

1687 The normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1688 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsd/wsrn-1.1-wsd-200608.wsd>

1689 The following non-normative copy is provided for reference.

```
1690 <?xml version="1.0" encoding="utf-8"?>
1691 <!--
1692 OASIS takes no position regarding the validity or scope of any intellectual
1693 property or other rights that might be claimed to pertain to the
1694 implementation or use of the technology described in this document or the
1695 extent to which any license under such rights might or might not be available;
1696 neither does it represent that it has made any effort to identify any such
1697 rights. Information on OASIS's procedures with respect to rights in OASIS
1698 specifications can be found at the OASIS website. Copies of claims of rights
1699 made available for publication and any assurances of licenses to be made
1700 available, or the result of an attempt made to obtain a general license or
1701 permission for the use of such proprietary rights by implementors or users of
1702 this specification, can be obtained from the OASIS Executive Director.
1703 OASIS invites any interested party to bring to its attention any copyrights,
1704 patents or patent applications, or other proprietary rights which may cover
1705 technology that may be required to implement this specification. Please
1706 address the information to the OASIS Executive Director.
1707 Copyright (c) OASIS Open 2002-2006. All Rights Reserved.
1708 This document and translations of it may be copied and furnished to others,
1709 and derivative works that comment on or otherwise explain it or assist in its
1710 implementation may be prepared, copied, published and distributed, in whole or
1711 in part, without restriction of any kind, provided that the above copyright
1712 notice and this paragraph are included on all such copies and derivative
1713 works. However, this document itself does not be modified in any way, such as
1714 by removing the copyright notice or references to OASIS, except as needed for
1715 the purpose of developing OASIS specifications, in which case the procedures
1716 for copyrights defined in the OASIS Intellectual Property Rights document must
1717 be followed, or as required to translate it into languages other than English.
1718 The limited permissions granted above are perpetual and will not be revoked by
1719 OASIS or its successors or assigns.
1720 This document and the information contained herein is provided on an "AS IS"
1721 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1722 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1723 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1724 FOR A PARTICULAR PURPOSE.
1725 -->
1726 <wsc:definitions xmlns:wsc="http://schemas.xmlsoap.org/wsc/"
1727 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1728 xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:rm="http://docs.oasis-
1729 open.org/ws-rx/wsrn/200608" xmlns:tns="http://docs.oasis-open.org/ws-
1730 rx/wsrn/200608/wsd" targetNamespace="http://docs.oasis-open.org/ws-
1731 rx/wsrn/200608/wsd">
1732 <wsc:types>
```

```

1733     <xs:schema>
1734         <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsr/200608"
1735 schemaLocation="http://docs.oasis-open.org/ws-rx/wsr/200608/wsr-1.1-schema-
1736 200608.xsd"/>
1737     </xs:schema>
1738 </wsdl:types>

1739     <wsdl:message name="CreateSequence">
1740         <wsdl:part name="create" element="rm:CreateSequence"/>
1741     </wsdl:message>
1742     <wsdl:message name="CreateSequenceResponse">
1743         <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1744     </wsdl:message>
1745     <wsdl:message name="CloseSequence">
1746         <wsdl:part name="close" element="rm:CloseSequence"/>
1747     </wsdl:message>
1748     <wsdl:message name="CloseSequenceResponse">
1749         <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1750     </wsdl:message>
1751     <wsdl:message name="TerminateSequence">
1752         <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1753     </wsdl:message>
1754     <wsdl:message name="TerminateSequenceResponse">
1755         <wsdl:part name="terminateResponse"
1756 element="rm:TerminateSequenceResponse"/>
1757     </wsdl:message>

1758     <wsdl:portType name="SequenceAbstractPortType">
1759         <wsdl:operation name="CreateSequence">
1760             <wsdl:input message="tns:CreateSequence" wsaw:Action="http://docs.oasis-
1761 open.org/ws-rx/wsr/200608/CreateSequence"/>
1762             <wsdl:output message="tns:CreateSequenceResponse"
1763 wsaw:Action="http://docs.oasis-open.org/ws-
1764 rx/wsr/200608/CreateSequenceResponse"/>
1765         </wsdl:operation>
1766         <wsdl:operation name="CloseSequence">
1767             <wsdl:input message="tns:CloseSequence" wsaw:Action="http://docs.oasis-
1768 open.org/ws-rx/wsr/200608/CloseSequence"/>
1769             <wsdl:output message="tns:CloseSequenceResponse"
1770 wsaw:Action="http://docs.oasis-open.org/ws-
1771 rx/wsr/200608/CloseSequenceResponse"/>
1772         </wsdl:operation>
1773         <wsdl:operation name="TerminateSequence">
1774             <wsdl:input message="tns:TerminateSequence"
1775 wsaw:Action="http://docs.oasis-open.org/ws-rx/wsr/200608/TerminateSequence"/>
1776             <wsdl:output message="tns:TerminateSequenceResponse"
1777 wsaw:Action="http://docs.oasis-open.org/ws-
1778 rx/wsr/200608/TerminateSequenceResponse"/>
1779         </wsdl:operation>
1780     </wsdl:portType>

1781 </wsdl:definitions>

```


1782 Appendix C. Message Examples

1783 Appendix C.1 Create Sequence

1784 Create Sequence

```
1785 <?xml version="1.0" encoding="UTF-8"?>
1786 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1787 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1788 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1789   <S:Header>
1790     <wsa:MessageID>
1791       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1792     </wsa:MessageID>
1793     <wsa:To>http://example.com/serviceB/123</wsa:To>
1794     <wsa:Action>http://docs.oasis-open.org/ws-
1795 rx/wsmr/200608/CreateSequence</wsa:Action>
1796     <wsa:ReplyTo>
1797       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1798     </wsa:ReplyTo>
1799   </S:Header>
1800   <S:Body>
1801     <wsmr:CreateSequence>
1802       <wsmr:AcksTo>
1803         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1804       </wsmr:AcksTo>
1805     </wsmr:CreateSequence>
1806   </S:Body>
1807 </S:Envelope>
```

1808 Create Sequence Response

```
1809 <?xml version="1.0" encoding="UTF-8"?>
1810 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1811 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1812 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1813   <S:Header>
1814     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1815     <wsa:RelatesTo>
1816       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1817     </wsa:RelatesTo>
1818     <wsa:Action>
1819       http://docs.oasis-open.org/ws-rx/wsmr/200608/CreateSequenceResponse
1820     </wsa:Action>
1821   </S:Header>
1822   <S:Body>
1823     <wsmr:CreateSequenceResponse>
1824       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1825     </wsmr:CreateSequenceResponse>
1826   </S:Body>
1827 </S:Envelope>
```

1828 Appendix C.2 Initial Transmission

1829 The following example WS-ReliableMessaging headers illustrate the message exchange in the above
1830 figure. The three messages have the following headers; the third message is identified as the last
1831 message in the Sequence:

1832 **Message 1**

```
1833 <?xml version="1.0" encoding="UTF-8"?>
1834 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1835 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1836 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1837   <S:Header>
1838     <wsa:MessageID>
1839       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1840     </wsa:MessageID>
1841     <wsa:To>http://example.com/serviceB/123</wsa:To>
1842     <wsa:From>
1843       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1844     </wsa:From>
1845     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1846     <wsmr:Sequence>
1847       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1848       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1849     </wsmr:Sequence>
1850   </S:Header>
1851   <S:Body>
1852     <!-- Some Application Data -->
1853   </S:Body>
1854 </S:Envelope>
```

1855 **Message 2**

```
1856 <?xml version="1.0" encoding="UTF-8"?>
1857 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1858 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1859 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1860   <S:Header>
1861     <wsa:MessageID>
1862       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1863     </wsa:MessageID>
1864     <wsa:To>http://example.com/serviceB/123</wsa:To>
1865     <wsa:From>
1866       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1867     </wsa:From>
1868     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1869     <wsmr:Sequence>
1870       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1871       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1872     </wsmr:Sequence>
1873   </S:Header>
1874   <S:Body>
1875     <!-- Some Application Data -->
1876   </S:Body>
1877 </S:Envelope>
```

1878 **Message 3**

```
1879 <?xml version="1.0" encoding="UTF-8"?>
1880 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1881 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1882 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1883   <S:Header>
1884     <wsa:MessageID>
1885       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1886     </wsa:MessageID>
1887     <wsa:To>http://example.com/serviceB/123</wsa:To>
1888     <wsa:From>
1889       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1890 </wsa:From>
1891 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1892 <wsrm:Sequence>
1893 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1894 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1895 </wsrm:Sequence>
1896 <wsrm:AckRequested>
1897 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1898 </wsrm:AckRequested>
1899 </S:Header>
1900 <S:Body>
1901 <!-- Some Application Data -->
1902 </S:Body>
1903 </S:Envelope>

```

1904 **Appendix C.3 First Acknowledgement**

1905 Message number 2 has not been accepted by the RM Destination due to some transmission error so it
1906 responds with an Acknowledgement for messages 1 and 3:

```

1907 <?xml version="1.0" encoding="UTF-8"?>
1908 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1909 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1910 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1911 <S:Header>
1912 <wsa:MessageID>
1913 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1914 </wsa:MessageID>
1915 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1916 <wsa:From>
1917 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1918 </wsa:From>
1919 <wsa:Action>
1920 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1921 </wsa:Action>
1922 <wsrm:SequenceAcknowledgement>
1923 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1924 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1925 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1926 </wsrm:SequenceAcknowledgement>
1927 </S:Header>
1928 <S:Body/>
1929 </S:Envelope>

```

1930 **Appendix C.4 Retransmission**

1931 The RM Sourcediscovers that message number 2 was not accepted so it resends the message and
1932 requests an Acknowledgement:

```

1933 <?xml version="1.0" encoding="UTF-8"?>
1934 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1935 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1936 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1937 <S:Header>
1938 <wsa:MessageID>
1939 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1940 </wsa:MessageID>
1941 <wsa:To>http://example.com/serviceB/123</wsa:To>
1942 <wsa:From>
1943 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1944 </wsa:From>

```

```

1945 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1946 <wsrm:Sequence>
1947 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1948 <wsrm:MessageNumber>2</wsrm:MessageNumber>
1949 </wsrm:Sequence>
1950 <wsrm:AckRequested>
1951 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1952 </wsrm:AckRequested>
1953 </S:Header>
1954 <S:Body>
1955 <!-- Some Application Data -->
1956 </S:Body>
1957 </S:Envelope>

```

1958 **Appendix C.5 Termination**

1959 The RM Destination now responds with an Acknowledgement for the complete Sequence which can then
1960 be terminated:

```

1961 <?xml version="1.0" encoding="UTF-8"?>
1962 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1963 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1964 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1965 <S:Header>
1966 <wsa:MessageID>
1967 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1968 </wsa:MessageID>
1969 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1970 <wsa:From>
1971 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1972 </wsa:From>
1973 <wsa:Action>
1974 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1975 </wsa:Action>
1976 <wsrm:SequenceAcknowledgement>
1977 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1978 <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1979 </wsrm:SequenceAcknowledgement>
1980 </S:Header>
1981 <S:Body/>
1982 </S:Envelope>

```

1983 **Terminate Sequence**

```

1984 <?xml version="1.0" encoding="UTF-8"?>
1985 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1986 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1987 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1988 <S:Header>
1989 <wsa:MessageID>
1990 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1991 </wsa:MessageID>
1992 <wsa:To>http://example.com/serviceB/123</wsa:To>
1993 <wsa:Action>
1994 http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence
1995 </wsa:Action>
1996 <wsa:From>
1997 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1998 </wsa:From>
1999 </S:Header>
2000 <S:Body>
2001 <wsrm:TerminateSequence>

```

```
2002     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2003     </wsrm:TerminateSequence>
2004     </S:Body>
2005     </S:Envelope>
```

2006 Terminate Sequence Response

```
2007     <?xml version="1.0" encoding="UTF-8"?>
2008     <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2009     xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsr/200608"
2010     xmlns:wsa="http://www.w3.org/2005/08/addressing">
2011     <S:Header>
2012     <wsa:MessageID>
2013     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
2014     </wsa:MessageID>
2015     <wsa:To>http://example.com/serviceA/789</wsa:To>
2016     <wsa:Action>
2017     http://docs.oasis-open.org/ws-rx/wsr/200608/TerminateSequenceResponse
2018     </wsa:Action>
2019     <wsa:RelatesTo>
2020     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
2021     </wsa:RelatesTo>
2022     <wsa:From>
2023     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2024     </wsa:From>
2025     </S:Header>
2026     <S:Body>
2027     <wsrm:TerminateSequenceResponse>
2028     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2029     </wsrm:TerminateSequenceResponse>
2030     </S:Body>
2031     </S:Envelope>
```

2032 Appendix D. State Tables

2033 This appendix specifies the non-normative state transition tables for RM Source and RM Destination.

2034 The state tables describe the lifetime of a sequence in both the RM Source and the RM Destination

2035 Legend:

2036 The first column of these tables contains the motivating event and has the following format:

Event
<i>Event name</i> [source] {ref}

2037 Where:

- 2038 ● Event Name: indicates the name of the event. Event Names surrounded by "<>" are optional as
2039 described by the specification.
- 2040 ● [source]: indicates the source of the event; one of:
 - 2041 ● [msg] a Received message
 - 2042 ● [int]: an internal event such as the firing of a timer
 - 2043 ● [app]: the application
 - 2044 ● [unspec]: the source is unspecified

2045 Each event / state combination cell in the tables in this appendix has the following format:

State Name
<i>Action to take</i> [next state] {ref}

2046 Where:

- 2047 ● action to take: indicates that the state machine performs the following action. Actions surrounded
2048 by "<>" are optional as described by the specification. "Xmit" is used as a short form for the word
2049 "Transmit"
- 2050 ● [next state]: indicates the state to which the state machine will advance upon the performance of
2051 the action. For ease of reading the next state "same" indicates that the state does not change.
- 2052 ● {ref} is a reference to the document section describing the behavior in this cell

2053 "N/A" in a cell indicates a state / event combination self-inconsistent with the state machine; should these
2054 conditions occur, it would indicate an implementation error. A blank cell indicates that the behavior is not
2055 described in this specification and does not indicate normal protocol operation. Implementations MAY
2056 generate a Sequence Terminated fault (see section 4.2) in these circumstances. Robust implementations
2057 MUST be able to operate in a stable manner despite the occurrence of unspecified event / state
2058 combinations.

2059 Table 1 RM Source Sequence State Transition Table

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
Create Sequence [unspec] {3.4}	Xmit Create Sequence [Creating] {3.4}	N/A	N/A	N/A	N/A	N/A
Create Sequence Response [msg] {3.4}		Process Create Sequence Response [Created] {3.4}				
Create Sequence Refused Fault [msg] {3.4}		No action [None] {4.6}				
Send message [app] {2.1}	N/A	N/A	Xmit message [Same] {2}	No action [Same] {2}	N/A	N/A
Retransmit of un-ack'd message [int]	N/A	N/A	Xmit message [Same] {2.4}	Xmit message [Same] {2.4}	N/A	N/A
SeqAck (non-final) [msg] {3.9}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Process Ack ranges [Same] {3.9}	Process Ack ranges [Same] {3.9}	Process Ack ranges [Same] {3.9}	Process Ack ranges [Same] {3.9}
Nack [msg] {3.9}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	<Xmit message(s)> [Same] {3.9}	<Xmit message(s)> [Same] {3.9}	No action [Same]	No action [Same]
Message Number Rollover Fault [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	No action [Rollover]	No action [Same]	No action [Same]	No action [Same]
CloseSequence [msg] {3.5}	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Xmit CloseSequence Response [Closed] {3.5}</u>	<u>Xmit CloseSequence Response [Closed] {3.5}</u>	<u>Xmit CloseSequence Response [Closed] {3.5}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>
<Close Sequence> [int] {3.5}	N/A		Xmit Close Sequence [Closing] {3.5}	N/A	N/A	N/A
Close Sequence Response [msg] {3.5}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}		No action [Closed] {3.5}	No action [Same] {3.5}	No action [Same] {3.5}

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
SeqAck (final) [msg] {3.9}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Process Ack ranges [Closed] {3.9}	Process Ack ranges [Closed] {3.9}	Process Ack ranges [Same]	Process Ack ranges [Same]
Sequence Closed Fault [msg] {4.7}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	No action [Closed] {4.7}	No action [Closed] {4.7}	No action [Same]	No action [Same]
Unknown Sequence Fault [msg] {4.3}			Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}
Sequence Terminated Fault [msg] {4.2}	N/A		Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}
TerminateSequence [msg] {3.6}	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Xmit Terminate Sequence Response [None] {3.6}</u>	<u>Xmit Terminate Sequence Response [None] {3.6}</u>	<u>Xmit Terminate Sequence Response [None] {3.6}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>
Terminate Sequence [int]	N/A	No action [None] {unspec}	Xmit Terminate Sequence [Terminating]	Xmit Terminate Sequence [Terminating]	Xmit Terminate Sequence [Terminating]	N/A
Terminate Sequence Response [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}				Terminate Sequence [None] {3.6}
Expires exceeded [int]	N/A	Terminate Sequence [None] {3.7}	Terminate Sequence [None] {3.7}	Terminate Sequence [None] {3.7}	Terminate Sequence [None] {3.7}	Terminate Sequence [None] {3.7}
Invalid Acknowledgement [msg] {4.4}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Invalid Acknowledgement Fault [Same] {4.4}	Generate Invalid Acknowledgement Fault [Same] {4.4}	Generate Invalid Acknowledgement Fault [Same] {4.4}	Generate Invalid Acknowledgement Fault [Same] {4.4}

2059 Table 2 RM Destination Sequence State Transition Table

Events	Sequence States			
	None	Created	Closed	Terminating
CreateSequence (successful) [msg/int] {3.4}	Xmit Create Sequence Response [Created] {3.4}	N/A	N/A	

Events	Sequence States			
	None	Created	Closed	Terminating
CreateSequence (unsuccessful) [msg/int] {3.4}	Generate Create Sequence Refused Fault [None] {3.4}	/AN		N/A
Message (with message number within range) [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Accept Message; <Xmit SeqAck> [Same]	<u>Generate Sequence Closed Fault (with SeqAck+Final)</u> [Same] {3.5}	<u>Generate Sequence-Closed-Fault (with SeqAck+Final)</u> [Same] {3.5} <u>Generate Unknown Sequence Fault</u> [Same] {4.3}
Message (with message number outside of range) [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Message Number Rollover Fault [Same] {3.7}{4.5}	<u>Generate Sequence Closed Fault (with SeqAck+Final)</u> [Same] {3.5}	<u>Generate Sequence-Closed-Fault (with SeqAck+Final)</u> [Same] {3.5} <u>Generate Unknown Sequence Fault</u> [Same] {4.3}
<AckRequested> [msg] {3.8}	Generate Unknown Seq Fault [Same] {4.3}	Xmit SeqAck [Same] {3.8}	<u>Xmit SeqAck+Final</u> [Same] {3.9}	<u>Xmit SeqAck+Final</u> [Same] {3.9} <u>Generate Unknown Sequence Fault</u> [Same] {4.3}
CloseSequence [msg] {3.5}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit CloseSequence Response with SeqAck+Final [Closed] {3.5}	<u>Xmit CloseSequenceResponse with SeqAck+Final</u> [Same] {3.5}	<u>Response with SeqAck+Final</u> [Closed] {3.5} <u>Xmit CloseSequenceGenerate Unknown Sequence Fault</u> [Same] {4.3}
<CloseSequence autonomously> [int]	N/A	<u>Xmit CloseSequence with SeqAck+Final</u> [Closed] {3.5} <u>No Action</u> [Closed]	<u>Xmit CloseSequence with SeqAck+Final</u> [Same] {3.5}	N/A
CloseSequenceResponse [msg] {3.5}	<u>Generate Unknown Sequence Fault</u> [Same] {4.3}		<u>No Action</u> [Closed] {3.5}	<u>Generate Unknown Sequence Fault</u> [Same] {4.3}
TerminateSequence [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Terminate Sequence-Response [None] {3.6}	<u>Xmit TerminateSequenceResponse</u> [None] {3.6}	Xmit Terminate Sequence-Response [None] {3.6}
<TerminateSequence autonomously> [int]		<u>Xmit TerminateSequence with SeqAck+Final</u> [Terminating] {3.6}	<u>Xmit TerminateSequence with SeqAck+Final</u> [Terminating] {3.6}	<u>Xmit TerminateSequence with SeqAck+Final</u> [None] {3.6}
TerminateSequenceResponse [msg]	<u>Generate Unknown Sequence Fault</u> [Same] {4.3}			<u>Terminate Sequence</u> [None]

Events	Sequence States			
	None	Created	Closed	Terminating
UnknownSequence Fault [msg] {4.3}	-	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}Terminate Sequence [None] {4.3}
SequenceTerminated Fault [msg] {4.2}	-	Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}Terminate Sequence [None] {4.2}
Invalid Acknowledgement Fault [msg] {4.4}	N/A	-		-
Expires exceeded [int]	N/A	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}
<Seq Acknowledgement autonomously> [int] {3.9}	N/A	Xmit SeqAck [Same] {3.9}	Xmit SeqAck+Final [Same] {3.9}	Xmit SeqAck+Final [Same] {3.9}
Non WSRM message when WSRM required [msg] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}Generate WSRMRequired Fault [Same] {4.8}

2060 **Appendix E. Acknowledgments**

2061 This document is based on initial contribution to OASIS WS-RX Technical Committee by the following
2062 authors:

2063 Ruslan Bilorusets(BEA), Don Box(Microsoft), Luis Felipe Cabrera(Microsoft), Doug Davis(IBM),
2064 Donald Ferguson(IBM), Christopher Ferris-Editor(BM), Tom Freund(IBM), Mary Ann Hondo(IBM),
2065 John Ibbotson(IBM), Lei Jin(BEA), Chris Kaler(Microsoft), David Langworthy-Editor(Microsoft),
2066 Amelia Lewis(TIBCO Software), Rodney Limprecht(Microsoft), Steve Lucco(Microsoft), Don
2067 Mullen(TIBCO Software), Anthony Nadalin(IBM), Mark Nottingham(BEA), David Orchard(BEA),
2068 Jamie Roots(IBM), Shivajee Samdarshi(TIBCO Software), John Shewchuk(Microsoft), Tony
2069 Storey(IBM).

2070 The following individuals have provided invaluable input into the initial contribution:

2071 Keith Ballinger(Microsoft), Stefan Batres(Microsoft), Rebecca Bergersen(Iona), Allen
2072 Brown(Microsoft), Michael Conner(IBM), George Copeland(Microsoft), Francisco Curbera(IBM),
2073 Paul Fremantle(IBM), Steve Graham(IBM), Pat Helland(Microsoft), Rick Hill(Microsoft), Scott
2074 Hinkelman(IBM), Tim Holloway(IBM), Efim Hudis(Microsoft), David Ingham(Microsoft), Gopal
2075 Kakivaya(Microsoft), Johannes Klein(Microsoft), Frank Leymann(IBM), Martin Nally(IBM), Peter
2076 Niblett(IBM), Jeffrey Schlimmer(Microsoft), James Snell(IBM), Keith Stobie(Microsoft), Satish
2077 Thatte(Microsoft), Stephen Todd(IBM), Sanjiva Weerawarana(IBM), Roger Wolter(Microsoft).

2078 The following individuals were members of the committee during the development of this specification:

2079 Abbie Barbir(Nortel), Charlton Barreto(Adobe), Stefan Batres(Microsoft), Hamid Ben
2080 Malek(Fujitsu), Andreas Bjarlestam(Ericsson), Toufic Boubez(Layer 7), Doug Bunting(Sun), Lloyd
2081 Burch(Novell), Steve Carter(Novell), Martin Chapman(Oracle), Dave Chappell(Sonic), Paul
2082 Cotton(Microsoft), Glen Daniels(Sonic), Doug Davis(IBM), Blake Dournaee(Intel), Jacques
2083 Durand(Fujitsu), Colleen Evans(Microsoft), Christopher Ferris(IBM), Paul Fremantle(WSO2),
2084 Robert Freund(Hitachi), Peter Furniss(Erebor), Marc Goodner(Microsoft), Alastair
2085 Green(Choreology), Mike Grogan(Sun), Ondrej Hrebicek(Microsoft), Kazunori Iwasa(Fujitsu),
2086 Chamikara Jayalath(WSO2), Lei Jin(BEA), Ian Jones(BTplc), Anish Karmarkar(Oracle), Paul
2087 Knight(Nortel), Dan Leshchiner(Tibco), Mark Little(JBoss), Lily Liu(webMethods), Matt
2088 Lovett(IBM), Ashok Malhotra(Oracle), Jonathan Marsh(Microsoft), Daniel Millwood(IBM), Jeff
2089 Mischkinsky(Oracle), Nilo Mitra(Ericsson), Peter Niblett(IBM), Duane Nickull(Adobe), Eisaku
2090 Nishiyama(Hitachi), Dave Orchard(BEA), Chouthri Palanisamy(NEC), Sanjay Patil(SAP), Gilbert
2091 Pilz(BEA), Martin Raepfle(SAP), Eric Rajkovic(Oracle), Stefan Rossmannith(SAP), Tom
2092 Rutt(Fujitsu), Rich Salz(IBM), Shivajee Samdarshi(Tibco), Vladimir Videlov(SAP), Claus von
2093 Riegen(SAP), Pete Wenzel(Sun), Steve Winkler(SAP), Ümit Yalçinalp(SAP), Nobuyuki
2094 Yamamoto(Hitachi).

Appendix F. Revision History

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

Rev	Date	By Whom	What
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-08	2005-12-15	Doug Davis	Add back in RM Source to glossary
wd-08	2005-12-15	Steve Winkler	Doug added Steve's editorial nits
wd-08	2005-12-21	Doug Davis	i050
wd-08	2005-12-21	Doug Davis	i081
wd-08	2005-12-21	Doug Davis	i080 – but i050 negates the need for any changes
wd-08	2005-12-21	Doug Davis	i079
wd-08	2005-12-21	Doug Davis	i076 – didn't add text about "replies" since the RMD to RMS sequence could be used for any message not just replies
wd-08	2005-12-21	Umit Yalcinalp	Action Su03: removed wsse from Table 1
wd-08	2005-12-21	Umit Yalcinalp	i057 per Sunnyvale F2F 2005, Cleaned up some formatting errors in contributors
wd-08	2005-12-27	Doug Davis	i060
wd-08	2005-12-27	Gilbert Pilz	Moved schema and WSDL files to their own artifacts. Converted source document to

Rev	Date	By Whom	What
			OpenDocument Text format. Changed line numbers to be a single style.
wd-08	2005-12-28	Anish Karmarkar	Included a section link to c:\temp\wsrm-1.1-schema-200510.xsd and to c:\temp\wsrm-1.1-wsdl-200510.wsdl
wd-08	2006-01-04	Gilbert Pilz	Fixed formatting for included sections.
wd-08	2006-01-05	Gilbert Pilz	Created links for unused references. Fixed exemplars for CloseSequence and CloseSequenceResponse.
wd-09	2006-01-11	Doug Davis	Minor tweaks to text/typos.
wd-10	2006-01-23	Doug Davis	Accept all changes from wd-09 Make some minor editorial tweaks from Marc's comments.
wd-10	2006-02-14	Doug Davis	Issue 082 resolution
wd-10	2006-02-14	Doug Davis	Issue 083 resolution
wd-10	2006-02-14	Doug Davis	Issue 085 resolution
wd-10	2006-02-14	Doug Davis	Issues 086, 087 resolutions Defined MessageNumberType
wd-10	2006-02-15	Doug Davis	Issue 078 resolution
wd-10	2006-02-15	Doug Davis	Issue 094 resolution
wd-10	2006-02-15	Doug Davis	Issue 095 resolution
wd-10	2006-02-15	Gilbert Pilz	Issue 088 – added namespace URI link to namespace URI; added text explaining that this URI could be dereferenced to produce the RDDDL doc; added non-normative reference to RDDDL 2.0
wd-10	2006-02-17	Anish Karmarkar	Namespace changed to 200602 for both WSDL and XSD docs.
wd-10	2006-02-17	Anish Karmarkar	Issue i087 as it applies to WSRM spec.
wd-10	2006-02-17	Anish Karmarkar	Added titles and minor text for state table (issue i058).
wd-11	2006-02-22	Doug Davis	Accept all changes for new WD Minor typos fixed
wd-11	2006-02-23	Doug Davis	s'/close'/close/g – per Marc Goodner Added first ref to [URI] – per Marc G again
wd-11	2006-02-27	Doug Davis	Issue i061 applied
wd-11	2006-02-28	Doug Davis	Fixed typo around the use of "above" and "below"
wd-11	2006-03-01	Doug Davis	Minor typos found by Marc Goodner
wd-11	2006-03-02	Doug Davis	Minor typos found by Matt Lovett
wd-11	2006-03-08	Doug Davis	Issue 091 applied
wd-11	2006-03-08	Doug Davis	Issue 092 applied
wd-11	2006-03-08	Doug Davis	Issue 100 applied

Rev	Date	By Whom	What
wd-12	2006-03-20	Doug Davis	Added space in "SOAP1.x" – PaulCotton
wd-12	2006-04-11	Doug Davis	Issue 007 applied
wd-12	2006-04-11	Doug Davis	Issue 090 applied
wd-12	2006-04-11	Doug Davis	Issue 098 applied
wd-12	2006-04-11	Doug Davis	Issue 099 applied
wd-12	2006-04-11	Doug Davis	Issue 101 applied
wd-12	2006-04-11	Doug Davis	Issue 103 applied
wd-12	2006-04-11	Doug Davis	Issue 104 applied
wd-12	2006-04-11	Doug Davis	Issue 105 applied
wd-12	2006-04-11	Doug Davis	Issue 107 applied
wd-12	2006-04-11	Doug Davis	Issue 109 applied
wd-12	2006-04-11	Doug Davis	Issue 110 applied
wd-12	2006-04-12	Doug Davis	Used "generated" instead of "issue" or "send" when talking about faults.
wd-12	2006-04-24	Gilbert Pilz	Update references to WS-Addressing to the Proposed Recommendations; update WS-RM namespace to "200604".
wd-13	2006-05-08	Gilbert Pilz	i093 part 1; more work needed
wd-13	2006-05-10	Doug Davis	Issue 096 applied
wd-13	2006-05-26	Gilbert Pilz	i093 part 2; reflects decisions from 2006-05-25 meeting
wd-13	2006-05-28	Gilbert Pilz	Issue 106 applied
wd-13	2006-05-29	Gilbert Pilz	Issue 118 applied
wd-13	2006-05-29	Gilbert Pilz	Issue 120 applied
wd-13	2006-05-30	Gilbert Pilz	Issue 114 applied
wd-13	2006-05-30	Gilbert Pilz	Issue 116 applied
wd-14	2006-06-05	Gilbert Pilz	Accept all changes; bump WD number
wd-14	2006-06-07	Doug Davis	Applied lots of minor edits from Marc Goodner
wd-14	2006-06-07	Doug Davis	Change a couple of period/sp/sp to period/sp
wd-14	2006-06-07	Doug Davis	Added a space in "URI]of" – per Marc Goodner
wd-14	2006-06-07	Doug Davis	Issue 131 applied
wd-14	2006-06-07	Doug Davis	Issue 132 applied
wd-14	2006-06-07	Doug Davis	Issue 119 applied
wd-14	2006-06-07	Doug Davis	Applied lots of minor edits from Doug Davis
wd-14	2006-06-07	Doug Davis	s/"none"/"full-uri"/ - per Marc Goodner
wd-14	2006-06-12	Doug Davis	Complete i106
wd-14	2006-06-12	Doug Davis	Issues 089 applied
wd-14	2006-06-12	Doug Davis	Fix for several RFC2119 keywords – per Anish
wd-15	2006-06-12	Doug Davis	Accept all changed, dump WD number
wd-15	2006-06-12	Doug Davis	Move WSDL after Schema
wd-15	2006-06-12	Doug Davis	Nits – remove tabs, extra [yyy]'s ...
wd-15	2006-06-14	Doug Davis	Remove extra "OPTIONAL"s – Matt Lovett

Rev	Date	By Whom	What
wd-15	2006-06-14	Doug Davis	Remove blank rows/columns from state table. Fix italics in state table
wd-15	2006-06-15	Doug Davis	Typo – section D was empty
wd-15	2006-06-16	Doug Davis	Issue 125 applied
wd-15	2006-06-16	Doug Davis	Issue 126 applied
wd-15	2006-06-16	Doug Davis	Issue 127 applied
wd-15	2006-06-16	Doug Davis	Issue 133 applied
wd-15	2006-06-16	Doug Davis	Issue 136 applied
wd-15	2006-06-16	Doug Davis	Issue 138 applied
wd-15	2006-06-16	Doug Davis	Issue 135 applied
wd-15	2006-06-20	Doug Davis	Added all TC members to the ack list
wd-15	2006-06-22	Doug Davis	Issue 129 applied
wd-15	2006-06-22	Doug Davis	Issue 130 applied
wd-15	2006-06-22	Doug Davis	Issue 137 applied
wd-15	2006-06-26	Doug Davis	Issue 111 applied
wd-15	2006-06-26	Doug Davis	Missed a part of issue 129
wd-15	2006-06-30	Doug Davis	Fixed a typo in schema
wd-15	2006-06-30	Doug Davis	Issue 141 applied
wd-15	2006-06-30	Doug Davis	Issue 142 applied
wd-15	2006-06-30	Doug Davis	Issue 148 applied
wd-15	2006-06-30	Doug Davis	Issue 149 applied
wd-15	2006-06-30	Doug Davis	Issue 150 applied
wd-15	2006-07-06	Doug Davis	Issue 121 applied
wd-15	2006-07-21	Doug Davis	Issue 139 applied
wd-15	2006-07-21	Doug Davis	Issue 144 applied
wd-15	2006-07-21	Doug Davis	Issue 147 applied
wd-15	2006-07-21	Doug Davis	Issues 122-124 applied
wd-15	2006-07-27	Doug Davis	Updated list of oasis TC members (i134)
wd-15	2006-07-27	Doug Davis	Issue 140 applied
wd-15	2006-07-27	Doug Davis	Issue 145 applied
wd-15	2006-07-27	Doug Davis	Issue 143 applied
wd-15	2006-07-28	Doug Davis	Lots of minor typos found by Matt L.
wd-15	2006-07-28	Doug Davis	Issue 113 applied
wd-15	2006-08-04	Doug Davis	Update old namespaces – found by PaulC
wd-15	2006-08-04	Doug Davis	Issue 150 applied
wd-15	2006-08-04	Doug Davis	Minor typos – found by PeterN
wd-15	2006-08-04	Doug Davis	Verify all [refs]
wd-15	2006-08-04	Doug Davis	Change namespace to 2006/08
wd-15	2006-08-04	Doug Davis	Issue 148 applied
wd-15	2006-08-07	Doug Davis	Add some new glossary terms – per GilP
cd-04	2006-08-10	Gilbert Pilz	Formatting changes for better HTML rendering.

Rev	Date	By Whom	What
cd-04	2006-08-11	Doug Davis	Issue 158 applied
cd-04	2006-08-11	Doug Davis	Issue 153 applied
cd-04	2006-08-11	Doug Davis	Issue 156 applied
cd-04	2006-08-15	Gilbert Pilz	More formatting changes for better HTML rendering.
wd-16	2006-10-25	Doug Davis	Accept all changes, update to wd16
wd-16	2006-10-26	Doug Davis	PR002 applied
wd-16	2006-10-26	Doug Davis	PR003 applied
wd-16	2006-10-26	Doug Davis	PR004 applied
wd-16	2006-10-27	Doug Davis	PR005 applied
wd-16	2006-10-27	Doug Davis	PR006 applied
wd-16	2006-10-27	Doug Davis	PR024 applied
wd-16	2006-11-13	Doug Davis	PR010 applied
wd-16	2006-11-13	Doug Davis	PR011 applied (technically as part of PR004)
wd-16	2006-11-13	Doug Davis	PR016 applied
wd-16	2006-11-13	Doug Davis	PR032 applied
wd-16	2006-11-20	Doug Davis	PR025 applied
wd-16	2006-11-20	Doug Davis	PR023 applied
wd-16	2006-12-03	Doug Davis	PR036 applied
wd-16	2006-12-03	Doug Davis	PR017 applied
wd-16	2006-12-11	Doug Davis	PR012 applied
wd-16	2006-12-14	Doug Davis	PR033 applied – changed a 'return' to 'generate' when talking about a fault
wd-16	2007-01-04	Doug Davis	PR018 applied
wd-16	2007-01-05	Doug Davis	Moved MakeConnection to new spec

2095 **Appendix G. Notices**

2096 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
2097 might be claimed to pertain to the implementation or use of the technology described in this document or
2098 the extent to which any license under such rights might or might not be available; neither does it represent
2099 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to
2100 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made
2101 available for publication and any assurances of licenses to be made available, or the result of an attempt
2102 made to obtain a general license or permission for the use of such proprietary rights by implementors or
2103 users of this specification, can be obtained from the OASIS Executive Director.

2104 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or
2105 other proprietary rights which may cover technology that may be required to implement this specification.
2106 Please address the information to the OASIS Executive Director.

2107 Copyright (C) OASIS Open (2006). All Rights Reserved.

2108 This document and translations of it may be copied and furnished to others, and derivative works that
2109 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and
2110 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and
2111 this paragraph are included on all such copies and derivative works. However, this document itself may
2112 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as
2113 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights
2114 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it
2115 into languages other than English.

2116 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
2117 or assigns.

2118 This document and the information contained herein is provided on an "AS IS" basis and OASIS
2119 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
2120 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR
2121 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.