



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 Yalçinalp, 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.

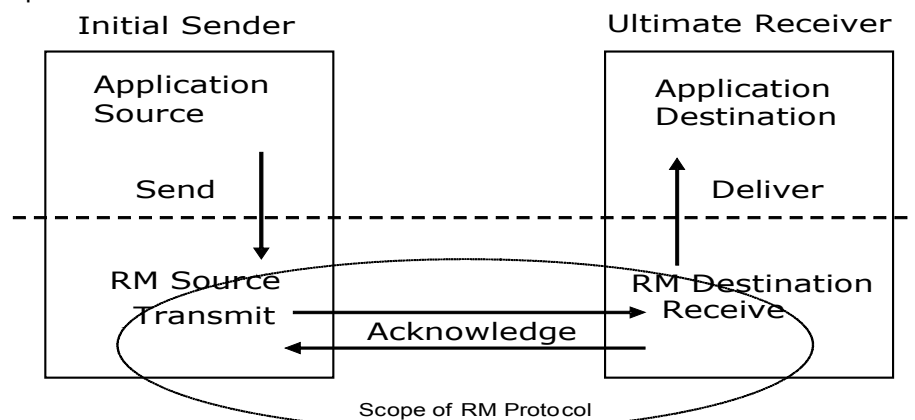
161 2 Reliable Messaging Model

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.

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

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

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



187 Figure 1: Reliable Messaging Model

188 2.1 Glossary

189 The following definitions are used throughout this specification:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

218 **2.2 Protocol Preconditions**

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

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

228 2.3 Protocol Invariants

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

- 230 • The RM Source MUST assign each message within a Sequence a message number (defined
- 231 below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers
- 232 MUST be assigned in the same order in which messages are sent by the Application Source.

- 233 • Within every Acknowledgement Message it issues, the RM Destination MUST include one or more
- 234 AcknowledgementRange child elements that contain, in their collective ranges, the message
- 235 number of every message accepted by the RM Destination. The RM Destination MUST exclude, in
- 236 the AcknowledgementRange elements, the message numbers of any messages it has not
- 237 accepted. If no messages have been received the RM Destination MUST return None instead of an
- 238 AcknowledgementRange (s). The RM Destination MAY transmit a Nack for a specific message
- 239 or messages in stead of an AcknowledgementRange (s).

240 2.4 Example Message Exchange

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

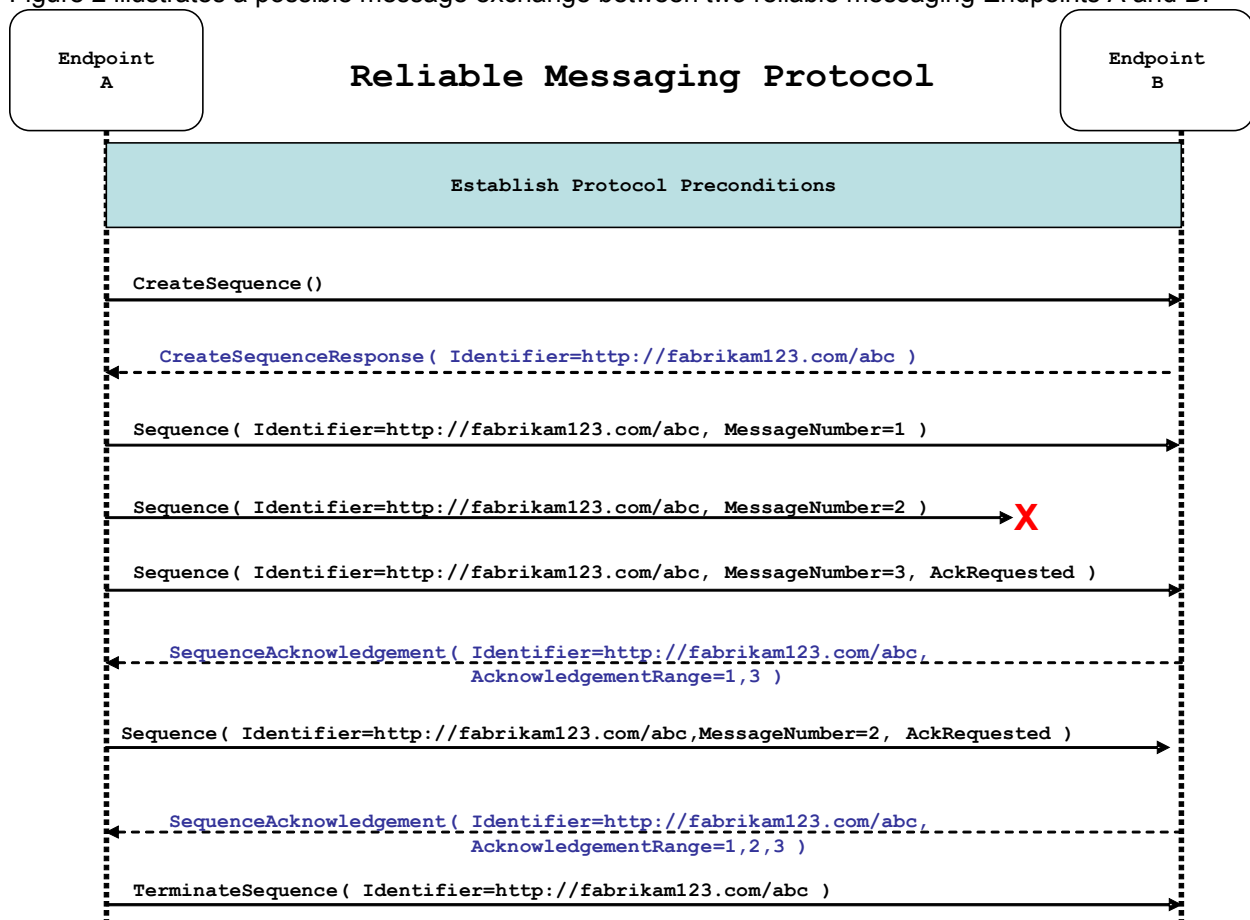


Figure 2: The WS-ReliableMessaging Protocol

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

- 244 2. The RM Source requests creation of a new Sequence.
- 245 3. The RM Destination creates a new Sequence and returns its unique identifier.
- 246 4. The RM Source begins Transmitting messages in the Sequence beginning with MessageNumber 1.
247 In the figure above, the RM Source sends 3 messages in the Sequence.
- 248 5. The 2nd message in the Sequence is lost in transit.
- 249 6. The 3rd message is the last in this Sequence and the RM Source includes an `AckRequested`
250 header to ensure that it gets a timely `SequenceAcknowledgement` for the Sequence.
- 251 7. The RM Destination acknowledges receipt of message numbers 1 and 3 as a result of receiving the
252 RM Source's `AckRequested` header.
- 253 8. The RM Source retransmits the unacknowledged message with MessageNumber 2. This is a new
254 message from the perspective of the underlying transport, but it has the same Sequence Identifier
255 and MessageNumber so the RM Destination can recognize it as a duplicate of the earlier message,
256 in case the original and retransmitted messages are both Received. The RM Source includes an
257 `AckRequested` header in the retransmitted message so the RM Destination will expedite an
258 acknowledgement.
- 259 9. The RM Destination Receives the second transmission of the message with MessageNumber 2
260 and acknowledges receipt of message numbers 1, 2, and 3.
- 261 10. The RM Source Receives this Acknowledgement and sends a `TerminateSequence` message to the
262 RM Destination indicating that the Sequence is completed and reclaims any resources associated
263 with the Sequence.
- 264 11. The RM Destination Receives the `TerminateSequence` message indicating that the RM Source will
265 not be sending any more messages. The RM Destination sends a `TerminateSequenceResponse`
266 message to the RM Source and reclaims any resources associated with the Sequence.
- 267 The RM Source will expect to Receive Acknowledgements from the RM Destination during the course of a
268 message exchange at occasions described in Section 3 below. Should an Acknowledgement not be
269 Received in a timely fashion, the RM Source MUST re-transmit the message since either the message or
270 the associated Acknowledgement might have been lost. Since the nature and dynamic characteristics of
271 the underlying transport and potential intermediaries are unknown in the general case, the timing of re-
272 transmissions cannot be specified. Additionally, over-aggressive re-transmissions have been
273 demonstrated to cause transport or intermediary flooding which are counterproductive to the intention of
274 providing a reliable exchange of messages. Consequently, implementers are encouraged to utilize
275 adaptive mechanisms that dynamically adjust re-transmission time and the back-off intervals that are
276 appropriate to the nature of the transports and intermediaries envisioned. For the case of TCP/IP
277 transports, a mechanism similar to that described as RTTM in RFC 1323 [[RTTM](#)] SHOULD be
278 considered.
- 279 Now that the basic model has been outlined, the details of the elements used in this protocol are now
280 provided in Section 3.

281 **3 RM Protocol Elements**

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

284 **3.1 Considerations on the Use of Extensibility Points**

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

289 **3.2 Considerations on the Use of "Piggy-Backing"**

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

295 **3.3 Composition with WS-Addressing**

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

- 298 1. When an Endpoint generates a message that carries an RM protocol element, that is defined in
299 section 3 below, in the body of a SOAP envelope that Endpoint MUST include in that envelope a
300 `wsa:Action` SOAP header block whose value is an IRI that is a concatenation of the WS-RM
301 namespace URI, followed by a "/", followed by the value of the local name of the child element of
302 the SOAP body. For example, for a Sequence creation request message as described in section
303 3.4 below, the value of the `wsa:Action` IRI would be:

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

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

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

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

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

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

313 **3.4 Sequence Creation**

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

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

321 The following exemplar defines the `CreateSequence` syntax:

```
322 <wsrm:CreateSequence ...>
323   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>
324   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
325   <wsrm:Offer ...>
326     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
327     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>
328     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
329     <wsrm:IncompleteSequenceBehavior>
330       wsrml:IncompleteSequenceBehaviorType
331     </wsrm:IncompleteSequenceBehavior> ?
332     ...
333   </wsrm:Offer> ?
334   ...
335 </wsrm:CreateSequence>
```

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

337 `/wsrm:CreateSequence`

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

342 `/wsrm:CreateSequence/wsrm:AcksTo`

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

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

352 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

360 `/wsrm:CreateSequence/wsrm:Offer`

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

363 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier`

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

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

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

369 /wsmr:CreateSequence/wsmr:Offer/wsmr:Endpoint

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

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

381 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

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

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

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

388 /wsmr:CreateSequence/wsmr:Offer/wsmr:IncompleteSequenceBehavior

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

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

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

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

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

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

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

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

405 /wsmr:CreateSequence/{any}

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

408 /wsmr:CreateSequence/@{any}

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

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

415 The following exemplar defines the `CreateSequenceResponse` syntax:

```
416 <wsmr:CreateSequenceResponse ...>  
417   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
418   <wsmr:Expires ...> xs:duration </wsmr:Expires> ?  
419   <wsmr:IncompleteSequenceBehavior>  
420     wsmr:IncompleteSequenceBehaviorType  
421   </wsmr:IncompleteSequenceBehavior> ?  
422   <wsmr:Accept ...>  
423     <wsmr:AcksTo wsa:EndpointReferenceType </wsmr:AcksTo>  
424     ...  
425   </wsmr:Accept> ?  
426   ...  
427 </wsmr:CreateSequenceResponse>
```

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

429 /wsmr:CreateSequenceResponse

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

433 /wsmr:CreateSequenceResponse/wsmr:Identifier

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

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

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

440 /wsmr:CreateSequenceResponse/wsmr:Expires

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

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

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

452 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`

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

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

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

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

463 `/wsrm:CreateSequenceResponse/wsrm:Accept`

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

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

469 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

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

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

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

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

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

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

484 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

490 3.5 Closing A Sequence

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

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

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

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

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

518 The following exemplar defines the `CloseSequence` syntax:

```
519 <wsmr:CloseSequence ...>  
520   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
521   ...  
522 </wsmr:CloseSequence>
```

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

524 `/wsmr:CloseSequence`

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

528 `/wsmr:CloseSequence/wsmr:Identifier`

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

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

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

535 /wsmr:CloseSequence/{any}

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

538 /wsmr:CloseSequence@{any}

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

541 A `CloseSequenceResponse` is sent in the body of a message in response to receipt of a
542 `CloseSequence` request message. It indicates that the responding party has closed the Sequence.

543 The following exemplar defines the `CloseSequenceResponse` syntax:

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

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

549 /wsmr:CloseSequenceResponse

550 This element is sent in the body of a message in response to receipt of a `CloseSequence` request
551 message. It indicates that the responding party has closed the Sequence.

552 /wsmr:CloseSequenceResponse/wsmr:Identifier

553 The responding party (RMS or RMD) MUST include this element in any `CloseSequenceResponse`
554 message it sends. The responding party MUST set the value of this element to the absolute URI
555 (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:

```
577 <wsm:TerminateSequence ...>  
577   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>  
577   ...  
577 </wsm:TerminateSequence>
```

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

577 `/wsm:TerminateSequence`

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

577 This element MAY also be sent by the RM Destination to indicate that it has unilaterally terminated the
578 Sequence. Upon sending this message the RM Destination MUST NOT accept any additional messages
579 (with the exception of the corresponding `TerminateSequenceResponse`) for this Sequence. Upon
580 receipt of a `TerminateSequence` the RM Source MUST NOT send any additional messages (with the
581 exception of the corresponding `TerminateSequenceResponse`) for this Sequence.

577 `/wsm:TerminateSequence/wsm:Identifier`

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

577 `/wsm:TerminateSequence/wsm:Identifier/@{any}`

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

577 `/wsm:TerminateSequence/{any}`

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

577 `/wsm:TerminateSequence/@{any}`

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

577 A `TerminateSequenceResponse` is sent in the body of a message in response to receipt of a
578 `TerminateSequence` request message. It indicates that the responding party has terminated the
579 Sequence.

577 The following exemplar defines the `TerminateSequenceResponse` syntax:

```
578 <wsm:TerminateSequenceResponse ...>  
579   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>  
580   ...  
581 </wsm:TerminateSequenceResponse>
```

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

583 `/wsrm:TerminateSequenceResponse`

584 This element is sent in the body of a message in response to receipt of a `TerminateSequence` request
585 message. It indicates that the RM Destination has terminated the Sequence. The RM Destination MUST
586 NOT send this element as a header block.

587 `/wsrm:TerminateSequenceResponse/wsrm:Identifier`

588 The responding party (RMS or RMD) MUST include this element in any `TerminateSequenceResponse`
589 message it sends. The responding party MUST set the value of this element to the absolute URI
590 (conformant with RFC3986) of the Sequence that is being terminated.

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

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

594 `/wsrm:TerminateSequenceResponse/{any}`

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

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

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

600 On receipt of a `TerminateSequence` message the receiving party (RMS or RMD) MUST respond with a
601 corresponding `TerminateSequenceResponse` message or generate a fault `UnknownSequenceFault`
602 if the Sequence is not known.

603 **3.7 Sequences**

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

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

611 A following exemplar defines its syntax:

```
612 <wsrm:Sequence ...>  
613   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
614   <wsrm:MessageNumber> wsrm:MessageNumberType </wsrm:MessageNumber>  
615   ...  
616 </wsrm:Sequence>
```

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

618 `/wsrm:Sequence`

619 This protocol element associates the message in which it is contained with a previously established RM
620 Sequence. It contains the Sequence's unique identifier and the containing message's ordinal position
621 within that Sequence. The RM Destination MUST understand the `Sequence` header block. The RM
622 Source MUST assign a `mustUnderstand` attribute with a value 1/true (from the namespace

623 corresponding to the version of SOAP to which the `Sequence` SOAP header block is bound) to the
624 `Sequence` header block element.

625 `/wsmr:Sequence/wsmr:Identifier`

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

629 `/wsmr:Sequence/wsmr:Identifier/@{any}`

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

632 `/wsmr:Sequence/wsmr:MessageNumber`

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

637 `/wsmr:Sequence/{any}`

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

640 `/wsmr:Sequence/@{any}`

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

643 The following example illustrates a `Sequence` header block.

```
644 <wsmr:Sequence>  
645   <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>  
646   <wsmr:MessageNumber>10</wsmr:MessageNumber>  
647 </wsmr:Sequence>
```

648 **3.8 Request Acknowledgement**

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

651 The RM Source MAY request an `Acknowledgement Message` from the RM Destination at any time by
652 transmitting an `AckRequested` header block independently or it MAY include an `AckRequested` header
653 block in any message targeted to the RM Destination. An RM Destination that Receives a message that
654 contains an `AckRequested` header block MUST send a message containing a
655 `SequenceAcknowledgement` header block to the `AcksTo` endpoint reference (see Section 3.4) for a
656 known `Sequence` or else generate an `UnknownSequence` fault. If a non-mustUnderstand fault occurs
657 when processing an RM header that was piggy-backed on another message, a fault MUST be generated,
658 but the processing of the original message MUST NOT be affected. It is RECOMMENDED that the RM
659 Destination return a `AcknowledgementRange` or `None` element instead of a `Nack` element (see Section
660 3.9).

661 The following exemplar defines its syntax:

```
662 <wsmr:AckRequested ...>  
663   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
664   ...
```

```
665 </wsm:AckRequested>
```

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

```
667 /wsm:AckRequested
```

668 This element requests an Acknowledgement for the identified Sequence.

```
669 /wsm:AckRequested/wsm:Identifier
```

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

```
673 /wsm:AckRequested/wsm:Identifier/@{any}
```

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

```
676 /wsm:AckRequested/{any}
```

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

```
679 /wsm:AckRequested/@{any}
```

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

682 3.9 Sequence Acknowledgement

683 The RM Destination informs the RM Source of successful message receipt using a
684 `SequenceAcknowledgement` header block. The RM Destination MAY Transmit the
685 `SequenceAcknowledgement` header block independently or it MAY include the
686 `SequenceAcknowledgement` header block on any message targeted to the `AcksTo` EPR.
687 Acknowledgements can be explicitly requested using the `AckRequested` directive (see Section 3.8). If a
688 non-mustUnderstand fault occurs when processing an RM header that was piggy-backed on another
689 message, a fault MUST be generated, but the processing of the original message MUST NOT be
690 affected.

691 A RM Destination MAY include a `SequenceAcknowledgement` header block on any SOAP envelope
692 targeted to the endpoint referenced by the `AcksTo` EPR.

693 During creation of a Sequence the RM Source MAY specify the WS-Addressing anonymous IRI as the
694 address of the `AcksTo` EPR for that Sequence. When the RM Source specifies the WS-Addressing
695 anonymous IRI as the address of the `AcksTo` EPR, the RM Destination MUST Transmit any
696 `SequenceAcknowledgement` headers for the created Sequence in a SOAP envelope to be Transmitted
697 on the protocol binding-specific back-channel. Such a channel is provided by the context of a Received
698 message containing a SOAP envelope that contains a `Sequence` header block and/or an `AckRequested`
699 header block for that same Sequence identifier. When the RM Destination receives an `AckRequested`
700 header, and the `AckTo` EPR for that sequence is the WS-Addressing anonymous IRI, the RM Destination
701 SHOULD respond on the protocol binding-specific back-channel provided by the Received message
702 containing the `AckRequested` header block.

703 The following exemplar defines its syntax:

```
704 <wsm:SequenceAcknowledgement ...>  
705   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>
```

```

706 [ [ [ <wsm:AcknowledgementRange ...
707         Upper="wsm:MessageType"
708         Lower="wsm:MessageType"/> +
709         | <wsm:None/> ]
710         <wsm:Final/> ? ]
711         | <wsm:Nack> wsm:MessageType </wsm:Nack> + ]
712
713     ...
714 </wsm:SequenceAcknowledgement>

```

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

716 `/wsm:SequenceAcknowledgement`

717 This element contains the Sequence Acknowledgement information.

718 `/wsm:SequenceAcknowledgement/wsm:Identifier`

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

724 `/wsm:SequenceAcknowledgement/wsm:Identifier/@{any}`

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

727 `/wsm:SequenceAcknowledgement/wsm:AcknowledgementRange`

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

733 `/wsm:SequenceAcknowledgement/wsm:AcknowledgementRange/@Upper`

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

736 `/wsm:SequenceAcknowledgement/wsm:AcknowledgementRange/@Lower`

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

739 `/wsm:SequenceAcknowledgement/wsm:AcknowledgementRange/@{any}`

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

742 `/wsm:SequenceAcknowledgement/wsm:None`

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

747 `/wsm:SequenceAcknowledgement/wsm:Final`

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

754 `/wsrm:SequenceAcknowledgement/wsrm:Nack`

755 The RM Destination MAY include this element within a `SequenceAcknowledgement` header block. If
756 used, the RM Destination MUST set the value of this element to a `MessageNumberType` representing
757 the `MessageNumber` of an unreceived message in a Sequence. The RM Destination MUST NOT include
758 a `Nack` element if a sibling `AcknowledgementRange` or `None` element is also present as a child of
759 `SequenceAcknowledgement`. Upon the receipt of a `Nack`, an RM Source SHOULD retransmit the
760 message identified by the `Nack`. The RM Destination MUST NOT issue a `SequenceAcknowledgement`
761 containing a `Nack` for a message that it has previously acknowledged within a
762 `AcknowledgementRange`. The RM Source SHOULD ignore a `SequenceAcknowledgement` containing
763 a `Nack` for a message that has previously been acknowledged within a `AcknowledgementRange`.

764 `/wsrm:SequenceAcknowledgement/{any}`

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

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

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

770 The following examples illustrate `SequenceAcknowledgement` elements:

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

```
772 <wsrm:SequenceAcknowledgement>  
773   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
774   <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
775 </wsrm:SequenceAcknowledgement>
```

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

```
778 <wsrm:SequenceAcknowledgement>  
779   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
780   <wsrm:AcknowledgementRange Upper="2" Lower="1"/>  
781   <wsrm:AcknowledgementRange Upper="6" Lower="4"/>  
782   <wsrm:AcknowledgementRange Upper="10" Lower="8"/>  
783 </wsrm:SequenceAcknowledgement>
```

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

```
785 <wsrm:SequenceAcknowledgement>  
786   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
787   <wsrm:Nack>3</wsrm:Nack>  
788 </wsrm:SequenceAcknowledgement>
```

789 4 Faults

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

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

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

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

803 The definitions of faults use the following properties:

804 [Code] The fault code.

805 [Subcode] The fault subcode.

806 [Reason] The English language reason element.

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

810 Entities that generate WS-ReliableMessaging faults MUST set the [Code] property to either "Sender" or
811 "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

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

```
813 <S:Envelope>
814   <S:Header>
815     <wsa:Action>
816       http://docs.oasis-open.org/ws-rx/wsrn/200608/fault
817     </wsa:Action>
818     <!-- Headers elided for brevity. -->
819   </S:Header>
820   <S:Body>
821     <S:Fault>
822       <S:Code>
823         <S:Value> [Code] </S:Value>
824         <S:Subcode>
825           <S:Value> [Subcode] </S:Value>
826         </S:Subcode>
827       </S:Code>
828       <S:Reason>
829         <S:Text xml:lang="en"> [Reason] </S:Text>
830       </S:Reason>
831     <S:Detail>
```

```

832     [Detail]
833     ...
834     </S:Detail>
835     </S:Fault>
836     </S:Body>
837 </S:Envelope>

```

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

```

840 <S11:Envelope>
841   <S11:Header>
842     <wsrm:SequenceFault>
843       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
844       <wsrm:Detail> [Detail] </wsrm:Detail>
845       ...
846     </wsrm:SequenceFault>
847     <!-- Headers elided for brevity. -->
848   </S11:Header>
849   <S11:Body>
850     <S11:Fault>
851       <faultcode> [Code] </faultcode>
852       <faultstring> [Reason] </faultstring>
853     </S11:Fault>
854   </S11:Body>
855 </S11:Envelope>

```

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

```

858 <S11:Envelope>
859   <S11:Body>
860     <S11:Fault>
861       <faultcode> [Subcode] </faultcode>
862       <faultstring> [Reason] </faultstring>
863     </S11:Fault>
864   </S11:Body>
865 </S11:Envelope>

```

866 4.1 SequenceFault Element

867 The purpose of the `SequenceFault` element is to carry the specific details of a fault generated during
868 the reliable messaging specific processing of a message belonging to a Sequence. WS-
869 ReliableMessaging nodes MUST use the `SequenceFault` container only in conjunction with the SOAP
870 1.1 fault mechanism. WS-ReliableMessaging nodes MUST NOT use the `SequenceFault` container in
871 conjunction with the SOAP 1.2 binding.

872 The following exemplar defines its syntax:

```

873 <wsrm:SequenceFault ...>
874   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
875   <wsrm:Detail> ... </wsrm:Detail> ?
876   ...
877 </wsrm:SequenceFault>

```

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

879 `/wsrm:SequenceFault`

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

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

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

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

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

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

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

896 4.2 Sequence Terminated

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

899 Properties:

900 [Code] Sender or Receiver
 901 [Subcode] wsm:SequenceTerminated
 902 [Reason] The Sequence has been terminated due to an unrecoverable error.
 903 [Detail]

904 `<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.

905 4.3 Unknown Sequence

906 Properties:

907 [Code] Sender
 908 [Subcode] wsm:UnknownSequence

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

910 [Detail]

```
911 <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.

912 4.4 Invalid Acknowledgement

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

915 [Code] Sender

916 [Subcode] wsrn:InvalidAcknowledgement

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

918 [Detail]

```
919 <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.

920 4.5 Message Number Rollover

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

922 Properties:

923 [Code] Sender

924 [Subcode] wsrn:MessageNumberRollover

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

926 [Detail]

```
927 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
928 <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.

929 4.6 Create Sequence Refused

930 Properties:

931 [Code] Sender or Receiver

932 [Subcode] wsrm:CreateSequenceRefused

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

934 [Detail]

```
935 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.

936 4.7 Sequence Closed

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

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

940 Properties:

941 [Code] Sender

942 [Subcode] wsrm:SequenceClosed

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

944 [Detail]

945 `<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.

946 **4.8 WSRM Required**

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

949 Properties:

950 [Code] Sender

951 [Subcode] wsrm:WSRMRequired

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

953 [Detail]

954 `xs:any`

955 **5 Security Threats and Countermeasures**

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

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

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

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

974 **5.1 Threats and Countermeasures**

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

980 **5.1.1 Integrity Threats**

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

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

989 **5.1.1.1 Countermeasures**

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

995 **5.1.2 Resource Consumption Threats**

996 The creation of a Sequence with an RM Destination consumes various resources on the systems used to
997 implement that RM Destination. These resources can include network connections, database tables,
998 message queues, etc. This behavior can be exploited to conduct denial of service attacks against an RM
999 Destination. For example, a simple attack is to repeatedly send `CreateSequence` messages to an RM
1000 Destination. Another attack is to create a Sequence for a service that is known to require in-order
1001 message Delivery and use this Sequence to send a stream of very large messages to that service,
1002 making sure to omit message number “1” from that stream.

1003 **5.1.2.1 Countermeasures**

1004 There are a number of countermeasures against the described resource consumption threats. The
1005 technique advocated by this specification is for the RM Destination to restrict the ability to create a
1006 Sequence to a specific set of entities/principals. This reduces the number of potential attackers and, in
1007 some cases, allows the identity of any attackers to be determined.

1008 The ability to restrict Sequence creation depends, in turn, upon the RM Destination's ability identify and
1009 authenticate the RM Source that issued the `CreateSequence` message.

1010 **5.1.3 Sequence Spoofing Threats**

1011 Sequence spoofing is a class of threats in which the attacker uses knowledge of the `Identifier` for a
1012 particular Sequence to forge Sequence Lifecycle or Traffic Messages. For example the attacker creates a
1013 fake `TerminateSequence` message that references the target Sequence and sends this message to the
1014 appropriate RM Destination. Some sequence spoofing attacks also require up-to-date knowledge of the
1015 current `MessageNumber` for their target Sequence.

1016 In general any Sequence Lifecycle Message, RM Protocol Header Block, or sequence-correlated SOAP
1017 fault (e.g. `InvalidAcknowledgement`) can be used by someone with knowledge of the Sequence identifier
1018 to attack the Sequence. These attacks are “two-way” in that an attacker may choose to target the RM
1019 Source by, for example, inserting a fake `SequenceAcknowledgement` header into a message that it sends
1020 to the `AcksTo` EPR of an RM Source.

1021 **5.1.3.1 Sequence Hijacking**

1022 Sequence hijacking is a specific case of a sequence spoofing attack. The attacker attempts to inject
1023 Sequence Traffic Messages into an existing Sequence by inserting fake `Sequence` headers into those
1024 messages.

1025 Note that “sequence hijacking” should not be equated with “security session hijacking”. Although a
1026 Sequence may be bound to some form of a security session in order to counter the threats described in
1027 this section, applications MUST NOT rely on WS-RM-related information to make determinations about
1028 the identity of the entity that created a message; applications SHOULD rely only upon information that is
1029 established by the security infrastructure to make such determinations. Failure to observe this rule
1030 creates, among other problems, a situation in which the absence of WS-RM may deprive an application of
1031 the ability to authenticate its peers even though the necessary security processing has taken place.

1032 **5.1.3.2 Countermeasures**

1033 There are a number of countermeasures against sequence spoofing threats. The technique advocated by
1034 this specification is to consider the Sequence to be a shared resource that is jointly owned by the RM

1035 Source that initiated its creation (i.e. that sent the `CreateSequence` message) and the RM Destination that
1036 serves as its terminus (i.e. that sent the `CreateSequenceResponse` message). To counter sequence
1037 spoofing attempts the RM Destination SHOULD ensure that every message or fault that it Receives that
1038 refers to a particular Sequence originated from the RM Source that jointly owns the referenced Sequence.
1039 For its part the RM Source SHOULD ensure that every message or fault that it Receives that refers to a
1040 particular Sequence originated from the RM Destination that jointly owns the referenced Sequence.

1041 For the RM Destination to be able to identify its sequence peer it MUST be able to identify and
1042 authenticate the entity that sent the `CreateSequence` message. Similarly for the RM Source to identify its
1043 sequence peer it MUST be able to identify and authenticate the entity that sent the
1044 `CreateSequenceResponse` message. For either the RM Destination or the RM Source to determine if a
1045 message was sent by its sequence peer it MUST be able to identify and authenticate the initiator of that
1046 message and, if necessary, correlate this identity with the sequence peer identity established at sequence
1047 creation time.

1048 **5.2 Security Solutions and Technologies**

1049 The security threats described in the previous sections are neither new nor unique. The solutions that
1050 have been developed to secure other SOAP-based protocols can be used to secure WS-RM as well. This
1051 section maps the facilities provided by common web services security solutions against countermeasures
1052 described in the previous sections.

1053 Before continuing this discussion, however, some examination of the underlying requirements of the
1054 previously described countermeasures is necessary. Specifically it should be noted that the technique
1055 described in Section 5.1.2.1 has two components. Firstly, the RM Destination identifies and authenticates
1056 the issuer of a `CreateSequence` message. Secondly, the RM Destination performs an authorization check
1057 against this authenticated identity and determines if the RM Source is permitted to create Sequences with
1058 the RM Destination. Since the facilities for performing this authorization check (runtime infrastructure,
1059 policy frameworks, etc.) lie completely within the domain of individual implementations, any discussion of
1060 such facilities is considered to be beyond the scope of this specification.

1061 **5.2.1 Transport Layer Security**

1062 This section describes how the facilities provided by SSL/TLS [RFC 4346] can be used to implement the
1063 countermeasures described in the previous sections. The use of SSL/TLS is subject to the constraints
1064 defined in Section 4 of the Basic Security Profile 1.0 [BSP 1.0].

1065 The description provided here is general in nature and is not intended to serve as a complete definition on
1066 the use of SSL/TLS to protect WS-RM. In order to interoperate implementations need to agree on the
1067 choice of features as well as the manner in which they will be used. The mechanisms described in the
1068 Web Services Security Policy Language [SecurityPolicy] MAY be used by services to describe the
1069 requirements and constraints of the use of SSL/TLS.

1070 **5.2.1.1 Model**

1071 The basic model for using SSL/TLS is as follows:

- 1072 1. The RM Source establishes an SSL/TLS session with the RM Destination.
- 1073 2. The RM Source uses this SSL/TLS session to send a `CreateSequence` message to the RM
1074 Destination.

- 1075 3. The RM Destination establishes an SSL/TLS session with the RM Source and sends an
1076 asynchronous `CreateSequenceResponse` using this session. Alternately it may respond with a
1077 synchronous `CreateSequenceResponse` using the session established in (1).
- 1078 4. For the lifetime of the Sequence the RM Source uses the SSL/TLS session from (1) to Transmit
1079 any and all messages or faults that refer to that Sequence.
- 1080 5. For the lifetime of the Sequence the RM Destination either uses the SSL/TLS session established
1081 in (3) to Transmit any and all messages or faults that refer to that Sequence or, for synchronous
1082 exchanges, the RM Destination uses the SSL/TLS session established in (1).

1083 5.2.1.2 Countermeasure Implementation

1084 Used in its simplest fashion (without relying upon any authentication mechanisms), SSL/TLS provides the
1085 necessary integrity qualities to counter the threats described in Section 5.1.1. Note, however, that the
1086 nature of SSL/TLS limits the scope of this integrity protection to a single transport level session. If
1087 SSL/TLS is the only mechanism used to provide integrity, any intermediaries between the RM Source and
1088 the RM Destination MUST be trusted to preserve the integrity of the messages that flow through them.

1089 As noted, the technique described in Sections 5.1.2.1 involves the use of authentication. This specification
1090 advocates either of two mechanisms for authenticating entities using SSL/TLS. In both of these methods
1091 the SSL/TLS server (the party accepting the SSL/TLS connection) authenticates itself to the SSL/TLS
1092 client using an X.509 certificate that is exchanged during the SSL/TLS handshake.

- 1093 • **HTTP Basic Authentication:** This method of authentication presupposes that a SOAP/HTTP
1094 binding is being used as part of the protocol stack beneath WS-RM. Subsequent to the
1095 establishment of the SSL/TLS session, the sending party authenticates itself to the receiving party
1096 using HTTP Basic Authentication [RFC 2617]. For example, a RM Source might authenticate itself
1097 to a RM Destination (e.g. when transmitting a Sequence Traffic Message) using BasicAuth.
1098 Similarly the RM Destination might authenticate itself to the RM Source (e.g. when sending an
1099 Acknowledgement) using BasicAuth.
- 1100 • **SSL/TLS Client Authentication:** In this method of authentication, the party initiating the
1101 connection authenticates itself to the party accepting the connection using an X.509 certificate
1102 that is exchanged during the SSL/TLS handshake.

1103 To implement the countermeasures described in section 5.1.2.1 the RM Source must authenticate itself
1104 using one the above mechanisms. The authenticated identity can then be used to determine if the RM
1105 Source is authorized to create a Sequence with the RM Destination.

1106 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring
1107 an RM node's Sequence peer to be equivalent to their SSL/TLS session peer. This allows the
1108 authorization decisions described in section 5.1.3.2 to be based on SSL/TLS session identity rather than
1109 on authentication information. For example, an RM Destination can determine that a Sequence Traffic
1110 Message rightfully belongs to its referenced Sequence if that message arrived over the same SSL/TLS
1111 session that was used to carry the `CreateSequence` message for that Sequence. Note that requiring a
1112 one-to-one relationship between SSL/TLS session peer and Sequence peer constrains the lifetime of a
1113 SSL/TLS-protected Sequence to be less than or equal to the lifetime of the SSL/TLS session that is used
1114 to protect that Sequence.

1115 This specification does not preclude the use of other methods of using SSL/TLS to implement the
1116 countermeasures (such as associating specific authentication information with a Sequence) although such
1117 methods are not covered by this document.

1118 Issues specific to the life-cycle management of SSL/TLS sessions (such as the resumption of a SSL/TLS
1119 session) are outside the scope of this specification.

1120 **5.2.2 SOAP Message Security**

1121 The mechanisms described in WS-Security may be used in various ways to implement the
1122 countermeasures described in the previous sections. This specification advocates using the protocol
1123 described by WS-SecureConversation [[SecureConversation](#)] (optionally in conjunction with WS-Trust
1124 [[Trust](#)]) as a mechanism for protecting Sequences. The use of WS-Security (as an underlying component
1125 of WS-SecureConversation) is subject to the constraints defined in the Basic Security Profile 1.0.

1126 The description provided here is general in nature and is not intended to serve as a complete definition on
1127 the use of WS-SecureConversation/WS-Trust to protect WS-RM. In order to interoperate implementations
1128 need to agree on the choice of features as well as the manner in which they will be used. The
1129 mechanisms described in the Web Services Security Policy Language MAY be used by services to
1130 describe the requirements and constraints of the use of WS-SecureConversation.

1131 **5.2.2.1 Model**

1132 The basic model for using WS-SecureConversation is as follows:

- 1133 1. The RM Source and the RM Destination create a WS-SecureConversation security context. This
1134 may involve the participation of third parties such as a security token service. The tokens
1135 exchanged may contain authentication claims (e.g. X.509 certificates or Kerberos service tickets).
- 1136 2. During the `CreateSequence` exchange, the RM Source SHOULD explicitly identify the security
1137 context that will be used to protect the Sequence. This is done so that, in cases where the
1138 `CreateSequence` message is signed by more than one security context, the RM Source can
1139 indicate which security context should be used to protect the newly created Sequence.
- 1140 3. For the lifetime of the Sequence the RM Source and the RM Destination use the session key(s)
1141 associated with the security context to sign (as defined by WS-Security) at least the body and any
1142 relevant WS-RM-defined headers of any and all messages or faults that refer to that Sequence.

1143 **5.2.2.2 Countermeasure Implementation**

1144 Without relying upon any authentication information, the per-message signatures provide the necessary
1145 integrity qualities to counter the threats described in Section 5.1.1.

1146 To implement the countermeasures described in section 5.1.2.1 some mutually agreed upon form of
1147 authentication claims must be provided by the RM Source to the RM Destination during the establishment
1148 of the Security Context. These claims can then be used to determine if the RM Source is authorized to
1149 create a Sequence with the RM Destination.

1150 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring
1151 an RM node's Sequence peer to be equivalent to their security context session peer. This allows the
1152 authorization decisions described in section 5.1.3.2 to be based on the identity of the message's security
1153 context rather than on any authentication claims that may have been established during security context
1154 initiation. Note that other methods of using WS-SecureConversation to implement the countermeasures
1155 (such as associating specific authentication claims to a Sequence) are possible but not covered by this
1156 document.

1157 As with transport security, the requisite equivalence of a security context peer and with a Sequence peer
1158 limits the lifetime of a Sequence to the lifetime of the protecting security context. Unlike transport security,

1159 the association between a Sequence and its protecting security context cannot always be established
1160 implicitly at Sequence creation time. This is due to the fact that the `CreateSequence` and
1161 `CreateSequenceResponse` messages may be signed by more than one security context.

1162 Issues specific to the life-cycle management of WS-SecureConversation security contexts (such as
1163 amending or renewing contexts) are outside the scope of this specification.

1164 6 Securing Sequences

1165 As noted in Section 5, the RM Source and RM Destination should be able to protect their shared
1166 Sequences against the threat of Sequence Spoofing attacks. There are a number of OPTIONAL means of
1167 achieving this objective depending upon the underlying security infrastructure.

1168 6.1 Securing Sequences Using WS-Security

1169 One mechanism for protecting a Sequence is to include a security token using a
1170 `wsse:SecurityTokenReference` element from WS-Security (see section 9 in WS-
1171 SecureConversation) in the `CreateSequence` element. This establishes an association between the
1172 created (and, if present, offered) Sequence(s) and the referenced security token, such that the RM Source
1173 and Destination MUST use the security token as the basis for authorization of all subsequent interactions
1174 related to the Sequence(s). The `wsse:SecurityTokenReference` explicitly identifies the token as
1175 there may be more than one token on a `CreateSequence` message or inferred from the communication
1176 context (e.g. transport protection).

1177 It is RECOMMENDED that a message independent referencing mechanism be used to identify the token,
1178 if the token being referenced supports such mechanism.

1179 The following exemplar defines the `CreateSequence` syntax when extended to include a
1180 `wsse:SecurityTokenReference`:

```
1181 <wsrm:CreateSequence ...>  
1182   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>  
1183   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1184   <wsrm:Offer ...>  
1185     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
1186     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>  
1187     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1188     <wsrm:IncompleteSequenceBehavior>  
1189       wsrml:IncompleteSequenceBehaviorType  
1190     </wsrm:IncompleteSequenceBehavior> ?  
1191     ...  
1192   </wsrm:Offer> ?  
1193   ...  
1194   <wsse:SecurityTokenReference>  
1195     ...  
1196   </wsse:SecurityTokenReference> ?  
1197   ...  
1198 </wsrm:CreateSequence>
```

1199 The following describes the content model of the additional `CreateSequence` elements.

1200 `/wsrm:CreateSequence/wsse:SecurityTokenReference`

1201 This element uses the extensibility mechanism defined for the `CreateSequence` element (defined in
1202 section 3.4) to communicate an explicit reference to the security token, using a
1203 `wsse:SecurityTokenReference` as documented in WS-Security, that the RM Source and Destination
1204 MUST use to authorize messages for the created (and, if present, the offered) Sequence(s). All
1205 subsequent messages related to the created (and, if present, the offered) Sequence(s) MUST
1206 demonstrate proof-of-possession of the secret associated with the token (e.g., by using or deriving from a
1207 private or secret key).

1208 When a RM Source transmits a `CreateSequence` that has been extended to include a
1209 `wsse:SecurityTokenReference` it SHOULD ensure that the RM Destination both understands and
1210 will conform to the requirements listed above. In order to achieve this, the RM Source SHOULD include

1211 the `UsesSequenceSTR` element as a SOAP header block within the `CreateSequence` message. This
1212 element MUST include a `soap:mustUnderstand` attribute with a value of 'true'. Thus the RM Source
1213 can be assured that a RM Destination that responds with a `CreateSequenceResponse` understands
1214 and conforms with the requirements listed above. Note that an RM Destination understanding this header
1215 does not mean that it has processed and understood any WS-Security headers, the fault behavior defined
1216 in WS-Security still applies.

1217 The following exemplar defines the `UsesSequenceSTR` syntax:

```
1218 <wsm:UsesSequenceSTR ... />
```

1219 The following describes the content model of the `UsesSequenceSTR` header block.

1220 `/wsm:UsesSequenceSTR`

1221 This element SHOULD be included as a SOAP header block in `CreateSequence` messages that use the
1222 extensibility mechanism described above in this section. The `soap:mustUnderstand` attribute value
1223 MUST be 'true'. The receiving RM Destination MUST understand and correctly implement the extension
1224 described above or else generate a `soap:MustUnderstand` fault, thus aborting the requested
1225 Sequence creation.

1226 The following is an example of a `CreateSequence` message using the
1227 `wsse:SecurityTokenReference` extension and the `UsesSequenceSTR` header block:

```
1228 <soap:Envelope ...>  
1229   <soap:Header>  
1230     ...  
1231     <wsm:UsesSequenceSTR soap:mustUnderstand='true' />  
1232     ...  
1233   </soap:Header>  
1234   <soap:Body>  
1235     <wsm:CreateSequence>  
1236       <wsm:AcksTo>  
1237         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>  
1238       </wsm:AcksTo>  
1239       <wsse:SecurityTokenReference>  
1240         ...  
1241       </wsse:SecurityTokenReference>  
1242     </wsm:CreateSequence>  
1243   </soap:Body>  
1244 </soap:Envelope>
```

1245 6.2 Securing Sequences Using SSL/TLS

1246 One mechanism for protecting a Sequence is to bind the Sequence to the underlying SSL/TLS session(s).
1247 The RM Source indicates to the RM Destination that a Sequence is to be bound to the underlying
1248 SSL/TLS session(s) via the `UsesSequenceSSL` header block. If the RM Source wishes to bind a
1249 Sequence to the underlying SSL/TLS sessions(s) it MUST include the `UsesSequenceSSL` element as a
1250 SOAP header block within the `CreateSequence` message.

1251 The following exemplar defines the `UsesSequenceSSL` syntax:

```
1252 <wsm:UsesSequenceSSL soap:mustUnderstand="true" ... />
```

1253 The following describes the content model of the `UsesSequenceSSL` header block.

1254 `/wsm:UsesSequenceSSL`

1255 The RM Source MAY include this element as a SOAP header block of a `CreateSequence` message to
1256 indicate to the RM Destination that the resulting Sequence is to be bound to the SSL/TLS session that was

1257 used to carry the `CreateSequence` message. If included, the RM Source MUST mark this header with a
1258 `soap:mustUnderstand` attribute with a value of 'true'. The receiving RM Destination MUST understand
1259 and correctly implement the functionality described in Section 5.2.1 or else generate a
1260 `soap:MustUnderstand` fault, thus aborting the requested Sequence creation.

1261 Note that the use inclusion of the above header by the RM Source implies that all Sequence-related
1262 information (Sequence Lifecycle or Acknowledgment messages or Sequence-related faults) flowing from
1263 the RM Destination to the RM Source will be bound to the SSL/TLS session that is used to carry the
1264 `CreateSequenceResponse` message.

1265 **7 References**

1266 **7.1 Normative**

1267 **[KEYWORDS]**

1268 S. Bradner, "[Key words for use in RFCs to Indicate Requirement Levels](#)," RFC 2119, Harvard University,
1269 March 1997

1270 <http://www.ietf.org/rfc/rfc2119.txt>

1271 **[WS-RM Policy]**

1272 OASIS WS-RX Technical Committee Draft, "[Web Services ReliableMessaging Policy Assertion\(WS-RM
1273 Policy\)](#)" October 2006

1274 <http://docs.oasis-open.org/ws-rx/wsrmp/200608/wsrmp-1.1-spec-wd-11.pdf>

1275 **[SOAP 1.1]**

1276 W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.

1277 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

1278 **[SOAP 1.2]**

1279 W3C Recommendation, "[SOAP Version 1.2 Part 1: Messaging Framework](#)" June 2003.

1280 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

1281 **[URI]**

1282 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#)," RFC 3986,
1283 MIT/LCS, U.C. Irvine, Xerox Corporation, January 2005.

1284 <http://ietf.org/rfc/rfc3986>

1285 **[UUID]**

1286 P. Leach, M. Mealling, R. Salz, "[A Universally Unique IDentifier \(UUID\) URN Namespace](#)," RFC 4122,
1287 Microsoft, Refactored Networks - LLC, DataPower Technology Inc, July 2005

1288 <http://www.ietf.org/rfc/rfc4122.txt>

1289 **[XML]**

1290 W3C Recommendation, "[Extensible Markup Language \(XML\) 1.0 \(Fourth Edition\)](#)", September 2006.

1291 <http://www.w3.org/TR/REC-xml/>

1292 **[XML-ns]**

1293 W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.

1294 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>

1295 **[XML-Schema Part1]**

1296 W3C Recommendation, "[XML Schema Part 1: Structures](#)," October 2004.

1297 <http://www.w3.org/TR/xmlschema-1/>

1298 **[XML-Schema Part2]**

1299 W3C Recommendation, "XML Schema Part 2: Datatypes," October 2004.

1300 <http://www.w3.org/TR/xmlschema-2/>

1301 **[XPath 1.0]**

1302 W3C Recommendation, "XML Path Language (XPath) Version 1.0," 16 November 1999.

1303 <http://www.w3.org/TR/xpath>

1304 **[WSDL 1.1]**

1305 W3C Note, "Web Services Description Language (WSDL 1.1)," 15 March 2001.

1306 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

1307 **[WS-Addressing]**

1308 W3C Recommendation, "Web Services Addressing 1.0 - Core", May 2006.

1309 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>

1310 W3C Recommendation, "Web Services Addressing 1.0 – SOAP Binding", May 2006.

1311 <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>

1312 **7.2 Non-Normative**

1313 **[BSP 1.0]**

1314 WS-I Working Group Draft. "Basic Security Profile Version 1.0," August 2006

1315 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>

1316 **[RDDL 2.0]**

1317 Jonathan Borden, Tim Bray, eds. "Resource Directory Description Language (RDDL) 2.0," January 2004

1318 <http://www.openhealth.org/RDDL/20040118/rddl-20040118.html>

1319 **[RFC 2617]**

1320 J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Loutonen, L. Stewart, "HTTP
1321 Authentication: Basic and Digest Access Authentication," June 1999.

1322 <http://www.ietf.org/rfc/rfc2617.txt>

1323 **[RFC 4346]**

1324 T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1," April 2006.

1325 <http://www.ietf.org/rfc/rfc4346.txt>

1326 **[WS-Policy]**

1327 W3C Member Submission, "Web Services Policy Framework (WS-Policy)," April 2006.

1328 <http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/>

1329 **[WS-PolicyAttachment]**

1330 W3C Member Submission, "Web Services Policy Attachment (WS-PolicyAttachment)," April 2006.

1331 [http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-
1332 20060425/](http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-20060425/)

1333 **[WS-Security]**

1334 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:
1335 SOAP Message Security 1.0 (WS-Security 2004)", OASIS Standard 200401, March 2004.

1336 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

1337 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:
1338 SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard 200602, February 2006.

1339 <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

1340 **[RTTM]**

1341 V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, May
1342 1992.

1343 <http://www.rfc-editor.org/rfc/rfc1323.txt>

1344 **[SecurityPolicy]**

1345 G. Della-Libra, et. al. "Web Services Security Policy Language (WS-SecurityPolicy)", July 2005

1346 <http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf>

1347 **[SecureConversation]**

1348 S. Anderson, et al, "Web Services Secure Conversation Language (WS-SecureConversation)," February
1349 2005.

1350 <http://schemas.xmlsoap.org/ws/2004/04/sc/>

1351 **[Trust]**

1352 S. Anderson, et al, "Web Services Trust Language (WS-Trust)," February 2005.

1353 <http://schemas.xmlsoap.org/ws/2005/02/trust>

1354 Appendix A. Schema

1355 The normative schema that is defined for WS-ReliableMessaging using [XML-Schema Part1] and [XML-
1356 Schema Part2] is located at:

1357 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-200608.xsd>

1358 The following copy is provided for reference.

```
1359 <?xml version="1.0" encoding="UTF-8"?>
1360 <!--
1361 OASIS takes no position regarding the validity or scope of any intellectual
1362 property or other rights that might be claimed to pertain to the
1363 implementation or use of the technology described in this document or the
1364 extent to which any license under such rights might or might not be available;
1365 neither does it represent that it has made any effort to identify any such
1366 rights. Information on OASIS's procedures with respect to rights in OASIS
1367 specifications can be found at the OASIS website. Copies of claims of rights
1368 made available for publication and any assurances of licenses to be made
1369 available, or the result of an attempt made to obtain a general license or
1370 permission for the use of such proprietary rights by implementors or users of
1371 this specification, can be obtained from the OASIS Executive Director.
1372 OASIS invites any interested party to bring to its attention any copyrights,
1373 patents or patent applications, or other proprietary rights which may cover
1374 technology that may be required to implement this specification. Please
1375 address the information to the OASIS Executive Director.
1376 Copyright © OASIS Open 2002-2006. All Rights Reserved.
1377 This document and translations of it may be copied and furnished to others,
1378 and derivative works that comment on or otherwise explain it or assist in its
1379 implementation may be prepared, copied, published and distributed, in whole or
1380 in part, without restriction of any kind, provided that the above copyright
1381 notice and this paragraph are included on all such copies and derivative
1382 works. However, this document itself does not be modified in any way, such as
1383 by removing the copyright notice or references to OASIS, except as needed for
1384 the purpose of developing OASIS specifications, in which case the procedures
1385 for copyrights defined in the OASIS Intellectual Property Rights document must
1386 be followed, or as required to translate it into languages other than English.
1387 The limited permissions granted above are perpetual and will not be revoked by
1388 OASIS or its successors or assigns.
1389 This document and the information contained herein is provided on an "AS IS"
1390 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1391 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1392 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1393 FOR A PARTICULAR PURPOSE.
1394 -->
1395 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
1396 xmlns:wsa="http://www.w3.org/2005/08/addressing"
1397 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1398 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1399 elementFormDefault="qualified" attributeFormDefault="unqualified">
1400   <xs:import namespace="http://www.w3.org/2005/08/addressing"
1401   schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd"/>
1402   <!-- Protocol Elements -->
1403   <xs:complexType name="SequenceType">
1404     <xs:sequence>
1405       <xs:element ref="wsrm:Identifier"/>
1406       <xs:element name="MessageNumber" type="wsrm:MessageNumberType"/>
1407       <xs:any namespace="##other" processContents="lax" minOccurs="0"
1408 maxOccurs="unbounded"/>
1409     </xs:sequence>
```

```

1410     <xs:anyAttribute namespace="##other" processContents="lax"/>
1411 </xs:complexType>
1412 <xs:element name="Sequence" type="wsrm:SequenceType"/>
1413 <xs:element name="SequenceAcknowledgement">
1414   <xs:complexType>
1415     <xs:sequence>
1416       <xs:element ref="wsrm:Identifier"/>
1417       <xs:choice>
1418         <xs:sequence>
1419           <xs:choice>
1420             <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
1421               <xs:complexType>
1422                 <xs:sequence/>
1423                 <xs:attribute name="Upper" type="xs:unsignedLong"
1424 use="required"/>
1425                 <xs:attribute name="Lower" type="xs:unsignedLong"
1426 use="required"/>
1427               <xs:anyAttribute namespace="##other" processContents="lax"/>
1428             </xs:complexType>
1429           </xs:element>
1430           <xs:element name="None">
1431             <xs:complexType>
1432               <xs:sequence/>
1433             </xs:complexType>
1434           </xs:element>
1435         </xs:choice>
1436         <xs:element name="Final" minOccurs="0">
1437           <xs:complexType>
1438             <xs:sequence/>
1439           </xs:complexType>
1440         </xs:element>
1441       </xs:sequence>
1442       <xs:element name="Nack" type="xs:unsignedLong"
1443 maxOccurs="unbounded"/>
1444     </xs:choice>
1445     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1446 maxOccurs="unbounded"/>
1447   </xs:sequence>
1448   <xs:anyAttribute namespace="##other" processContents="lax"/>
1449 </xs:complexType>
1450 </xs:element>
1451 <xs:complexType name="AckRequestedType">
1452   <xs:sequence>
1453     <xs:element ref="wsrm:Identifier"/>
1454     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1455 maxOccurs="unbounded"/>
1456   </xs:sequence>
1457   <xs:anyAttribute namespace="##other" processContents="lax"/>
1458 </xs:complexType>
1459 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1460 <xs:element name="Identifier">
1461   <xs:complexType>
1462     <xs:annotation>
1463       <xs:documentation>
1464         This type is for elements whose [children] is an anyURI and can have
1465 arbitrary attributes.
1466       </xs:documentation>
1467     </xs:annotation>
1468     <xs:simpleContent>
1469       <xs:extension base="xs:anyURI">
1470         <xs:anyAttribute namespace="##other" processContents="lax"/>
1471       </xs:extension>
1472     </xs:simpleContent>

```

```

1473     </xs:complexType>
1474 </xs:element>
1475 <xs:element name="Address">
1476   <xs:complexType>
1477     <xs:simpleContent>
1478       <xs:extension base="xs:anyURI">
1479         <xs:anyAttribute namespace="##other" processContents="lax"/>
1480       </xs:extension>
1481     </xs:simpleContent>
1482   </xs:complexType>
1483 </xs:element>
1484 <xs:simpleType name="MessageNumberType">
1485   <xs:restriction base="xs:unsignedLong">
1486     <xs:minInclusive value="1"/>
1487     <xs:maxInclusive value="9223372036854775807"/>
1488   </xs:restriction>
1489 </xs:simpleType>
1490 <!-- Fault Container and Codes -->
1491 <xs:simpleType name="FaultCodes">
1492   <xs:restriction base="xs:QName">
1493     <xs:enumeration value="wsrm:SequenceTerminated"/>
1494     <xs:enumeration value="wsrm:UnknownSequence"/>
1495     <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1496     <xs:enumeration value="wsrm:MessageNumberRollover"/>
1497     <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1498     <xs:enumeration value="wsrm:SequenceClosed"/>
1499     <xs:enumeration value="wsrm:WSRMRequired"/>
1500     <xs:enumeration value="wsrm:UnsupportedSelection"/>
1501   </xs:restriction>
1502 </xs:simpleType>
1503 <xs:complexType name="SequenceFaultType">
1504   <xs:sequence>
1505     <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1506     <xs:element name="Detail" type="wsrm:DetailType" minOccurs="0"/>
1507     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1508 maxOccurs="unbounded"/>
1509   </xs:sequence>
1510   <xs:anyAttribute namespace="##other" processContents="lax"/>
1511 </xs:complexType>
1512 <xs:complexType name="DetailType">
1513   <xs:sequence>
1514     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1515 maxOccurs="unbounded"/>
1516   </xs:sequence>
1517   <xs:anyAttribute namespace="##other" processContents="lax"/>
1518 </xs:complexType>
1519 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1520 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1521 <xs:element name="CreateSequenceResponse"
1522 type="wsrm:CreateSequenceResponseType"/>
1523 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1524 <xs:element name="CloseSequenceResponse"
1525 type="wsrm:CloseSequenceResponseType"/>
1526 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1527 <xs:element name="TerminateSequenceResponse"
1528 type="wsrm:TerminateSequenceResponseType"/>
1529 <xs:complexType name="CreateSequenceType">
1530   <xs:sequence>
1531     <xs:element ref="wsrm:AcksTo"/>
1532     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1533     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1534     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1535 maxOccurs="unbounded"/>

```

```

1536     <xs:annotation>
1537         <xs:documentation>
1538             It is the authors intent that this extensibility be used to
1539 transfer a Security Token Reference as defined in WS-Security.
1540         </xs:documentation>
1541     </xs:annotation>
1542 </xs:any>
1543 </xs:sequence>
1544 <xs:anyAttribute namespace="##other" processContents="lax"/>
1545 </xs:complexType>
1546 <xs:complexType name="CreateSequenceResponseType">
1547     <xs:sequence>
1548         <xs:element ref="wsrm:Identifier"/>
1549         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1550         <xs:element name="IncompleteSequenceBehavior"
1551 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1552         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1553         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1554 maxOccurs="unbounded"/>
1555     </xs:sequence>
1556     <xs:anyAttribute namespace="##other" processContents="lax"/>
1557 </xs:complexType>
1558 <xs:complexType name="CloseSequenceType">
1559     <xs:sequence>
1560         <xs:element ref="wsrm:Identifier"/>
1561         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1562 maxOccurs="unbounded"/>
1563     </xs:sequence>
1564     <xs:anyAttribute namespace="##other" processContents="lax"/>
1565 </xs:complexType>
1566 <xs:complexType name="CloseSequenceResponseType">
1567     <xs:sequence>
1568         <xs:element ref="wsrm:Identifier"/>
1569         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1570 maxOccurs="unbounded"/>
1571     </xs:sequence>
1572     <xs:anyAttribute namespace="##other" processContents="lax"/>
1573 </xs:complexType>
1574 <xs:complexType name="TerminateSequenceType">
1575     <xs:sequence>
1576         <xs:element ref="wsrm:Identifier"/>
1577         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1578 maxOccurs="unbounded"/>
1579     </xs:sequence>
1580     <xs:anyAttribute namespace="##other" processContents="lax"/>
1581 </xs:complexType>
1582 <xs:complexType name="TerminateSequenceResponseType">
1583     <xs:sequence>
1584         <xs:element ref="wsrm:Identifier"/>
1585         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1586 maxOccurs="unbounded"/>
1587     </xs:sequence>
1588     <xs:anyAttribute namespace="##other" processContents="lax"/>
1589 </xs:complexType>
1590 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1591 <xs:complexType name="OfferType">
1592     <xs:sequence>
1593         <xs:element ref="wsrm:Identifier"/>
1594         <xs:element name="Endpoint" type="wsa:EndpointReferenceType"/>
1595         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1596         <xs:element name="IncompleteSequenceBehavior"
1597 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1598         <xs:any namespace="##other" processContents="lax" minOccurs="0"

```

```

1599 maxOccurs="unbounded"/>
1600     </xs:sequence>
1601     <xs:anyAttribute namespace="##other" processContents="lax"/>
1602 </xs:complexType>
1603 <xs:complexType name="AcceptType">
1604     <xs:sequence>
1605         <xs:element ref="wsrm:AcksTo"/>
1606         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1607 maxOccurs="unbounded"/>
1608     </xs:sequence>
1609     <xs:anyAttribute namespace="##other" processContents="lax"/>
1610 </xs:complexType>
1611 <xs:element name="Expires">
1612     <xs:complexType>
1613         <xs:simpleContent>
1614             <xs:extension base="xs:duration">
1615                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1616             </xs:extension>
1617         </xs:simpleContent>
1618     </xs:complexType>
1619 </xs:element>
1620 <xs:simpleType name="IncompleteSequenceBehaviorType">
1621     <xs:restriction base="xs:string">
1622         <xs:enumeration value="DiscardEntireSequence"/>
1623         <xs:enumeration value="DiscardFollowingFirstGap"/>
1624         <xs:enumeration value="NoDiscard"/>
1625     </xs:restriction>
1626 </xs:simpleType>
1627 <xs:element name="UsesSequenceSTR">
1628     <xs:complexType>
1629         <xs:sequence/>
1630         <xs:anyAttribute namespace="##other" processContents="lax"/>
1631     </xs:complexType>
1632 </xs:element>
1633 <xs:element name="UsesSequenceSSL">
1634     <xs:complexType>
1635         <xs:sequence/>
1636         <xs:anyAttribute namespace="##other" processContents="lax"/>
1637     </xs:complexType>
1638 </xs:element>
1639 <xs:element name="UnsupportedElement">
1640     <xs:simpleType>
1641         <xs:restriction base="xs:QName"/>
1642     </xs:simpleType>
1643 </xs:element>
1644 </xs:schema>

```

1645 Appendix B. WSDL

1646 This WSDL describes the WS-RM protocol from the point of view of an RM Destination. In the case where
1647 an endpoint acts both as an RM Destination and an RM Source, note that additional messages may be
1648 present in exchanges with that endpoint.

1649 Also note that this WSDL is intended to describe the internal structure of the WS-RM protocol, and will not
1650 generally appear in a description of a WS-RM-capable Web service. See WS-RM Policy [WS-RM Policy]
1651 for a higher-level mechanism to indicate that WS-RM is engaged.

1652 The normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1653 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsd/wsrn-1.1-wsd-200608.wsd>

1654 The following non-normative copy is provided for reference.

```
1655 <?xml version="1.0" encoding="utf-8"?>
1656 <!--
1657 OASIS takes no position regarding the validity or scope of any intellectual
1658 property or other rights that might be claimed to pertain to the
1659 implementation or use of the technology described in this document or the
1660 extent to which any license under such rights might or might not be available;
1661 neither does it represent that it has made any effort to identify any such
1662 rights. Information on OASIS's procedures with respect to rights in OASIS
1663 specifications can be found at the OASIS website. Copies of claims of rights
1664 made available for publication and any assurances of licenses to be made
1665 available, or the result of an attempt made to obtain a general license or
1666 permission for the use of such proprietary rights by implementors or users of
1667 this specification, can be obtained from the OASIS Executive Director.
1668 OASIS invites any interested party to bring to its attention any copyrights,
1669 patents or patent applications, or other proprietary rights which may cover
1670 technology that may be required to implement this specification. Please
1671 address the information to the OASIS Executive Director.
1672 Copyright (c) OASIS Open 2002-2006. All Rights Reserved.
1673 This document and translations of it may be copied and furnished to others,
1674 and derivative works that comment on or otherwise explain it or assist in its
1675 implementation may be prepared, copied, published and distributed, in whole or
1676 in part, without restriction of any kind, provided that the above copyright
1677 notice and this paragraph are included on all such copies and derivative
1678 works. However, this document itself does not be modified in any way, such as
1679 by removing the copyright notice or references to OASIS, except as needed for
1680 the purpose of developing OASIS specifications, in which case the procedures
1681 for copyrights defined in the OASIS Intellectual Property Rights document must
1682 be followed, or as required to translate it into languages other than English.
1683 The limited permissions granted above are perpetual and will not be revoked by
1684 OASIS or its successors or assigns.
1685 This document and the information contained herein is provided on an "AS IS"
1686 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1687 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1688 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1689 FOR A PARTICULAR PURPOSE.
1690 -->
1691 <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1692 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1693 xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:rm="http://docs.oasis-
1694 open.org/ws-rx/wsrn/200608" xmlns:tns="http://docs.oasis-open.org/ws-
1695 rx/wsrn/200608/wsd" targetNamespace="http://docs.oasis-open.org/ws-
1696 rx/wsrn/200608/wsd">
1697 <wsdl:types>
```

```

1698     <xs:schema>
1699         <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsr/200608"
1700 schemaLocation="http://docs.oasis-open.org/ws-rx/wsr/200608/wsr-1.1-schema-
1701 200608.xsd"/>
1702     </xs:schema>
1703 </wsdl:types>

1704     <wsdl:message name="CreateSequence">
1705         <wsdl:part name="create" element="rm:CreateSequence"/>
1706     </wsdl:message>
1707     <wsdl:message name="CreateSequenceResponse">
1708         <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1709     </wsdl:message>
1710     <wsdl:message name="CloseSequence">
1711         <wsdl:part name="close" element="rm:CloseSequence"/>
1712     </wsdl:message>
1713     <wsdl:message name="CloseSequenceResponse">
1714         <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1715     </wsdl:message>
1716     <wsdl:message name="TerminateSequence">
1717         <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1718     </wsdl:message>
1719     <wsdl:message name="TerminateSequenceResponse">
1720         <wsdl:part name="terminateResponse"
1721 element="rm:TerminateSequenceResponse"/>
1722     </wsdl:message>

1723     <wsdl:portType name="SequenceAbstractPortType">
1724         <wsdl:operation name="CreateSequence">
1725             <wsdl:input message="tns:CreateSequence" wsaw:Action="http://docs.oasis-
1726 open.org/ws-rx/wsr/200608/CreateSequence"/>
1727             <wsdl:output message="tns:CreateSequenceResponse"
1728 wsaw:Action="http://docs.oasis-open.org/ws-
1729 rx/wsr/200608/CreateSequenceResponse"/>
1730         </wsdl:operation>
1731         <wsdl:operation name="CloseSequence">
1732             <wsdl:input message="tns:CloseSequence" wsaw:Action="http://docs.oasis-
1733 open.org/ws-rx/wsr/200608/CloseSequence"/>
1734             <wsdl:output message="tns:CloseSequenceResponse"
1735 wsaw:Action="http://docs.oasis-open.org/ws-
1736 rx/wsr/200608/CloseSequenceResponse"/>
1737         </wsdl:operation>
1738         <wsdl:operation name="TerminateSequence">
1739             <wsdl:input message="tns:TerminateSequence"
1740 wsaw:Action="http://docs.oasis-open.org/ws-rx/wsr/200608/TerminateSequence"/>
1741             <wsdl:output message="tns:TerminateSequenceResponse"
1742 wsaw:Action="http://docs.oasis-open.org/ws-
1743 rx/wsr/200608/TerminateSequenceResponse"/>
1744         </wsdl:operation>
1745     </wsdl:portType>

1746 </wsdl:definitions>

```

1747 Appendix C. Message Examples

1748 Appendix C.1 Create Sequence

1749 Create Sequence

```
1750 <?xml version="1.0" encoding="UTF-8"?>
1751 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1752 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1753 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1754   <S:Header>
1755     <wsa:MessageID>
1756       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1757     </wsa:MessageID>
1758     <wsa:To>http://example.com/serviceB/123</wsa:To>
1759     <wsa:Action>http://docs.oasis-open.org/ws-
1760 rx/wsmr/200608/CreateSequence</wsa:Action>
1761     <wsa:ReplyTo>
1762       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1763     </wsa:ReplyTo>
1764   </S:Header>
1765   <S:Body>
1766     <wsmr:CreateSequence>
1767       <wsmr:AcksTo>
1768         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1769       </wsmr:AcksTo>
1770     </wsmr:CreateSequence>
1771   </S:Body>
1772 </S:Envelope>
```

1773 Create Sequence Response

```
1774 <?xml version="1.0" encoding="UTF-8"?>
1775 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1776 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1777 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1778   <S:Header>
1779     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1780     <wsa:RelatesTo>
1781       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1782     </wsa:RelatesTo>
1783     <wsa:Action>
1784       http://docs.oasis-open.org/ws-rx/wsmr/200608/CreateSequenceResponse
1785     </wsa:Action>
1786   </S:Header>
1787   <S:Body>
1788     <wsmr:CreateSequenceResponse>
1789       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1790     </wsmr:CreateSequenceResponse>
1791   </S:Body>
1792 </S:Envelope>
```

1793 Appendix C.2 Initial Transmission

1794 The following example WS-ReliableMessaging headers illustrate the message exchange in the above
1795 figure. The three messages have the following headers; the third message is identified as the last
1796 message in the Sequence:

1797 **Message 1**

```
1798 <?xml version="1.0" encoding="UTF-8"?>
1799 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1800 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1801 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1802   <S:Header>
1803     <wsa:MessageID>
1804       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1805     </wsa:MessageID>
1806     <wsa:To>http://example.com/serviceB/123</wsa:To>
1807     <wsa:From>
1808       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1809     </wsa:From>
1810     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1811     <wsmr:Sequence>
1812       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1813       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1814     </wsmr:Sequence>
1815   </S:Header>
1816   <S:Body>
1817     <!-- Some Application Data -->
1818   </S:Body>
1819 </S:Envelope>
```

1820 **Message 2**

```
1821 <?xml version="1.0" encoding="UTF-8"?>
1822 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1823 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1824 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1825   <S:Header>
1826     <wsa:MessageID>
1827       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1828     </wsa:MessageID>
1829     <wsa:To>http://example.com/serviceB/123</wsa:To>
1830     <wsa:From>
1831       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1832     </wsa:From>
1833     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1834     <wsmr:Sequence>
1835       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1836       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1837     </wsmr:Sequence>
1838   </S:Header>
1839   <S:Body>
1840     <!-- Some Application Data -->
1841   </S:Body>
1842 </S:Envelope>
```

1843 **Message 3**

```
1844 <?xml version="1.0" encoding="UTF-8"?>
1845 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1846 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1847 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1848   <S:Header>
1849     <wsa:MessageID>
1850       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1851     </wsa:MessageID>
1852     <wsa:To>http://example.com/serviceB/123</wsa:To>
1853     <wsa:From>
1854       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1855 </wsa:From>
1856 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1857 <wsrm:Sequence>
1858 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1859 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1860 </wsrm:Sequence>
1861 <wsrm:AckRequested>
1862 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1863 </wsrm:AckRequested>
1864 </S:Header>
1865 <S:Body>
1866 <!-- Some Application Data -->
1867 </S:Body>
1868 </S:Envelope>

```

1869 **Appendix C.3 First Acknowledgement**

1870 Message number 2 has not been accepted by the RM Destination due to some transmission error so it
1871 responds with an Acknowledgement for messages 1 and 3:

```

1872 <?xml version="1.0" encoding="UTF-8"?>
1873 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1874 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsr/200608"
1875 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1876 <S:Header>
1877 <wsa:MessageID>
1878 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1879 </wsa:MessageID>
1880 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1881 <wsa:From>
1882 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1883 </wsa:From>
1884 <wsa:Action>
1885 http://docs.oasis-open.org/ws-rx/wsr/200608/SequenceAcknowledgement
1886 </wsa:Action>
1887 <wsrm:SequenceAcknowledgement>
1888 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1889 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1890 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1891 </wsrm:SequenceAcknowledgement>
1892 </S:Header>
1893 <S:Body/>
1894 </S:Envelope>

```

1895 **Appendix C.4 Retransmission**

1896 The RM Sourcediscovers that message number 2 was not accepted so it resends the message and
1897 requests an Acknowledgement:

```

1898 <?xml version="1.0" encoding="UTF-8"?>
1899 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1900 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsr/200608"
1901 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1902 <S:Header>
1903 <wsa:MessageID>
1904 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1905 </wsa:MessageID>
1906 <wsa:To>http://example.com/serviceB/123</wsa:To>
1907 <wsa:From>
1908 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1909 </wsa:From>

```

```

1910 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1911 <wsrm:Sequence>
1912 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1913 <wsrm:MessageNumber>2</wsrm:MessageNumber>
1914 </wsrm:Sequence>
1915 <wsrm:AckRequested>
1916 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1917 </wsrm:AckRequested>
1918 </S:Header>
1919 <S:Body>
1920 <!-- Some Application Data -->
1921 </S:Body>
1922 </S:Envelope>

```

1923 Appendix C.5 Termination

1924 The RM Destination now responds with an Acknowledgement for the complete Sequence which can then
1925 be terminated:

```

1926 <?xml version="1.0" encoding="UTF-8"?>
1927 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1928 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1929 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1930 <S:Header>
1931 <wsa:MessageID>
1932 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1933 </wsa:MessageID>
1934 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1935 <wsa:From>
1936 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1937 </wsa:From>
1938 <wsa:Action>
1939 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1940 </wsa:Action>
1941 <wsrm:SequenceAcknowledgement>
1942 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1943 <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1944 </wsrm:SequenceAcknowledgement>
1945 </S:Header>
1946 <S:Body/>
1947 </S:Envelope>

```

1948 Terminate Sequence

```

1949 <?xml version="1.0" encoding="UTF-8"?>
1950 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1951 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1952 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1953 <S:Header>
1954 <wsa:MessageID>
1955 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1956 </wsa:MessageID>
1957 <wsa:To>http://example.com/serviceB/123</wsa:To>
1958 <wsa:Action>
1959 http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence
1960 </wsa:Action>
1961 <wsa:From>
1962 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1963 </wsa:From>
1964 </S:Header>
1965 <S:Body>
1966 <wsrm:TerminateSequence>

```

```
1967     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1968     </wsrm:TerminateSequence>
1969     </S:Body>
1970 </S:Envelope>
```

1971 Terminate Sequence Response

```
1972 <?xml version="1.0" encoding="UTF-8"?>
1973 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1974 xmlns:wsm="http://docs.oasis-open.org/ws-rx/wsm/200608"
1975 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1976   <S:Header>
1977     <wsa:MessageID>
1978       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
1979     </wsa:MessageID>
1980     <wsa:To>http://example.com/serviceA/789</wsa:To>
1981     <wsa:Action>
1982       http://docs.oasis-open.org/ws-rx/wsm/200608/TerminateSequenceResponse
1983     </wsa:Action>
1984     <wsa:RelatesTo>
1985       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1986     </wsa:RelatesTo>
1987     <wsa:From>
1988       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1989     </wsa:From>
1990   </S:Header>
1991   <S:Body>
1992     <wsrm:TerminateSequenceResponse>
1993       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1994     </wsrm:TerminateSequenceResponse>
1995   </S:Body>
1996 </S:Envelope>
```

1997 Appendix D. State Tables

1998 This appendix specifies the non-normative state transition tables for RM Source and RM Destination.

1999 The state tables describe the lifetime of a sequence in both the RM Source and the RM Destination

2000 Legend:

2001 The first column of these tables contains the motivating event and has the following format:

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

2002 Where:

- 2003 ● Event Name: indicates the name of the event. Event Names surrounded by "<>" are optional as
2004 described by the specification.
- 2005 ● [source]: indicates the source of the event; one of:
 - 2006 ● [msg] a Received message
 - 2007 ● [int]: an internal event such as the firing of a timer
 - 2008 ● [app]: the application
 - 2009 ● [unspec]: the source is unspecified

2010 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}

2011 Where:

- 2012 ● action to take: indicates that the state machine performs the following action. Actions surrounded
2013 by "<>" are optional as described by the specification. "Xmit" is used as a short form for the word
2014 "Transmit"
 - 2015 ● [next state]: indicates the state to which the state machine will advance upon the performance of
2016 the action. For ease of reading the next state "same" indicates that the state does not change.
 - 2017 ● {ref} is a reference to the document section describing the behavior in this cell
- 2018 "N/A" in a cell indicates a state / event combination self-inconsistent with the state machine; should these
2019 conditions occur, it would indicate an implementation error. A blank cell indicates that the behavior is not
2020 described in this specification and does not indicate normal protocol operation. Implementations MAY
2021 generate a Sequence Terminated fault (see section 4.2) in these circumstances. Robust implementations
2022 MUST be able to operate in a stable manner despite the occurrence of unspecified event / state
2023 combinations.

2024 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] {3.4}	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] {3.4}	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]
<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}
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] {3.4}	Generate Unknown Sequence Fault	Generate Unknown Sequence Fault	No action [Closed] {4.7}	No action [Closed] {4.7}	No action [Same]	No action [Same]

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
{4.7}	[Same] {4.3}	[Same] {4.3}				
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}
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 Acknowledgment [msg] {4.4}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Invalid Acknowledgment Fault [Same] {4.4}	Generate Invalid Acknowledgment Fault [Same] {4.4}	Generate Invalid Acknowledgment Fault [Same] {4.4}	Generate Invalid Acknowledgment Fault [Same] {4.4}

2024 Table 2 RM Destination Sequence State Transition Table

Events	Sequence States		
	None	Created	Closed
CreateSequence (successful) [msg/int] {3.4}	Xmit Create Sequence Response [Created] {3.4}	N/A	N/A
CreateSequence (unsuccessful) [msg/int] {3.4}	Generate Create Sequence Refused Fault [None] {3.4}	N/A	N/A
Message (with message number within range) [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Accept Message; <Xmit SeqAck> [Same]	Generate Sequence Closed Fault (with SeqAck+Final) [Same] {3.5}
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}	Generate Sequence Closed Fault (with SeqAck+Final) [Same] {3.5}
<AckRequested> [msg] {3.8}	Generate Unknown Seq Fault [Same] {4.3}	Xmit SeqAck [Same] {3.8}	Xmit SeqAck+Final [Same] {3.9}

Events	Sequence States		
	None	Created	Closed
CloseSequence [msg] {3.5}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit CloseSequence Response with SeqAck+Final [Closed] {3.5}	Xmit CloseSequence Response with SeqAck+Final [Closed] {3.5}
<CloseSequence autonomously> [int]	N/A	No Action [Closed]	N/A
TerminateSequence [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Terminate Sequence Response [None] {3.6}	Xmit Terminate Sequence Response [None] {3.6}
UnknownSequence Fault [msg] {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}
Invalid Acknowledgement Fault [msg] {4.4}	N/A		
Expires exceeded [int]	N/A	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}
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}

2025 **Appendix E. Acknowledgments**

2026 This document is based on initial contribution to OASIS WS-RX Technical Committee by the following
2027 authors:

2028 Ruslan Bilorusets(BEA), Don Box(Microsoft), Luis Felipe Cabrera(Microsoft), Doug Davis(IBM),
2029 Donald Ferguson(IBM), Christopher Ferris-Editor(BM), Tom Freund(IBM), Mary Ann Hondo(IBM),
2030 John Ibbotson(IBM), Lei Jin(BEA), Chris Kaler(Microsoft), David Langworthy-Editor(Microsoft),
2031 Amelia Lewis(TIBCO Software), Rodney Limprecht(Microsoft), Steve Lucco(Microsoft), Don
2032 Mullen(TIBCO Software), Anthony Nadalin(IBM), Mark Nottingham(BEA), David Orchard(BEA),
2033 Jamie Roots(IBM), Shivajee Samdarshi(TIBCO Software), John Shewchuk(Microsoft), Tony
2034 Storey(IBM).

2035 The following individuals have provided invaluable input into the initial contribution:

2036 Keith Ballinger(Microsoft), Stefan Batres(Microsoft), Rebecca Bergersen(Iona), Allen
2037 Brown(Microsoft), Michael Conner(IBM), George Copeland(Microsoft), Francisco Curbera(IBM),
2038 Paul Fremantle(IBM), Steve Graham(IBM), Pat Helland(Microsoft), Rick Hill(Microsoft), Scott
2039 Hinkelman(IBM), Tim Holloway(IBM), Efim Hudis(Microsoft), David Ingham(Microsoft), Gopal
2040 Kakivaya(Microsoft), Johannes Klein(Microsoft), Frank Leymann(IBM), Martin Nally(IBM), Peter
2041 Niblett(IBM), Jeffrey Schlimmer(Microsoft), James Snell(IBM), Keith Stobie(Microsoft), Satish
2042 Thatte(Microsoft), Stephen Todd(IBM), Sanjiva Weerawarana(IBM), Roger Wolter(Microsoft).

2043 The following individuals were members of the committee during the development of this specification:

2044 Abbie Barbir(Nortel), Charlton Barreto(Adobe), Stefan Batres(Microsoft), Hamid Ben
2045 Malek(Fujitsu), Andreas Bjarlestam(Ericsson), Toufic Boubez(Layer 7), Doug Bunting(Sun), Lloyd
2046 Burch(Novell), Steve Carter(Novell), Martin Chapman(Oracle), Dave Chappell(Sonic), Paul
2047 Cotton(Microsoft), Glen Daniels(Sonic), Doug Davis(IBM), Blake Dournaee(Intel), Jacques
2048 Durand(Fujitsu), Colleen Evans(Microsoft), Christopher Ferris(IBM), Paul Fremantle(WSO2),
2049 Robert Freund(Hitachi), Peter Furniss(Erebor), Marc Goodner(Microsoft), Alastair
2050 Green(Choreology), Mike Grogan(Sun), Ondrej Hrebicek(Microsoft), Kazunori Iwasa(Fujitsu),
2051 Chamikara Jayalath(WSO2), Lei Jin(BEA), Ian Jones(BTplc), Anish Karmarkar(Oracle), Paul
2052 Knight(Nortel), Dan Leshchiner(Tibco), Mark Little(JBoss), Lily Liu(webMethods), Matt
2053 Lovett(IBM), Ashok Malhotra(Oracle), Jonathan Marsh(Microsoft), Daniel Millwood(IBM), Jeff
2054 Mischkinsky(Oracle), Nilo Mitra(Ericsson), Peter Niblett(IBM), Duane Nickull(Adobe), Eisaku
2055 Nishiyama(Hitachi), Dave Orchard(BEA), Chouthri Palanisamy(NEC), Sanjay Patil(SAP), Gilbert
2056 Pilz(BEA), Martin Raepfle(SAP), Eric Rajkovic(Oracle), Stefan Rossmannith(SAP), Tom
2057 Rutt(Fujitsu), Rich Salz(IBM), Shivajee Samdarshi(Tibco), Vladimir Videlov(SAP), Claus von
2058 Riegen(SAP), Pete Wenzel(Sun), Steve Winkler(SAP), Ümit Yalçinalp(SAP), Nobuyuki
2059 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

2060 **Appendix G. Notices**

2061 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
2062 might be claimed to pertain to the implementation or use of the technology described in this document or
2063 the extent to which any license under such rights might or might not be available; neither does it represent
2064 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to
2065 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made
2066 available for publication and any assurances of licenses to be made available, or the result of an attempt
2067 made to obtain a general license or permission for the use of such proprietary rights by implementors or
2068 users of this specification, can be obtained from the OASIS Executive Director.

2069 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or
2070 other proprietary rights which may cover technology that may be required to implement this specification.
2071 Please address the information to the OASIS Executive Director.

2072 Copyright (C) OASIS Open (2006). All Rights Reserved.

2073 This document and translations of it may be copied and furnished to others, and derivative works that
2074 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and
2075 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and
2076 this paragraph are included on all such copies and derivative works. However, this document itself may
2077 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as
2078 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights
2079 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it
2080 into languages other than English.

2081 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
2082 or assigns.

2083 This document and the information contained herein is provided on an "AS IS" basis and OASIS
2084 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
2085 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR
2086 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.