



# 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](mailto:dug@us.ibm.com)>  
10 Anish Karmarkar, Oracle <[Anish.Karmarkar@oracle.com](mailto:Anish.Karmarkar@oracle.com)>  
11 Gilbert Pilz, BEA <[gpilz@bea.com](mailto:gpilz@bea.com)>  
12 Steve Winkler, SAP <[steve.winkler@sap.com](mailto:steve.winkler@sap.com)>  
13 Ümit Yalçınalp, SAP <[umit.yalcinalp@sap.com](mailto: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	<a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>
S12	<a href="http://www.w3.org/2003/05/soap-envelope">http://www.w3.org/2003/05/soap-envelope</a>
wsrn	<a href="http://docs.oasis-open.org/ws-rx/wsrn/200608">http://docs.oasis-open.org/ws-rx/wsrn/200608</a>
wsa	<a href="http://www.w3.org/2005/08/addressing">http://www.w3.org/2005/08/addressing</a>
wsaw	<a href="http://www.w3.org/2006/05/addressing/wsdl">http://www.w3.org/2006/05/addressing/wsdl</a>
wsse	<a href="http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd">http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd</a>
xs	<a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a>

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.

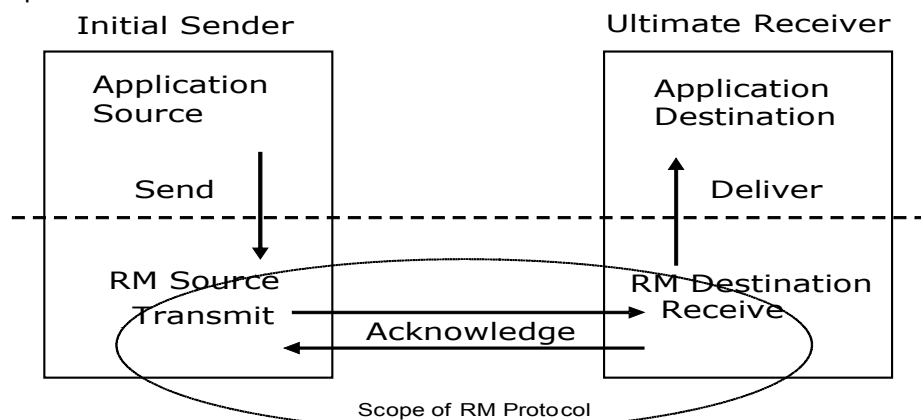
## 161 2 Reliable Messaging Model

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 2<sup>nd</sup> message in the Sequence is lost in transit.
- 249 6. The 3<sup>rd</sup> 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. The `TerminateSequence` message  
263 indicates that message number 3 was the last message in the Sequence. The RM Destination then  
264 and reclaims any resources associated with the Sequence.
- 265 11. The RM Destination Receives the `TerminateSequence` message indicating that the RM Source will  
266 not be sending any more messages. The RM Destination sends a `TerminateSequenceResponse`  
267 message to the RM Source and reclaims any resources associated with the Sequence.
- 268 The RM Source will expect to Receive Acknowledgements from the RM Destination during the course of a  
269 message exchange at occasions described in Section 3 below. Should an Acknowledgement not be  
270 Received in a timely fashion, the RM Source MUST re-transmit the message since either the message or  
271 the associated Acknowledgement might have been lost. Since the nature and dynamic characteristics of  
272 the underlying transport and potential intermediaries are unknown in the general case, the timing of re-  
273 transmissions cannot be specified. Additionally, over-aggressive re-transmissions have been  
274 demonstrated to cause transport or intermediary flooding which are counterproductive to the intention of  
275 providing a reliable exchange of messages. Consequently, implementers are encouraged to utilize  
276 adaptive mechanisms that dynamically adjust re-transmission time and the back-off intervals that are  
277 appropriate to the nature of the transports and intermediaries envisioned. For the case of TCP/IP  
278 transports, a mechanism similar to that described as RTTM in RFC 1323 [RTTM] SHOULD be  
279 considered.
- 280 Now that the basic model has been outlined, the details of the elements used in this protocol are now  
281 provided in Section 3.

## 282 **3 RM Protocol Elements**

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

### 285 **3.1 Considerations on the Use of Extensibility Points**

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

### 290 **3.2 Considerations on the Use of "Piggy-Backing"**

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

### 296 **3.3 Composition with WS-Addressing**

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

- 299 1. When an Endpoint generates a message that carries an RM protocol element, that is defined in  
300 section 3 below, in the body of a SOAP envelope that Endpoint MUST include in that envelope a  
301 `wsa:Action` SOAP header block whose value is an IRI that is a concatenation of the WS-RM  
302 namespace URI, followed by a "/", followed by the value of the local name of the child element of  
303 the SOAP body. For example, for a Sequence creation request message as described in section  
304 3.4 below, the value of the `wsa:Action` IRI would be:

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

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

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

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

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

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

### 314 **3.4 Sequence Creation**

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

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

322 The following exemplar defines the `CreateSequence` syntax:

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

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

338 `/wsrm:CreateSequence`

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

343 `/wsrm:CreateSequence/wsrm:AcksTo`

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

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

353 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

361 `/wsrm:CreateSequence/wsrm:Offer`

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

364 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier`

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

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

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

370 /wsmr:CreateSequence/wsmr:Offer/wsmr:Endpoint

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

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

382 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

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

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

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

389 /wsmr:CreateSequence/wsmr:Offer/wsmr:IncompleteSequenceBehavior

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

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

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

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

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

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

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

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

406 /wsmr:CreateSequence/{any}

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

409 /wsmr:CreateSequence/@{any}

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

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

416 The following exemplar defines the `CreateSequenceResponse` syntax:

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

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

430 /wsmr:CreateSequenceResponse

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

434 /wsmr:CreateSequenceResponse/wsmr:Identifier

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

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

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

441 /wsmr:CreateSequenceResponse/wsmr:Expires

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

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

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

453 /wsmr:CreateSequenceResponse/wsmr:IncompleteSequenceBehavior

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

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

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

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

464 /wsmr:CreateSequenceResponse/wsmr:Accept

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

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

470 /wsmr:CreateSequenceResponse/wsmr:Accept/wsmr:AcksTo

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

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

479 /wsmr:CreateSequenceResponse/wsmr:Accept/{any}

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

482 /wsmr:CreateSequenceResponse/wsmr:Accept/@{any}

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

485 /wsmr:CreateSequenceResponse/{any}

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

488 /wsmr:CreateSequenceResponse/@{any}

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

### 491 3.5 Closing A Sequence

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

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

506 In order to allow the RM Destination to determine if it has received all of the messages in a Sequence, the  
507 RM Source SHOULD include the `LastMsgNumber` element in any `CloseSequence` messages it sends.  
508 The RM Destination can use this information, for example, to implement the behavior indicated by  
509 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`. The value of the  
510 `LastMsgNumber` element MUST be the same in all the `CloseSequence` messages for a single Sequence.

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

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

519 The following exemplar defines the `CloseSequence` syntax:

```
520 <wsrm:CloseSequence ...>  
521   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
522   <wsrm>LastMsgNumber> wsrm:MessageNumberType </wsrm>LastMsgNumber> ?  
523   ...  
524 </wsrm:CloseSequence>
```

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

526 `/wsrm:CloseSequence`

527 This element is sent by an RM Source to indicate that the RM Destination MUST NOT accept any new  
528 messages for this Sequence.

529 `/wsrm:CloseSequence/wsrm:Identifier`

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

533 `/wsrm:CloseSequence/wsrm>LastMsgNumber`

534 The RM Source SHOULD include this element in any CloseSequence messages it sends. The  
535 LastMsgNumber element specifies the highest assigned message number of all the Sequence Traffic  
536 Messages for the Sequence being closed.

537 /wsmr:CloseSequence/wsmr:Identifier/{any}

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

540 /wsmr:CloseSequence/{any}

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

543 /wsmr:CloseSequence@{any}

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

546 A CloseSequenceResponse is sent in the body of a response message by an RM Destination in  
547 response to receipt of a CloseSequence request message. It indicates that the RM Destination has  
548 closed the Sequence.

549 The following exemplar defines the CloseSequenceResponse syntax:

```
550 <wsmr:CloseSequenceResponse ...>  
551   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
552   ...  
553 </wsmr:CloseSequenceResponse>
```

554 The following describes the content model of the CloseSequenceResponse element.

555 /wsmr:CloseSequenceResponse

556 This element is sent in the body of a response message by an RM Destination in response to receipt of a  
557 CloseSequence request message. It indicates that the RM Destination has closed the Sequence.

558 /wsmr:CloseSequenceResponse/wsmr:Identifier

559 The RM Destination MUST include this element in any CloseSequenceResponse message it sends. The  
560 RM Destination MUST set the value of this element to the absolute URI (conformant with RFC3986) of the  
561 Sequence that is being closed.

562 /wsmr:CloseSequenceResponse/wsmr:Identifier/{any}

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

565 /wsmr:CloseSequenceResponse/{any}

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

568 /wsmr:CloseSequenceResponse@{any}

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



## 571 3.6 Sequence Termination

572 When the RM Source has completed its use of the Sequence it sends a `TerminateSequence` element,  
573 in the body of a message, to the RM Destination to indicate that the Sequence is complete and that it will  
574 not be sending any further messages related to the Sequence. The RM Destination can safely reclaim any  
575 resources associated with the Sequence upon receipt of the `TerminateSequence` message. Under  
576 normal usage the RM Source will complete its use of the Sequence when all of the messages in the  
577 Sequence have been acknowledged. However, the RM Source is free to Terminate or Close a Sequence  
578 at any time regardless of the acknowledgement state of the messages.

579 In order to allow the RM Destination to determine if it has received all of the messages in a Sequence, the  
580 RM Source SHOULD include the `LastMsgNumber` element in any `TerminateSequence` messages it  
581 sends. The RM Destination can use this information, for example, to implement the behavior indicated by  
582 `/wsmr:CreateSequenceResponse/wsmr:IncompleteSequenceBehavior`. The value of the  
583 `LastMsgNumber` element in the `TerminateSequence` message MUST be equal to the value of the  
584 `LastMsgNumber` element in any `CloseSequence` message(s) sent by the RMS for the same Sequence.

585 The following exemplar defines the `TerminateSequence` syntax:

```
586 <wsmr:TerminateSequence ...>  
587   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
588   ....<wsmr>LastMsgNumber> wsmr:MessageNumberType </wsmr>LastMsgNumber> ?  
589   ...  
590 </wsmr:TerminateSequence>
```

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

592 `/wsmr:TerminateSequence`

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

598 `/wsmr:TerminateSequence/wsmr:Identifier`

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

602 `/wsmr:TerminateSequence/wsmr>LastMsgNumber`

603 The RM Source SHOULD include this element in any `TerminateSequence` messages it sends. The  
604 `LastMsgNumber` element specifies the highest assigned message number of all the Sequence Traffic  
605 Messages for the Sequence being closed.

606 `/wsmr:TerminateSequence/wsmr:Identifier/@{any}`

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

609 `/wsmr:TerminateSequence/{any}`

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

612 `/wsmr:TerminateSequence/@{any}`

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

615 A `TerminateSequenceResponse` is sent in the body of a response message by an RM Destination in  
616 response to receipt of a `TerminateSequence` request message. It indicates that the RM Destination has  
617 terminated the Sequence.

618 The following exemplar defines the `TerminateSequenceResponse` syntax:

```
619 <wsm:TerminateSequenceResponse ...>  
620   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>  
621   ...  
622 </wsm:TerminateSequenceResponse>
```

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

624 `/wsm:TerminateSequenceResponse`

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

628 `/wsm:TerminateSequenceResponse/wsm:Identifier`

629 The RM Destination MUST include this element in any `TerminateSequenceResponse` message it  
630 sends. The RM Destination MUST set the value of this element to the absolute URI (conformant with  
631 RFC3986) of the Sequence that is being terminated.

632 `/wsm:TerminateSequenceResponse/wsm:Identifier/@{any}`

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

635 `/wsm:TerminateSequenceResponse/{any}`

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

638 `/wsm:TerminateSequenceResponse/@{any}`

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

641 On receipt of a `TerminateSequence` message an RM Destination MUST respond with a corresponding  
642 `TerminateSequenceResponse` message or generate a fault `UnknownSequenceFault` if the  
643 Sequence is not known.

## 644 3.7 Sequences

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

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

652 A following exemplar defines its syntax:

```

653 <wsm:Sequence ...>
654   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>
655   <wsm:MessageNumber> wsm:MessageNumberType </wsm:MessageNumber>
656   ...
657 </wsm:Sequence>

```

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

659 `/wsm:Sequence`

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

666 `/wsm:Sequence/wsm:Identifier`

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

670 `/wsm:Sequence/wsm:Identifier/@{any}`

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

673 `/wsm:Sequence/wsm:MessageNumber`

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

678 `/wsm:Sequence/{any}`

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

681 `/wsm:Sequence/@{any}`

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

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

```

685 <wsm:Sequence>
686   <wsm:Identifier>http://example.com/abc</wsm:Identifier>
687   <wsm:MessageNumber>10</wsm:MessageNumber>
688 </wsm:Sequence>

```

### 689 **3.8 Request Acknowledgement**

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

692 The RM Source MAY request an Acknowledgement Message from the RM Destination at any time by  
693 transmitting an `AckRequested` header block independently or it MAY include an `AckRequested` header  
694 block in any message targeted to the RM Destination. An RM Destination that Receives a message that

695 contains an `AckRequested` header block MUST send a message containing a  
696 `SequenceAcknowledgement` header block to the `AcksTo` endpoint reference (see Section 3.4) for a  
697 known Sequence or else generate an `UnknownSequence` fault. If a non-mustUnderstand fault occurs  
698 when processing an RM header that was piggy-backed on another message, a fault MUST be generated,  
699 but the processing of the original message MUST NOT be affected. It is RECOMMENDED that the RM  
700 Destination return a `AcknowledgementRange` or `None` element instead of a `Nack` element (see Section  
701 3.9).

702 The following exemplar defines its syntax:

```
703 <wsmr:AckRequested ...>  
704   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
705   ...  
706 </wsmr:AckRequested>
```

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

708 `/wsmr:AckRequested`

709 This element requests an Acknowledgement for the identified Sequence.

710 `/wsmr:AckRequested/wsmr:Identifier`

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

714 `/wsmr:AckRequested/wsmr:Identifier/@{any}`

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

717 `/wsmr:AckRequested/{any}`

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

720 `/wsmr:AckRequested/@{any}`

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

## 723 3.9 Sequence Acknowledgement

724 The RM Destination informs the RM Source of successful message receipt using a  
725 `SequenceAcknowledgement` header block. The RM Destination MAY Transmit the  
726 `SequenceAcknowledgement` header block independently or it MAY include the  
727 `SequenceAcknowledgement` header block on any message targeted to the `AcksTo` EPR.  
728 Acknowledgements can be explicitly requested using the `AckRequested` directive (see Section 3.8). If a  
729 non-mustUnderstand fault occurs when processing an RM header that was piggy-backed on another  
730 message, a fault MUST be generated, but the processing of the original message MUST NOT be  
731 affected.

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

734 During creation of a Sequence the RM Source MAY specify the WS-Addressing anonymous IRI as the  
735 address of the `AcksTo` EPR for that Sequence. When the RM Source specifies the WS-Addressing

736 anonymous IRI as the address of the `AcksTo` EPR, the RM Destination MUST Transmit any  
737 `SequenceAcknowledgement` headers for the created Sequence in a SOAP envelope to be Transmitted  
738 on the protocol binding-specific back-channel. Such a channel is provided by the context of a Received  
739 message containing a SOAP envelope that contains a `Sequence` header block and/or an `AckRequested`  
740 header block for that same Sequence identifier. When the RM Destination receives an `AckRequested`  
741 header, and the `AckTo` EPR for that sequence is the WS-Addressing anonymous IRI, the RM Destination  
742 SHOULD respond on the protocol binding-specific back-channel provided by the Received message  
743 containing the `AckRequested` header block.

744 The following exemplar defines its syntax:

```
745 <wsrm:SequenceAcknowledgement ...>  
746   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
747   [ [ [ <wsrm:AcknowledgementRange ...  
748     Upper="wsrm:MessageNumberType"  
749     Lower="wsrm:MessageNumberType" /> +  
750     | <wsrm:None/> ]  
751     <wsrm:Final/> ? ]  
752     | <wsrm:Nack> wsrm:MessageNumberType </wsrm:Nack> + ]  
753  
754     ...  
755 </wsrm:SequenceAcknowledgement>
```

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

757 `/wsrm:SequenceAcknowledgement`

758 This element contains the Sequence Acknowledgement information.

759 `/wsrm:SequenceAcknowledgement/wsrm:Identifier`

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

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

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

768 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange`

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

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

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

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

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

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

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

783 /wsmr:SequenceAcknowledgement/wsmr:None

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

788 /wsmr:SequenceAcknowledgement/wsmr:Final

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

795 /wsmr:SequenceAcknowledgement/wsmr:Nack

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

805 /wsmr:SequenceAcknowledgement/{any}

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

808 /wsmr:SequenceAcknowledgement/@{any}

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

811 The following examples illustrate `SequenceAcknowledgement` elements:

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

```
813 <wsmr:SequenceAcknowledgement>  
814   <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>  
815   <wsmr:AcknowledgementRange Upper="10" Lower="1"/>  
816 </wsmr:SequenceAcknowledgement>
```

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

```
819 <wsmr:SequenceAcknowledgement>  
820   <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>  
821   <wsmr:AcknowledgementRange Upper="2" Lower="1"/>  
822   <wsmr:AcknowledgementRange Upper="6" Lower="4"/>  
823   <wsmr:AcknowledgementRange Upper="10" Lower="8"/>
```

824 </wsm:SequenceAcknowledgement>

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

```
826 <wsm:SequenceAcknowledgement>
827   <wsm:Identifier>http://example.com/abc</wsm:Identifier>
828   <wsm:Nack>3</wsm:Nack>
829 </wsm:SequenceAcknowledgement>
```

## 830 4 Faults

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

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

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

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

844 The definitions of faults use the following properties:

845 [Code] The fault code.

846 [Subcode] The fault subcode.

847 [Reason] The English language reason element.

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

851 Entities that generate WS-ReliableMessaging faults MUST set the [Code] property to either "Sender" or  
852 "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

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

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



```

873     [Detail]
874     ...
875     </S:Detail>
876     </S:Fault>
877     </S:Body>
878     </S:Envelope>

```

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

```

881 <S11:Envelope>
882   <S11:Header>
883     <wsrm:SequenceFault>
884       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
885       <wsrm:Detail> [Detail] </wsrm:Detail>
886       ...
887     </wsrm:SequenceFault>
888     <!-- Headers elided for brevity. -->
889   </S11:Header>
890   <S11:Body>
891     <S11:Fault>
892       <faultcode> [Code] </faultcode>
893       <faultstring> [Reason] </faultstring>
894     </S11:Fault>
895   </S11:Body>
896 </S11:Envelope>

```

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

```

899 <S11:Envelope>
900   <S11:Body>
901     <S11:Fault>
902       <faultcode> [Subcode] </faultcode>
903       <faultstring> [Reason] </faultstring>
904     </S11:Fault>
905   </S11:Body>
906 </S11:Envelope>

```

## 907 4.1 SequenceFault Element

908 The purpose of the `SequenceFault` element is to carry the specific details of a fault generated during  
909 the reliable messaging specific processing of a message belonging to a Sequence. WS-  
910 ReliableMessaging nodes MUST use the `SequenceFault` container only in conjunction with the SOAP  
911 1.1 fault mechanism. WS-ReliableMessaging nodes MUST NOT use the `SequenceFault` container in  
912 conjunction with the SOAP 1.2 binding.

913 The following exemplar defines its syntax:

```

914 <wsrm:SequenceFault ...>
915   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
916   <wsrm:Detail> ... </wsrm:Detail> ?
917   ...
918 </wsrm:SequenceFault>

```

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

920 `/wsrm:SequenceFault`

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

922 /wsm:SequenceFault/wsm:FaultCode

923 WS-ReliableMessaging nodes that generate a `SequenceFault` MUST set the value of this element to a  
924 qualified name from the set of fault [Subcodes] defined below.

925 /wsm:SequenceFault/wsm:Detail

926 This element, if present, carries application specific error information related to the fault being described.

927 /wsm:SequenceFault/wsm:Detail/{any}

928 The application specific error information related to the fault being described.

929 /wsm:SequenceFault/wsm:Detail/@{any}

930 The application specific error information related to the fault being described.

931 /wsm:SequenceFault/{any}

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

934 /wsm:SequenceFault/@{any}

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

## 937 4.2 Sequence Terminated

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

940 Properties:

941 [Code] Sender or Receiver

942 [Subcode] wsm:SequenceTerminated

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

944 [Detail]

945 `<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.

## 946 4.3 Unknown Sequence

947 Properties:

948 [Code] Sender

949 [Subcode] wsm:UnknownSequence

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

951 [Detail]

952 `<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.

#### 953 **4.4 Invalid Acknowledgement**

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

956 [Code] Sender

957 [Subcode] wsrn:InvalidAcknowledgement

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

959 [Detail]

960 `<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.

#### 961 **4.5 Message Number Rollover**

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

963 Properties:

964 [Code] Sender

965 [Subcode] wsrn:MessageNumberRollover

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

967 [Detail]

```
968 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
969 <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.

## 970 4.6 Create Sequence Refused

971 Properties:

972 [Code] Sender or Receiver

973 [Subcode] wsrm:CreateSequenceRefused

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

975 [Detail]

```
976 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.

## 977 4.7 Sequence Closed

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

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

981 Properties:

982 [Code] Sender

983 [Subcode] wsrm:SequenceClosed

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

985 [Detail]

986 `<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.

## 987 **4.8 WSRM Required**

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

990 Properties:

991 [Code] Sender

992 [Subcode] wsrm:WSRMRequired

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

994 [Detail]

995 `xs:any`

## 996 **5 Security Threats and Countermeasures**

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

999 This specification makes no assumptions about the security requirements of the applications that use WS-  
1000 RM. However, once those requirements have been satisfied within a given operational context, the  
1001 addition of WS-RM to this operational context should not undermine the fulfillment of those requirements;  
1002 the use of WS-RM should not create additional attack vectors within an otherwise secure system.

1003 There are many other security concerns that one may need to consider when implementing or using this  
1004 protocol. The material below should not be considered as a "check list". Implementers and users of this  
1005 protocol are urged to perform a security analysis to determine their particular threat profile and the  
1006 appropriate responses to those threats.

1007 Implementers are also advised that there is a core tension between security and reliable messaging that  
1008 can be problematic if not addressed by implementations; one aspect of security is to prevent message  
1009 replay but one of the invariants of this protocol is to resend messages until they are acknowledged.  
1010 Consequently, if the security sub-system processes a message but a failure occurs before the reliable  
1011 messaging sub-system Receives that message, then it is possible (and likely) that the security sub-system  
1012 will treat subsequent copies as replays and discard them. At the same time, the reliable messaging sub-  
1013 system will likely continue to expect and even solicit the missing message(s). Care should be taken to  
1014 avoid and prevent this condition.

### 1015 **5.1 Threats and Countermeasures**

1016 The primary security requirement of this protocol is to protect the specified semantics and protocol  
1017 invariants against various threats. The following sections describe several threats to the integrity and  
1018 operation of this protocol and provide some general outlines of countermeasures to those threats.  
1019 Implementers and users of this protocol should keep in mind that all threats are not necessarily applicable  
1020 to all operational contexts.

#### 1021 **5.1.1 Integrity Threats**

1022 In general, any mechanism which allows an attacker to alter the information in a Sequence Traffic  
1023 Message, Sequence Lifecycle Message, Acknowledgement Messages, Acknowledgement Request, or  
1024 Sequence-related fault, or which allows an attacker to alter the correlation of a RM Protocol Header Block  
1025 to its intended message represents a threat to the WS-RM protocol.

1026 For example, if an attacker is able to swap `Sequence` headers on messages in transit between the RM  
1027 Source and RM Destination then they have undermined the implementation's ability to guarantee the first  
1028 invariant described in Section 2.3. The result is that there is no way of guaranteeing that messages will be  
1029 Delivered to the Application Destination in the same order that they were sent by the Application Source.

##### 1030 **5.1.1.1 Countermeasures**

1031 Integrity threats are generally countered via the use of digital signatures some level of the communication  
1032 protocol stack. Note that, in order to counter header swapping attacks, the signature SHOULD include  
1033 both the SOAP body and any relevant SOAP headers (e.g. `Sequence` header). Because some headers  
1034 (`AckRequested`, `SequenceAcknowledgement`) are independent of the body of the SOAP message in which  
1035 they occur, implementations MUST allow for signatures that cover only these headers.

## 1036 **5.1.2 Resource Consumption Threats**

1037 The creation of a Sequence with an RM Destination consumes various resources on the systems used to  
1038 implement that RM Destination. These resources can include network connections, database tables,  
1039 message queues, etc. This behavior can be exploited to conduct denial of service attacks against an RM  
1040 Destination. For example, a simple attack is to repeatedly send `CreateSequence` messages to an RM  
1041 Destination. Another attack is to create a Sequence for a service that is known to require in-order  
1042 message Delivery and use this Sequence to send a stream of very large messages to that service,  
1043 making sure to omit message number “1” from that stream.

### 1044 **5.1.2.1 Countermeasures**

1045 There are a number of countermeasures against the described resource consumption threats. The  
1046 technique advocated by this specification is for the RM Destination to restrict the ability to create a  
1047 Sequence to a specific set of entities/principals. This reduces the number of potential attackers and, in  
1048 some cases, allows the identity of any attackers to be determined.

1049 The ability to restrict Sequence creation depends, in turn, upon the RM Destination's ability identify and  
1050 authenticate the RM Source that issued the `CreateSequence` message.

## 1051 **5.1.3 Sequence Spoofing Threats**

1052 Sequence spoofing is a class of threats in which the attacker uses knowledge of the `Identifier` for a  
1053 particular Sequence to forge Sequence Lifecycle or Traffic Messages. For example the attacker creates a  
1054 fake `TerminateSequence` message that references the target Sequence and sends this message to the  
1055 appropriate RM Destination. Some sequence spoofing attacks also require up-to-date knowledge of the  
1056 current `MessageNumber` for their target Sequence.

1057 In general any Sequence Lifecycle Message, RM Protocol Header Block, or sequence-correlated SOAP  
1058 fault (e.g. `InvalidAcknowledgement`) can be used by someone with knowledge of the Sequence identifier  
1059 to attack the Sequence. These attacks are “two-way” in that an attacker may choose to target the RM  
1060 Source by, for example, inserting a fake `SequenceAcknowledgement` header into a message that it sends  
1061 to the `AcksTo` EPR of an RM Source.

### 1062 **5.1.3.1 Sequence Hijacking**

1063 Sequence hijacking is a specific case of a sequence spoofing attack. The attacker attempts to inject  
1064 Sequence Traffic Messages into an existing Sequence by inserting fake `Sequence` headers into those  
1065 messages.

1066 Note that “sequence hijacking” should not be equated with “security session hijacking”. Although a  
1067 Sequence may be bound to some form of a security session in order to counter the threats described in  
1068 this section, applications MUST NOT rely on WS-RM-related information to make determinations about  
1069 the identity of the entity that created a message; applications SHOULD rely only upon information that is  
1070 established by the security infrastructure to make such determinations. Failure to observe this rule  
1071 creates, among other problems, a situation in which the absence of WS-RM may deprive an application of  
1072 the ability to authenticate its peers even though the necessary security processing has taken place.

### 1073 **5.1.3.2 Countermeasures**

1074 There are a number of countermeasures against sequence spoofing threats. The technique advocated by  
1075 this specification is to consider the Sequence to be a shared resource that is jointly owned by the RM

1076 Source that initiated its creation (i.e. that sent the `CreateSequence` message) and the RM Destination that  
1077 serves as its terminus (i.e. that sent the `CreateSequenceResponse` message). To counter sequence  
1078 spoofing attempts the RM Destination SHOULD ensure that every message or fault that it Receives that  
1079 refers to a particular Sequence originated from the RM Source that jointly owns the referenced Sequence.  
1080 For its part the RM Source SHOULD ensure that every message or fault that it Receives that refers to a  
1081 particular Sequence originated from the RM Destination that jointly owns the referenced Sequence.

1082 For the RM Destination to be able to identify its sequence peer it MUST be able to identify and  
1083 authenticate the entity that sent the `CreateSequence` message. Similarly for the RM Source to identify its  
1084 sequence peer it MUST be able to identify and authenticate the entity that sent the  
1085 `CreateSequenceResponse` message. For either the RM Destination or the RM Source to determine if a  
1086 message was sent by its sequence peer it MUST be able to identify and authenticate the initiator of that  
1087 message and, if necessary, correlate this identity with the sequence peer identity established at sequence  
1088 creation time.

## 1089 **5.2 Security Solutions and Technologies**

1090 The security threats described in the previous sections are neither new nor unique. The solutions that  
1091 have been developed to secure other SOAP-based protocols can be used to secure WS-RM as well. This  
1092 section maps the facilities provided by common web services security solutions against countermeasures  
1093 described in the previous sections.

1094 Before continuing this discussion, however, some examination of the underlying requirements of the  
1095 previously described countermeasures is necessary. Specifically it should be noted that the technique  
1096 described in Section 5.1.2.1 has two components. Firstly, the RM Destination identifies and authenticates  
1097 the issuer of a `CreateSequence` message. Secondly, the RM Destination performs an authorization check  
1098 against this authenticated identity and determines if the RM Source is permitted to create Sequences with  
1099 the RM Destination. Since the facilities for performing this authorization check (runtime infrastructure,  
1100 policy frameworks, etc.) lie completely within the domain of individual implementations, any discussion of  
1101 such facilities is considered to be beyond the scope of this specification.

### 1102 **5.2.1 Transport Layer Security**

1103 This section describes how the facilities provided by SSL/TLS [RFC 4346] can be used to implement the  
1104 countermeasures described in the previous sections. The use of SSL/TLS is subject to the constraints  
1105 defined in Section 4 of the Basic Security Profile 1.0 [BSP 1.0].

1106 The description provided here is general in nature and is not intended to serve as a complete definition on  
1107 the use of SSL/TLS to protect WS-RM. In order to interoperate implementations need to agree on the  
1108 choice of features as well as the manner in which they will be used. The mechanisms described in the  
1109 Web Services Security Policy Language [SecurityPolicy] MAY be used by services to describe the  
1110 requirements and constraints of the use of SSL/TLS.

#### 1111 **5.2.1.1 Model**

1112 The basic model for using SSL/TLS is as follows:

- 1113 1. The RM Source establishes an SSL/TLS session with the RM Destination.
- 1114 2. The RM Source uses this SSL/TLS session to send a `CreateSequence` message to the RM  
1115 Destination.



- 1116 3. The RM Destination establishes an SSL/TLS session with the RM Source and sends an  
1117 asynchronous `CreateSequenceResponse` using this session. Alternately it may respond with a  
1118 synchronous `CreateSequenceResponse` using the session established in (1).
- 1119 4. For the lifetime of the Sequence the RM Source uses the SSL/TLS session from (1) to Transmit  
1120 any and all messages or faults that refer to that Sequence.
- 1121 5. For the lifetime of the Sequence the RM Destination either uses the SSL/TLS session established  
1122 in (3) to Transmit any and all messages or faults that refer to that Sequence or, for synchronous  
1123 exchanges, the RM Destination uses the SSL/TLS session established in (1).

### 1124 5.2.1.2 Countermeasure Implementation

1125 Used in its simplest fashion (without relying upon any authentication mechanisms), SSL/TLS provides the  
1126 necessary integrity qualities to counter the threats described in Section 5.1.1. Note, however, that the  
1127 nature of SSL/TLS limits the scope of this integrity protection to a single transport level session. If  
1128 SSL/TLS is the only mechanism used to provide integrity, any intermediaries between the RM Source and  
1129 the RM Destination MUST be trusted to preserve the integrity of the messages that flow through them.

1130 As noted, the technique described in Sections 5.1.2.1 involves the use of authentication. This specification  
1131 advocates either of two mechanisms for authenticating entities using SSL/TLS. In both of these methods  
1132 the SSL/TLS server (the party accepting the SSL/TLS connection) authenticates itself to the SSL/TLS  
1133 client using an X.509 certificate that is exchanged during the SSL/TLS handshake.

- 1134 • **HTTP Basic Authentication:** This method of authentication presupposes that a SOAP/HTTP  
1135 binding is being used as part of the protocol stack beneath WS-RM. Subsequent to the  
1136 establishment of the SSL/TLS session, the sending party authenticates itself to the receiving party  
1137 using HTTP Basic Authentication [RFC 2617]. For example, a RM Source might authenticate itself  
1138 to a RM Destination (e.g. when transmitting a Sequence Traffic Message) using BasicAuth.  
1139 Similarly the RM Destination might authenticate itself to the RM Source (e.g. when sending an  
1140 Acknowledgement) using BasicAuth.
- 1141 • **SSL/TLS Client Authentication:** In this method of authentication, the party initiating the  
1142 connection authenticates itself to the party accepting the connection using an X.509 certificate  
1143 that is exchanged during the SSL/TLS handshake.

1144 To implement the countermeasures described in section 5.1.2.1 the RM Source must authenticate itself  
1145 using one the above mechanisms. The authenticated identity can then be used to determine if the RM  
1146 Source is authorized to create a Sequence with the RM Destination.

1147 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring  
1148 an RM node's Sequence peer to be equivalent to their SSL/TLS session peer. This allows the  
1149 authorization decisions described in section 5.1.3.2 to be based on SSL/TLS session identity rather than  
1150 on authentication information. For example, an RM Destination can determine that a Sequence Traffic  
1151 Message rightfully belongs to its referenced Sequence if that message arrived over the same SSL/TLS  
1152 session that was used to carry the `CreateSequence` message for that Sequence. Note that requiring a  
1153 one-to-one relationship between SSL/TLS session peer and Sequence peer constrains the lifetime of a  
1154 SSL/TLS-protected Sequence to be less than or equal to the lifetime of the SSL/TLS session that is used  
1155 to protect that Sequence.

1156 This specification does not preclude the use of other methods of using SSL/TLS to implement the  
1157 countermeasures (such as associating specific authentication information with a Sequence) although such  
1158 methods are not covered by this document.

1159 Issues specific to the life-cycle management of SSL/TLS sessions (such as the resumption of a SSL/TLS  
1160 session) are outside the scope of this specification.

## 1161 **5.2.2 SOAP Message Security**

1162 The mechanisms described in WS-Security may be used in various ways to implement the  
1163 countermeasures described in the previous sections. This specification advocates using the protocol  
1164 described by WS-SecureConversation [[SecureConversation](#)] (optionally in conjunction with WS-Trust  
1165 [[Trust](#)]) as a mechanism for protecting Sequences. The use of WS-Security (as an underlying component  
1166 of WS-SecureConversation) is subject to the constraints defined in the Basic Security Profile 1.0.

1167 The description provided here is general in nature and is not intended to serve as a complete definition on  
1168 the use of WS-SecureConversation/WS-Trust to protect WS-RM. In order to interoperate implementations  
1169 need to agree on the choice of features as well as the manner in which they will be used. The  
1170 mechanisms described in the Web Services Security Policy Language MAY be used by services to  
1171 describe the requirements and constraints of the use of WS-SecureConversation.

### 1172 **5.2.2.1 Model**

1173 The basic model for using WS-SecureConversation is as follows:

- 1174 1. The RM Source and the RM Destination create a WS-SecureConversation security context. This  
1175 may involve the participation of third parties such as a security token service. The tokens  
1176 exchanged may contain authentication claims (e.g. X.509 certificates or Kerberos service tickets).
- 1177 2. During the `CreateSequence` exchange, the RM Source SHOULD explicitly identify the security  
1178 context that will be used to protect the Sequence. This is done so that, in cases where the  
1179 `CreateSequence` message is signed by more than one security context, the RM Source can  
1180 indicate which security context should be used to protect the newly created Sequence.
- 1181 3. For the lifetime of the Sequence the RM Source and the RM Destination use the session key(s)  
1182 associated with the security context to sign (as defined by WS-Security) at least the body and any  
1183 relevant WS-RM-defined headers of any and all messages or faults that refer to that Sequence.

### 1184 **5.2.2.2 Countermeasure Implementation**

1185 Without relying upon any authentication information, the per-message signatures provide the necessary  
1186 integrity qualities to counter the threats described in Section 5.1.1.

1187 To implement the countermeasures described in section 5.1.2.1 some mutually agreed upon form of  
1188 authentication claims must be provided by the RM Source to the RM Destination during the establishment  
1189 of the Security Context. These claims can then be used to determine if the RM Source is authorized to  
1190 create a Sequence with the RM Destination.

1191 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring  
1192 an RM node's Sequence peer to be equivalent to their security context session peer. This allows the  
1193 authorization decisions described in section 5.1.3.2 to be based on the identity of the message's security  
1194 context rather than on any authentication claims that may have been established during security context  
1195 initiation. Note that other methods of using WS-SecureConversation to implement the countermeasures  
1196 (such as associating specific authentication claims to a Sequence) are possible but not covered by this  
1197 document.

1198 As with transport security, the requisite equivalence of a security context peer and with a Sequence peer  
1199 limits the lifetime of a Sequence to the lifetime of the protecting security context. Unlike transport security,

1200 the association between a Sequence and its protecting security context cannot always be established  
1201 implicitly at Sequence creation time. This is due to the fact that the `CreateSequence` and  
1202 `CreateSequenceResponse` messages may be signed by more than one security context.

1203 Issues specific to the life-cycle management of WS-SecureConversation security contexts (such as  
1204 amending or renewing contexts) are outside the scope of this specification.

## 1205 6 Securing Sequences

1206 As noted in Section 5, the RM Source and RM Destination should be able to protect their shared  
1207 Sequences against the threat of Sequence Spoofing attacks. There are a number of OPTIONAL means of  
1208 achieving this objective depending upon the underlying security infrastructure.

### 1209 6.1 Securing Sequences Using WS-Security

1210 One mechanism for protecting a Sequence is to include a security token using a  
1211 `wsse:SecurityTokenReference` element from WS-Security (see section 9 in WS-  
1212 SecureConversation) in the `CreateSequence` element. This establishes an association between the  
1213 created (and, if present, offered) Sequence(s) and the referenced security token, such that the RM Source  
1214 and Destination MUST use the security token as the basis for authorization of all subsequent interactions  
1215 related to the Sequence(s). The `wsse:SecurityTokenReference` explicitly identifies the token as  
1216 there may be more than one token on a `CreateSequence` message or inferred from the communication  
1217 context (e.g. transport protection).

1218 It is RECOMMENDED that a message independent referencing mechanism be used to identify the token,  
1219 if the token being referenced supports such mechanism.

1220 The following exemplar defines the `CreateSequence` syntax when extended to include a  
1221 `wsse:SecurityTokenReference`:

```
1222 <wsrm:CreateSequence ...>  
1223   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>  
1224   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1225   <wsrm:Offer ...>  
1226     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
1227     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>  
1228     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1229     <wsrm:IncompleteSequenceBehavior>  
1230       wsrml:IncompleteSequenceBehaviorType  
1231     </wsrm:IncompleteSequenceBehavior> ?  
1232     ...  
1233   </wsrm:Offer> ?  
1234   ...  
1235   <wsse:SecurityTokenReference>  
1236     ...  
1237   </wsse:SecurityTokenReference> ?  
1238   ...  
1239 </wsrm:CreateSequence>
```

1240 The following describes the content model of the additional `CreateSequence` elements.

1241 `/wsrm:CreateSequence/wsse:SecurityTokenReference`

1242 This element uses the extensibility mechanism defined for the `CreateSequence` element (defined in  
1243 section 3.4) to communicate an explicit reference to the security token, using a  
1244 `wsse:SecurityTokenReference` as documented in WS-Security, that the RM Source and Destination  
1245 MUST use to authorize messages for the created (and, if present, the offered) Sequence(s). All  
1246 subsequent messages related to the created (and, if present, the offered) Sequence(s) MUST  
1247 demonstrate proof-of-possession of the secret associated with the token (e.g., by using or deriving from a  
1248 private or secret key).

1249 When a RM Source transmits a `CreateSequence` that has been extended to include a  
1250 `wsse:SecurityTokenReference` it SHOULD ensure that the RM Destination both understands and  
1251 will conform to the requirements listed above. In order to achieve this, the RM Source SHOULD include

1252 the `UsesSequenceSTR` element as a SOAP header block within the `CreateSequence` message. This  
1253 element MUST include a `soap:mustUnderstand` attribute with a value of 'true'. Thus the RM Source  
1254 can be assured that a RM Destination that responds with a `CreateSequenceResponse` understands  
1255 and conforms with the requirements listed above. Note that an RM Destination understanding this header  
1256 does not mean that it has processed and understood any WS-Security headers, the fault behavior defined  
1257 in WS-Security still applies.

1258 The following exemplar defines the `UsesSequenceSTR` syntax:

```
1259 <wsm:UsesSequenceSTR ... />
```

1260 The following describes the content model of the `UsesSequenceSTR` header block.

1261 `/wsm:UsesSequenceSTR`

1262 This element SHOULD be included as a SOAP header block in `CreateSequence` messages that use the  
1263 extensibility mechanism described above in this section. The `soap:mustUnderstand` attribute value  
1264 MUST be 'true'. The receiving RM Destination MUST understand and correctly implement the extension  
1265 described above or else generate a `soap:MustUnderstand` fault, thus aborting the requested  
1266 Sequence creation.

1267 The following is an example of a `CreateSequence` message using the  
1268 `wsse:SecurityTokenReference` extension and the `UsesSequenceSTR` header block:

```
1269 <soap:Envelope ...>  
1270   <soap:Header>  
1271     ...  
1272     <wsm:UsesSequenceSTR soap:mustUnderstand='true' />  
1273     ...  
1274   </soap:Header>  
1275   <soap:Body>  
1276     <wsm:CreateSequence>  
1277       <wsm:AcksTo>  
1278         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>  
1279       </wsm:AcksTo>  
1280       <wsse:SecurityTokenReference>  
1281         ...  
1282       </wsse:SecurityTokenReference>  
1283     </wsm:CreateSequence>  
1284   </soap:Body>  
1285 </soap:Envelope>
```

## 1286 6.2 Securing Sequences Using SSL/TLS

1287 One mechanism for protecting a Sequence is to bind the Sequence to the underlying SSL/TLS session(s).  
1288 The RM Source indicates to the RM Destination that a Sequence is to be bound to the underlying  
1289 SSL/TLS session(s) via the `UsesSequenceSSL` header block. If the RM Source wishes to bind a  
1290 Sequence to the underlying SSL/TLS sessions(s) it MUST include the `UsesSequenceSSL` element as a  
1291 SOAP header block within the `CreateSequence` message.

1292 The following exemplar defines the `UsesSequenceSSL` syntax:

```
1293 <wsm:UsesSequenceSSL soap:mustUnderstand="true" ... />
```

1294 The following describes the content model of the `UsesSequenceSSL` header block.

1295 `/wsm:UsesSequenceSSL`

1296 The RM Source MAY include this element as a SOAP header block of a `CreateSequence` message to  
1297 indicate to the RM Destination that the resulting Sequence is to be bound to the SSL/TLS session that was

1298 used to carry the `CreateSequence` message. If included, the RM Source MUST mark this header with a  
1299 `soap:mustUnderstand` attribute with a value of 'true'. The receiving RM Destination MUST understand  
1300 and correctly implement the functionality described in Section 5.2.1 or else generate a  
1301 `soap:MustUnderstand` fault, thus aborting the requested Sequence creation.

1302 Note that the use inclusion of the above header by the RM Source implies that all Sequence-related  
1303 information (Sequence Lifecycle or Acknowledgment messages or Sequence-related faults) flowing from  
1304 the RM Destination to the RM Source will be bound to the SSL/TLS session that is used to carry the  
1305 `CreateSequenceResponse` message.

## 1306 **7 References**

### 1307 **7.1 Normative**

#### 1308 **[KEYWORDS]**

1309 S. Bradner, "[Key words for use in RFCs to Indicate Requirement Levels](#)," RFC 2119, Harvard University,  
1310 March 1997

1311 <http://www.ietf.org/rfc/rfc2119.txt>

#### 1312 **[WS-RM Policy]**

1313 OASIS WS-RX Technical Committee Draft, "[Web Services ReliableMessaging Policy Assertion\( WS-RM  
1314 Policy\)](#)" October 2006

1315 <http://docs.oasis-open.org/ws-rx/wsrmp/200608/wsrmp-1.1-spec-wd-11.pdf>

#### 1316 **[SOAP 1.1]**

1317 W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.

1318 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

#### 1319 **[SOAP 1.2]**

1320 W3C Recommendation, "[SOAP Version 1.2 Part 1: Messaging Framework](#)" June 2003.

1321 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

#### 1322 **[URI]**

1323 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#)," RFC 3986,  
1324 MIT/LCS, U.C. Irvine, Xerox Corporation, January 2005.

1325 <http://ietf.org/rfc/rfc3986>

#### 1326 **[UUID]**

1327 P. Leach, M. Mealling, R. Salz, "[A Universally Unique Identifier \(UUID\) URN Namespace](#)," RFC 4122,  
1328 Microsoft, Refactored Networks - LLC, DataPower Technology Inc, July 2005

1329 <http://www.ietf.org/rfc/rfc4122.txt>

#### 1330 **[XML]**

1331 W3C Recommendation, "[Extensible Markup Language \(XML\) 1.0 \(Fourth Edition\)](#)", September 2006.

1332 <http://www.w3.org/TR/REC-xml/>

#### 1333 **[XML-ns]**

1334 W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.

1335 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>

#### 1336 **[XML-Schema Part1]**

1337 W3C Recommendation, "[XML Schema Part 1: Structures](#)," October 2004.

1338 <http://www.w3.org/TR/xmlschema-1/>

1339 **[XML-Schema Part2]**

1340 W3C Recommendation, "XML Schema Part 2: Datatypes," October 2004.

1341 <http://www.w3.org/TR/xmlschema-2/>

1342 **[XPath 1.0]**

1343 W3C Recommendation, "XML Path Language (XPath) Version 1.0," 16 November 1999.

1344 <http://www.w3.org/TR/xpath>

1345 **[WSDL 1.1]**

1346 W3C Note, "Web Services Description Language (WSDL 1.1)," 15 March 2001.

1347 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

1348 **[WS-Addressing]**

1349 W3C Recommendation, "Web Services Addressing 1.0 - Core", May 2006.

1350 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>

1351 W3C Recommendation, "Web Services Addressing 1.0 – SOAP Binding", May 2006.

1352 <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>

1353 **7.2 Non-Normative**

1354 **[BSP 1.0]**

1355 WS-I Working Group Draft. "Basic Security Profile Version 1.0," August 2006

1356 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>

1357 **[RDDL 2.0]**

1358 Jonathan Borden, Tim Bray, eds. "Resource Directory Description Language (RDDL) 2.0," January 2004

1359 <http://www.openhealth.org/RDDL/20040118/rddl-20040118.html>

1360 **[RFC 2617]**

1361 J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Loutonen, L. Stewart, "HTTP  
1362 Authentication: Basic and Digest Access Authentication," June 1999.

1363 <http://www.ietf.org/rfc/rfc2617.txt>

1364 **[RFC 4346]**

1365 T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1," April 2006.

1366 <http://www.ietf.org/rfc/rfc4346.txt>

1367 **[WS-Policy]**

1368 W3C Member Submission, "Web Services Policy Framework (WS-Policy)," April 2006.

1369 <http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/>

1370 **[WS-PolicyAttachment]**

1371 W3C Member Submission, "Web Services Policy Attachment (WS-PolicyAttachment)," April 2006.

1372 [http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-  
1373 20060425/](http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-20060425/)



1374 **[WS-Security]**

1375 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:  
1376 SOAP Message Security 1.0 (WS-Security 2004)", OASIS Standard 200401, March 2004.

1377 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

1378 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:  
1379 SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard 200602, February 2006.

1380 <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

1381 **[RTTM]**

1382 V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, May  
1383 1992.

1384 <http://www.rfc-editor.org/rfc/rfc1323.txt>

1385 **[SecurityPolicy]**

1386 G. Della-Libra, et. al. "Web Services Security Policy Language (WS-SecurityPolicy)", July 2005

1387 <http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf>

1388 **[SecureConversation]**

1389 S. Anderson, et al, "Web Services Secure Conversation Language (WS-SecureConversation)," February  
1390 2005.

1391 <http://schemas.xmlsoap.org/ws/2004/04/sc/>

1392 **[Trust]**

1393 S. Anderson, et al, "Web Services Trust Language (WS-Trust)," February 2005.

1394 <http://schemas.xmlsoap.org/ws/2005/02/trust>

## 1395 Appendix A. Schema

1396 The normative schema that is defined for WS-ReliableMessaging using [XML-Schema Part1] and [XML-  
1397 Schema Part2] is located at:

1398 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-200608.xsd>

1399 The following copy is provided for reference.

```
1400 <?xml version="1.0" encoding="UTF-8"?>
1401 <!--
1402 OASIS takes no position regarding the validity or scope of any intellectual
1403 property or other rights that might be claimed to pertain to the
1404 implementation or use of the technology described in this document or the
1405 extent to which any license under such rights might or might not be available;
1406 neither does it represent that it has made any effort to identify any such
1407 rights. Information on OASIS's procedures with respect to rights in OASIS
1408 specifications can be found at the OASIS website. Copies of claims of rights
1409 made available for publication and any assurances of licenses to be made
1410 available, or the result of an attempt made to obtain a general license or
1411 permission for the use of such proprietary rights by implementors or users of
1412 this specification, can be obtained from the OASIS Executive Director.
1413 OASIS invites any interested party to bring to its attention any copyrights,
1414 patents or patent applications, or other proprietary rights which may cover
1415 technology that may be required to implement this specification. Please
1416 address the information to the OASIS Executive Director.
1417 Copyright © OASIS Open 2002-2006. All Rights Reserved.
1418 This document and translations of it may be copied and furnished to others,
1419 and derivative works that comment on or otherwise explain it or assist in its
1420 implementation may be prepared, copied, published and distributed, in whole or
1421 in part, without restriction of any kind, provided that the above copyright
1422 notice and this paragraph are included on all such copies and derivative
1423 works. However, this document itself does not be modified in any way, such as
1424 by removing the copyright notice or references to OASIS, except as needed for
1425 the purpose of developing OASIS specifications, in which case the procedures
1426 for copyrights defined in the OASIS Intellectual Property Rights document must
1427 be followed, or as required to translate it into languages other than English.
1428 The limited permissions granted above are perpetual and will not be revoked by
1429 OASIS or its successors or assigns.
1430 This document and the information contained herein is provided on an "AS IS"
1431 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1432 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1433 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1434 FOR A PARTICULAR PURPOSE.
1435 -->
1436 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
1437 xmlns:wsa="http://www.w3.org/2005/08/addressing"
1438 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1439 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1440 elementFormDefault="qualified" attributeFormDefault="unqualified">
1441   <xs:import namespace="http://www.w3.org/2005/08/addressing"
1442   schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd"/>
1443   <!-- Protocol Elements -->
1444   <xs:complexType name="SequenceType">
1445     <xs:sequence>
1446       <xs:element ref="wsrm:Identifier"/>
1447       <xs:element name="MessageNumber" type="wsrm:MessageNumberType"/>
1448       <xs:any namespace="##other" processContents="lax" minOccurs="0"
1449 maxOccurs="unbounded"/>
1450     </xs:sequence>
```

```

1451     <xs:anyAttribute namespace="##other" processContents="lax"/>
1452 </xs:complexType>
1453 <xs:element name="Sequence" type="wsrm:SequenceType"/>
1454 <xs:element name="SequenceAcknowledgement">
1455     <xs:complexType>
1456         <xs:sequence>
1457             <xs:element ref="wsrm:Identifier"/>
1458             <xs:choice>
1459                 <xs:sequence>
1460                     <xs:choice>
1461                         <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
1462                             <xs:complexType>
1463                                 <xs:sequence/>
1464                                 <xs:attribute name="Upper" type="xs:unsignedLong"
1465 use="required"/>
1466                                 <xs:attribute name="Lower" type="xs:unsignedLong"
1467 use="required"/>
1468                             <xs:anyAttribute namespace="##other" processContents="lax"/>
1469                             </xs:complexType>
1470                         </xs:element>
1471                         <xs:element name="None">
1472                             <xs:complexType>
1473                                 <xs:sequence/>
1474                             </xs:complexType>
1475                         </xs:element>
1476                     </xs:choice>
1477                 <xs:element name="Final" minOccurs="0">
1478                     <xs:complexType>
1479                         <xs:sequence/>
1480                     </xs:complexType>
1481                 </xs:element>
1482             </xs:sequence>
1483             <xs:element name="Nack" type="xs:unsignedLong"
1484 maxOccurs="unbounded"/>
1485         </xs:choice>
1486         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1487 maxOccurs="unbounded"/>
1488     </xs:sequence>
1489     <xs:anyAttribute namespace="##other" processContents="lax"/>
1490 </xs:complexType>
1491 </xs:element>
1492 <xs:complexType name="AckRequestedType">
1493     <xs:sequence>
1494         <xs:element ref="wsrm:Identifier"/>
1495         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1496 maxOccurs="unbounded"/>
1497     </xs:sequence>
1498     <xs:anyAttribute namespace="##other" processContents="lax"/>
1499 </xs:complexType>
1500 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1501 <xs:element name="Identifier">
1502     <xs:complexType>
1503         <xs:annotation>
1504             <xs:documentation>
1505                 This type is for elements whose [children] is an anyURI and can have
1506 arbitrary attributes.
1507             </xs:documentation>
1508         </xs:annotation>
1509         <xs:simpleContent>
1510             <xs:extension base="xs:anyURI">
1511                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1512             </xs:extension>
1513         </xs:simpleContent>

```

```

1514     </xs:complexType>
1515 </xs:element>
1516 <xs:element name="Address">
1517   <xs:complexType>
1518     <xs:simpleContent>
1519       <xs:extension base="xs:anyURI">
1520         <xs:anyAttribute namespace="##other" processContents="lax"/>
1521       </xs:extension>
1522     </xs:simpleContent>
1523   </xs:complexType>
1524 </xs:element>
1525 <xs:simpleType name="MessageNumberType">
1526   <xs:restriction base="xs:unsignedLong">
1527     <xs:minInclusive value="1"/>
1528     <xs:maxInclusive value="9223372036854775807"/>
1529   </xs:restriction>
1530 </xs:simpleType>
1531 <!-- Fault Container and Codes -->
1532 <xs:simpleType name="FaultCodes">
1533   <xs:restriction base="xs:QName">
1534     <xs:enumeration value="wsrm:SequenceTerminated"/>
1535     <xs:enumeration value="wsrm:UnknownSequence"/>
1536     <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1537     <xs:enumeration value="wsrm:MessageNumberRollover"/>
1538     <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1539     <xs:enumeration value="wsrm:SequenceClosed"/>
1540     <xs:enumeration value="wsrm:WSRMRequired"/>
1541     <xs:enumeration value="wsrm:UnsupportedSelection"/>
1542   </xs:restriction>
1543 </xs:simpleType>
1544 <xs:complexType name="SequenceFaultType">
1545   <xs:sequence>
1546     <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1547     <xs:element name="Detail" type="wsrm:DetailType" minOccurs="0"/>
1548     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1549 maxOccurs="unbounded"/>
1550   </xs:sequence>
1551   <xs:anyAttribute namespace="##other" processContents="lax"/>
1552 </xs:complexType>
1553 <xs:complexType name="DetailType">
1554   <xs:sequence>
1555     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1556 maxOccurs="unbounded"/>
1557   </xs:sequence>
1558   <xs:anyAttribute namespace="##other" processContents="lax"/>
1559 </xs:complexType>
1560 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1561 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1562 <xs:element name="CreateSequenceResponse"
1563 type="wsrm:CreateSequenceResponseType"/>
1564 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1565 <xs:element name="CloseSequenceResponse"
1566 type="wsrm:CloseSequenceResponseType"/>
1567 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1568 <xs:element name="TerminateSequenceResponse"
1569 type="wsrm:TerminateSequenceResponseType"/>
1570 <xs:complexType name="CreateSequenceType">
1571   <xs:sequence>
1572     <xs:element ref="wsrm:AcksTo"/>
1573     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1574     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1575     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1576 maxOccurs="unbounded"/>

```

```

1577     <xs:annotation>
1578         <xs:documentation>
1579             It is the authors intent that this extensibility be used to
1580 transfer a Security Token Reference as defined in WS-Security.
1581         </xs:documentation>
1582     </xs:annotation>
1583 </xs:any>
1584 </xs:sequence>
1585 <xs:anyAttribute namespace="##other" processContents="lax"/>
1586 </xs:complexType>
1587 <xs:complexType name="CreateSequenceResponseType">
1588     <xs:sequence>
1589         <xs:element ref="wsrm:Identifier"/>
1590         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1591         <xs:element name="IncompleteSequenceBehavior"
1592 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1593         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1594         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1595 maxOccurs="unbounded"/>
1596     </xs:sequence>
1597     <xs:anyAttribute namespace="##other" processContents="lax"/>
1598 </xs:complexType>
1599 <xs:complexType name="CloseSequenceType">
1600     <xs:sequence>
1601         <xs:element ref="wsrm:Identifier"/>
1602         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1603 maxOccurs="unbounded"/>
1604     </xs:sequence>
1605     <xs:anyAttribute namespace="##other" processContents="lax"/>
1606 </xs:complexType>
1607 <xs:complexType name="CloseSequenceResponseType">
1608     <xs:sequence>
1609         <xs:element ref="wsrm:Identifier"/>
1610         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1611 maxOccurs="unbounded"/>
1612     </xs:sequence>
1613     <xs:anyAttribute namespace="##other" processContents="lax"/>
1614 </xs:complexType>
1615 <xs:complexType name="TerminateSequenceType">
1616     <xs:sequence>
1617         <xs:element ref="wsrm:Identifier"/>
1618         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1619 maxOccurs="unbounded"/>
1620     </xs:sequence>
1621     <xs:anyAttribute namespace="##other" processContents="lax"/>
1622 </xs:complexType>
1623 <xs:complexType name="TerminateSequenceResponseType">
1624     <xs:sequence>
1625         <xs:element ref="wsrm:Identifier"/>
1626         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1627 maxOccurs="unbounded"/>
1628     </xs:sequence>
1629     <xs:anyAttribute namespace="##other" processContents="lax"/>
1630 </xs:complexType>
1631 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1632 <xs:complexType name="OfferType">
1633     <xs:sequence>
1634         <xs:element ref="wsrm:Identifier"/>
1635         <xs:element name="Endpoint" type="wsa:EndpointReferenceType"/>
1636         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1637         <xs:element name="IncompleteSequenceBehavior"
1638 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1639         <xs:any namespace="##other" processContents="lax" minOccurs="0"

```

```

1640 maxOccurs="unbounded"/>
1641     </xs:sequence>
1642     <xs:anyAttribute namespace="##other" processContents="lax"/>
1643 </xs:complexType>
1644 <xs:complexType name="AcceptType">
1645     <xs:sequence>
1646         <xs:element ref="wsrm:AcksTo"/>
1647         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1648 maxOccurs="unbounded"/>
1649     </xs:sequence>
1650     <xs:anyAttribute namespace="##other" processContents="lax"/>
1651 </xs:complexType>
1652 <xs:element name="Expires">
1653     <xs:complexType>
1654         <xs:simpleContent>
1655             <xs:extension base="xs:duration">
1656                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1657             </xs:extension>
1658         </xs:simpleContent>
1659     </xs:complexType>
1660 </xs:element>
1661 <xs:simpleType name="IncompleteSequenceBehaviorType">
1662     <xs:restriction base="xs:string">
1663         <xs:enumeration value="DiscardEntireSequence"/>
1664         <xs:enumeration value="DiscardFollowingFirstGap"/>
1665         <xs:enumeration value="NoDiscard"/>
1666     </xs:restriction>
1667 </xs:simpleType>
1668 <xs:element name="UsesSequenceSTR">
1669     <xs:complexType>
1670         <xs:sequence/>
1671         <xs:anyAttribute namespace="##other" processContents="lax"/>
1672     </xs:complexType>
1673 </xs:element>
1674 <xs:element name="UsesSequenceSSL">
1675     <xs:complexType>
1676         <xs:sequence/>
1677         <xs:anyAttribute namespace="##other" processContents="lax"/>
1678     </xs:complexType>
1679 </xs:element>
1680 <xs:element name="UnsupportedElement">
1681     <xs:simpleType>
1682         <xs:restriction base="xs:QName"/>
1683     </xs:simpleType>
1684 </xs:element>
1685 </xs:schema>

```

## 1686 Appendix B. WSDL

1687 This WSDL describes the WS-RM protocol from the point of view of an RM Destination. In the case where  
1688 an endpoint acts both as an RM Destination and an RM Source, note that additional messages may be  
1689 present in exchanges with that endpoint.

1690 Also note that this WSDL is intended to describe the internal structure of the WS-RM protocol, and will not  
1691 generally appear in a description of a WS-RM-capable Web service. See WS-RM Policy [WS-RM Policy]  
1692 for a higher-level mechanism to indicate that WS-RM is engaged.

1693 The normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1694 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsd/wsrn-1.1-wsd-200608.wsd>

1695 The following non-normative copy is provided for reference.

```
1696 <?xml version="1.0" encoding="utf-8"?>
1697 <!--
1698 OASIS takes no position regarding the validity or scope of any intellectual
1699 property or other rights that might be claimed to pertain to the
1700 implementation or use of the technology described in this document or the
1701 extent to which any license under such rights might or might not be available;
1702 neither does it represent that it has made any effort to identify any such
1703 rights. Information on OASIS's procedures with respect to rights in OASIS
1704 specifications can be found at the OASIS website. Copies of claims of rights
1705 made available for publication and any assurances of licenses to be made
1706 available, or the result of an attempt made to obtain a general license or
1707 permission for the use of such proprietary rights by implementors or users of
1708 this specification, can be obtained from the OASIS Executive Director.
1709 OASIS invites any interested party to bring to its attention any copyrights,
1710 patents or patent applications, or other proprietary rights which may cover
1711 technology that may be required to implement this specification. Please
1712 address the information to the OASIS Executive Director.
1713 Copyright (c) OASIS Open 2002-2006. All Rights Reserved.
1714 This document and translations of it may be copied and furnished to others,
1715 and derivative works that comment on or otherwise explain it or assist in its
1716 implementation may be prepared, copied, published and distributed, in whole or
1717 in part, without restriction of any kind, provided that the above copyright
1718 notice and this paragraph are included on all such copies and derivative
1719 works. However, this document itself does not be modified in any way, such as
1720 by removing the copyright notice or references to OASIS, except as needed for
1721 the purpose of developing OASIS specifications, in which case the procedures
1722 for copyrights defined in the OASIS Intellectual Property Rights document must
1723 be followed, or as required to translate it into languages other than English.
1724 The limited permissions granted above are perpetual and will not be revoked by
1725 OASIS or its successors or assigns.
1726 This document and the information contained herein is provided on an "AS IS"
1727 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1728 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1729 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1730 FOR A PARTICULAR PURPOSE.
1731 -->
1732 <wsc:definitions xmlns:wsc="http://schemas.xmlsoap.org/wsc/"
1733 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1734 xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:rm="http://docs.oasis-
1735 open.org/ws-rx/wsrn/200608" xmlns:tns="http://docs.oasis-open.org/ws-
1736 rx/wsrn/200608/wsd" targetNamespace="http://docs.oasis-open.org/ws-
1737 rx/wsrn/200608/wsd">
1738 <wsc:types>
```

```

1739     <xs:schema>
1740         <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsr/200608"
1741 schemaLocation="http://docs.oasis-open.org/ws-rx/wsr/200608/wsr-1.1-schema-
1742 200608.xsd"/>
1743     </xs:schema>
1744 </wsdl:types>

1745     <wsdl:message name="CreateSequence">
1746         <wsdl:part name="create" element="rm:CreateSequence"/>
1747     </wsdl:message>
1748     <wsdl:message name="CreateSequenceResponse">
1749         <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1750     </wsdl:message>
1751     <wsdl:message name="CloseSequence">
1752         <wsdl:part name="close" element="rm:CloseSequence"/>
1753     </wsdl:message>
1754     <wsdl:message name="CloseSequenceResponse">
1755         <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1756     </wsdl:message>
1757     <wsdl:message name="TerminateSequence">
1758         <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1759     </wsdl:message>
1760     <wsdl:message name="TerminateSequenceResponse">
1761         <wsdl:part name="terminateResponse"
1762 element="rm:TerminateSequenceResponse"/>
1763     </wsdl:message>

1764     <wsdl:portType name="SequenceAbstractPortType">
1765         <wsdl:operation name="CreateSequence">
1766             <wsdl:input message="tns:CreateSequence" wsaw:Action="http://docs.oasis-
1767 open.org/ws-rx/wsr/200608/CreateSequence"/>
1768             <wsdl:output message="tns:CreateSequenceResponse"
1769 wsaw:Action="http://docs.oasis-open.org/ws-
1770 rx/wsr/200608/CreateSequenceResponse"/>
1771         </wsdl:operation>
1772         <wsdl:operation name="CloseSequence">
1773             <wsdl:input message="tns:CloseSequence" wsaw:Action="http://docs.oasis-
1774 open.org/ws-rx/wsr/200608/CloseSequence"/>
1775             <wsdl:output message="tns:CloseSequenceResponse"
1776 wsaw:Action="http://docs.oasis-open.org/ws-
1777 rx/wsr/200608/CloseSequenceResponse"/>
1778         </wsdl:operation>
1779         <wsdl:operation name="TerminateSequence">
1780             <wsdl:input message="tns:TerminateSequence"
1781 wsaw:Action="http://docs.oasis-open.org/ws-rx/wsr/200608/TerminateSequence"/>
1782             <wsdl:output message="tns:TerminateSequenceResponse"
1783 wsaw:Action="http://docs.oasis-open.org/ws-
1784 rx/wsr/200608/TerminateSequenceResponse"/>
1785         </wsdl:operation>
1786     </wsdl:portType>

1787 </wsdl:definitions>

```



## 1788 Appendix C. Message Examples

### 1789 Appendix C.1 Create Sequence

#### 1790 Create Sequence

```
1791 <?xml version="1.0" encoding="UTF-8"?>
1792 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1793 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1794 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1795   <S:Header>
1796     <wsa:MessageID>
1797       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1798     </wsa:MessageID>
1799     <wsa:To>http://example.com/serviceB/123</wsa:To>
1800     <wsa:Action>http://docs.oasis-open.org/ws-
1801 rx/wsmr/200608/CreateSequence</wsa:Action>
1802     <wsa:ReplyTo>
1803       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1804     </wsa:ReplyTo>
1805   </S:Header>
1806   <S:Body>
1807     <wsmr:CreateSequence>
1808       <wsmr:AcksTo>
1809         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1810       </wsmr:AcksTo>
1811     </wsmr:CreateSequence>
1812   </S:Body>
1813 </S:Envelope>
```

#### 1814 Create Sequence Response

```
1815 <?xml version="1.0" encoding="UTF-8"?>
1816 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1817 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1818 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1819   <S:Header>
1820     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1821     <wsa:RelatesTo>
1822       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1823     </wsa:RelatesTo>
1824     <wsa:Action>
1825       http://docs.oasis-open.org/ws-rx/wsmr/200608/CreateSequenceResponse
1826     </wsa:Action>
1827   </S:Header>
1828   <S:Body>
1829     <wsmr:CreateSequenceResponse>
1830       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1831     </wsmr:CreateSequenceResponse>
1832   </S:Body>
1833 </S:Envelope>
```

### 1834 Appendix C.2 Initial Transmission

1835 The following example WS-ReliableMessaging headers illustrate the message exchange in the above  
1836 figure. The three messages have the following headers; the third message is identified as the last  
1837 message in the Sequence:

1838 **Message 1**

```
1839 <?xml version="1.0" encoding="UTF-8"?>
1840 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1841 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1842 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1843   <S:Header>
1844     <wsa:MessageID>
1845       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1846     </wsa:MessageID>
1847     <wsa:To>http://example.com/serviceB/123</wsa:To>
1848     <wsa:From>
1849       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1850     </wsa:From>
1851     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1852     <wsmr:Sequence>
1853       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1854       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1855     </wsmr:Sequence>
1856   </S:Header>
1857   <S:Body>
1858     <!-- Some Application Data -->
1859   </S:Body>
1860 </S:Envelope>
```

1861 **Message 2**

```
1862 <?xml version="1.0" encoding="UTF-8"?>
1863 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1864 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1865 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1866   <S:Header>
1867     <wsa:MessageID>
1868       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1869     </wsa:MessageID>
1870     <wsa:To>http://example.com/serviceB/123</wsa:To>
1871     <wsa:From>
1872       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1873     </wsa:From>
1874     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1875     <wsmr:Sequence>
1876       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1877       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1878     </wsmr:Sequence>
1879   </S:Header>
1880   <S:Body>
1881     <!-- Some Application Data -->
1882   </S:Body>
1883 </S:Envelope>
```

1884 **Message 3**

```
1885 <?xml version="1.0" encoding="UTF-8"?>
1886 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1887 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1888 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1889   <S:Header>
1890     <wsa:MessageID>
1891       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1892     </wsa:MessageID>
1893     <wsa:To>http://example.com/serviceB/123</wsa:To>
1894     <wsa:From>
1895       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1896 </wsa:From>
1897 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1898 <wsrm:Sequence>
1899 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1900 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1901 </wsrm:Sequence>
1902 <wsrm:AckRequested>
1903 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1904 </wsrm:AckRequested>
1905 </S:Header>
1906 <S:Body>
1907 <!-- Some Application Data -->
1908 </S:Body>
1909 </S:Envelope>

```

### 1910 **Appendix C.3 First Acknowledgement**

1911 Message number 2 has not been accepted by the RM Destination due to some transmission error so it  
1912 responds with an Acknowledgement for messages 1 and 3:

```

1913 <?xml version="1.0" encoding="UTF-8"?>
1914 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1915 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1916 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1917 <S:Header>
1918 <wsa:MessageID>
1919 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1920 </wsa:MessageID>
1921 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1922 <wsa:From>
1923 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1924 </wsa:From>
1925 <wsa:Action>
1926 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1927 </wsa:Action>
1928 <wsrm:SequenceAcknowledgement>
1929 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1930 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1931 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1932 </wsrm:SequenceAcknowledgement>
1933 </S:Header>
1934 <S:Body/>
1935 </S:Envelope>

```

### 1936 **Appendix C.4 Retransmission**

1937 The RM Sourcediscovers that message number 2 was not accepted so it resends the message and  
1938 requests an Acknowledgement:

```

1939 <?xml version="1.0" encoding="UTF-8"?>
1940 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1941 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1942 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1943 <S:Header>
1944 <wsa:MessageID>
1945 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1946 </wsa:MessageID>
1947 <wsa:To>http://example.com/serviceB/123</wsa:To>
1948 <wsa:From>
1949 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1950 </wsa:From>

```

```

1951 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1952 <wsrm:Sequence>
1953 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1954 <wsrm:MessageNumber>2</wsrm:MessageNumber>
1955 </wsrm:Sequence>
1956 <wsrm:AckRequested>
1957 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1958 </wsrm:AckRequested>
1959 </S:Header>
1960 <S:Body>
1961 <!-- Some Application Data -->
1962 </S:Body>
1963 </S:Envelope>

```

## 1964 Appendix C.5 Termination

1965 The RM Destination now responds with an Acknowledgement for the complete Sequence which can then  
1966 be terminated:

```

1967 <?xml version="1.0" encoding="UTF-8"?>
1968 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1969 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1970 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1971 <S:Header>
1972 <wsa:MessageID>
1973 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1974 </wsa:MessageID>
1975 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1976 <wsa:From>
1977 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1978 </wsa:From>
1979 <wsa:Action>
1980 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1981 </wsa:Action>
1982 <wsrm:SequenceAcknowledgement>
1983 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1984 <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1985 </wsrm:SequenceAcknowledgement>
1986 </S:Header>
1987 <S:Body/>
1988 </S:Envelope>

```

## 1989 Terminate Sequence

```

1990 <?xml version="1.0" encoding="UTF-8"?>
1991 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1992 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1993 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1994 <S:Header>
1995 <wsa:MessageID>
1996 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1997 </wsa:MessageID>
1998 <wsa:To>http://example.com/serviceB/123</wsa:To>
1999 <wsa:Action>
2000 http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence
2001 </wsa:Action>
2002 <wsa:From>
2003 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2004 </wsa:From>
2005 </S:Header>
2006 <S:Body>
2007 <wsrm:TerminateSequence>

```

```
2008     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2009     </wsrm:TerminateSequence>
2010     </S:Body>
2011     </S:Envelope>
```

## 2012 Terminate Sequence Response

```
2013     <?xml version="1.0" encoding="UTF-8"?>
2014     <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2015     xmlns:wsm="http://docs.oasis-open.org/ws-rx/wsm/200608"
2016     xmlns:wsa="http://www.w3.org/2005/08/addressing">
2017     <S:Header>
2018     <wsa:MessageID>
2019     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
2020     </wsa:MessageID>
2021     <wsa:To>http://example.com/serviceA/789</wsa:To>
2022     <wsa:Action>
2023     http://docs.oasis-open.org/ws-rx/wsm/200608/TerminateSequenceResponse
2024     </wsa:Action>
2025     <wsa:RelatesTo>
2026     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
2027     </wsa:RelatesTo>
2028     <wsa:From>
2029     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2030     </wsa:From>
2031     </S:Header>
2032     <S:Body>
2033     <wsrm:TerminateSequenceResponse>
2034     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2035     </wsrm:TerminateSequenceResponse>
2036     </S:Body>
2037     </S:Envelope>
```

## 2038 Appendix D. State Tables

2039 This appendix specifies the non-normative state transition tables for RM Source and RM Destination.

2040 The state tables describe the lifetime of a sequence in both the RM Source and the RM Destination

2041 Legend:

2042 The first column of these tables contains the motivating event and has the following format:

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

2043 Where:

- 2044 ● Event Name: indicates the name of the event. Event Names surrounded by "<>" are optional as  
2045 described by the specification.
- 2046 ● [source]: indicates the source of the event; one of:
  - 2047 ● [msg] a Received message
  - 2048 ● [int]: an internal event such as the firing of a timer
  - 2049 ● [app]: the application
  - 2050 ● [unspec]: the source is unspecified

2051 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}

2052 Where:

- 2053 ● action to take: indicates that the state machine performs the following action. Actions surrounded  
2054 by "<>" are optional as described by the specification. "Xmit" is used as a short form for the word  
2055 "Transmit"
- 2056 ● [next state]: indicates the state to which the state machine will advance upon the performance of  
2057 the action. For ease of reading the next state "same" indicates that the state does not change.
- 2058 ● {ref} is a reference to the document section describing the behavior in this cell

2059 "N/A" in a cell indicates a state / event combination self-inconsistent with the state machine; should these  
2060 conditions occur, it would indicate an implementation error. A blank cell indicates that the behavior is not  
2061 described in this specification and does not indicate normal protocol operation. Implementations MAY  
2062 generate a Sequence Terminated fault (see section 4.2) in these circumstances. Robust implementations  
2063 MUST be able to operate in a stable manner despite the occurrence of unspecified event / state  
2064 combinations.

2065 Table 1 RM Source Sequence State Transition Table

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
<b>Create Sequence</b> [unspec] {3.4}	Xmit Create Sequence [Creating] {3.4}	N/A	N/A	N/A	N/A	N/A
<b>Create Sequence Response</b> [msg] {3.4}		Process Create Sequence Response [Created] {3.4}				
<b>Create Sequence Refused Fault</b> [msg] {3.4}		No action [None] {4.6}				
<b>Send message</b> [app] {2.1}	N/A	N/A	Xmit message [Same] {2}	No action [Same] {2}	N/A	N/A
<b>Retransmit of un-ack'd message</b> [int] {3.4}	N/A	N/A	Xmit message [Same] {2.4}	Xmit message [Same] {2.4}	N/A	N/A
<b>SeqAck (non-final)</b> [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}
<b>Nack</b> [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]
<b>Message Number Rollover Fault</b> [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]
<b>&lt;Close Sequence&gt;</b> [int] {3.5}	N/A		Xmit Close Sequence [Closing] {3.5}	N/A	N/A	N/A
<b>Close Sequence Response</b> [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}
<b>SeqAck (final)</b> [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]
<b>Sequence Closed Fault</b> [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}				
<b>Unknown Sequence Fault</b> [msg] {4.3}			Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}
<b>Sequence Terminated Fault</b> [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}
<b>Terminate Sequence</b> [int]	N/A	No action [None] {unspec}	Xmit Terminate Sequence [Terminating]	Xmit Terminate Sequence [Terminating]	Xmit Terminate Sequence [Terminating]	N/A
<b>Terminate Sequence Response</b> [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}				Terminate Sequence [None] {3.6}
<b>Expires exceeded</b> [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}
<b>Invalid Acknowledgment</b> [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}

2065 Table 2 RM Destination Sequence State Transition Table

Events	Sequence States		
	None	Created	Closed
<b>CreateSequence (successful)</b> [msg/int] {3.4}	Xmit Create Sequence Response [Created] {3.4}	N/A	N/A
<b>CreateSequence (unsuccessful)</b> [msg/int] {3.4}	Generate Create Sequence Refused Fault [None] {3.4}	N/A	N/A
<b>Message (with message number within range)</b> [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Accept Message; <Xmit SeqAck> [Same]	Generate Sequence Closed Fault (with SeqAck+Final) [Same] {3.5}
<b>Message (with message number outside of range)</b> [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}
<b>&lt;AckRequested&gt;</b> [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
<b>CloseSequence</b> [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}
<b>&lt;CloseSequence autonomously&gt;</b> [int]	N/A	No Action [Closed]	N/A
<b>TerminateSequence</b> [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Terminate Sequence Response [None] {3.6}	Xmit Terminate Sequence Response [None] {3.6}
<b>UnknownSequence Fault</b> [msg] {4.3}		Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}
<b>SequenceTerminated Fault</b> [msg] {4.2}		Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}
<b>Invalid Acknowledgement Fault</b> [msg] {4.4}	N/A		
<b>Expires exceeded</b> [int]	N/A	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}
<b>&lt;Seq Acknowledgement autonomously&gt;</b> [int] {3.9}	N/A	Xmit SeqAck [Same] {3.9}	Xmit SeqAck+Final [Same] {3.9}
<b>Non WSRM message when WSRM required</b> [msg] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}

## 2065 **Appendix E. Acknowledgments**

2066 This document is based on initial contribution to OASIS WS-RX Technical Committee by the following  
2067 authors:

2068 Ruslan Bilorusets(BEA), Don Box(Microsoft), Luis Felipe Cabrera(Microsoft), Doug Davis(IBM),  
2069 Donald Ferguson(IBM), Christopher Ferris-Editor(BM), Tom Freund(IBM), Mary Ann Hondo(IBM),  
2070 John Ibbotson(IBM), Lei Jin(BEA), Chris Kaler(Microsoft), David Langworthy-Editor(Microsoft),  
2071 Amelia Lewis(TIBCO Software), Rodney Limprecht(Microsoft), Steve Lucco(Microsoft), Don  
2072 Mullen(TIBCO Software), Anthony Nadalin(IBM), Mark Nottingham(BEA), David Orchard(BEA),  
2073 Jamie Roots(IBM), Shivajee Samdarshi(TIBCO Software), John Shewchuk(Microsoft), Tony  
2074 Storey(IBM).

2075 The following individuals have provided invaluable input into the initial contribution:

2076 Keith Ballinger(Microsoft), Stefan Batres(Microsoft), Rebecca Bergersen(Iona), Allen  
2077 Brown(Microsoft), Michael Conner(IBM), George Copeland(Microsoft), Francisco Curbera(IBM),  
2078 Paul Fremantle(IBM), Steve Graham(IBM), Pat Helland(Microsoft), Rick Hill(Microsoft), Scott  
2079 Hinkelman(IBM), Tim Holloway(IBM), Efim Hudis(Microsoft), David Ingham(Microsoft), Gopal  
2080 Kakivaya(Microsoft), Johannes Klein(Microsoft), Frank Leymann(IBM), Martin Nally(IBM), Peter  
2081 Niblett(IBM), Jeffrey Schlimmer(Microsoft), James Snell(IBM), Keith Stobie(Microsoft), Satish  
2082 Thatte(Microsoft), Stephen Todd(IBM), Sanjiva Weerawarana(IBM), Roger Wolter(Microsoft).

2083 The following individuals were members of the committee during the development of this specification:

2084 Abbie Barbir(Nortel), Charlton Barreto(Adobe), Stefan Batres(Microsoft), Hamid Ben  
2085 Malek(Fujitsu), Andreas Bjarlestam(Ericsson), Toufic Boubez(Layer 7), Doug Bunting(Sun), Lloyd  
2086 Burch(Novell), Steve Carter(Novell), Martin Chapman(Oracle), Dave Chappell(Sonic), Paul  
2087 Cotton(Microsoft), Glen Daniels(Sonic), Doug Davis(IBM), Blake Dournaee(Intel), Jacques  
2088 Durand(Fujitsu), Colleen Evans(Microsoft), Christopher Ferris(IBM), Paul Fremantle(WSO2),  
2089 Robert Freund(Hitachi), Peter Furniss(Erebor), Marc Goodner(Microsoft), Alastair  
2090 Green(Choreology), Mike Grogan(Sun), Ondrej Hrebicek(Microsoft), Kazunori Iwasa(Fujitsu),  
2091 Chamikara Jayalath(WSO2), Lei Jin(BEA), Ian Jones(BTplc), Anish Karmarkar(Oracle), Paul  
2092 Knight(Nortel), Dan Leshchiner(Tibco), Mark Little(JBoss), Lily Liu(webMethods), Matt  
2093 Lovett(IBM), Ashok Malhotra(Oracle), Jonathan Marsh(Microsoft), Daniel Millwood(IBM), Jeff  
2094 Mischkinsky(Oracle), Nilo Mitra(Ericsson), Peter Niblett(IBM), Duane Nickull(Adobe), Eisaku  
2095 Nishiyama(Hitachi), Dave Orchard(BEA), Chouthri Palanisamy(NEC), Sanjay Patil(SAP), Gilbert  
2096 Pilz(BEA), Martin Raepfle(SAP), Eric Rajkovic(Oracle), Stefan Rossmannith(SAP), Tom  
2097 Rutt(Fujitsu), Rich Salz(IBM), Shivajee Samdarshi(Tibco), Vladimir Videlov(SAP), Claus von  
2098 Riegen(SAP), Pete Wenzel(Sun), Steve Winkler(SAP), Ümit Yalçinalp(SAP), Nobuyuki  
2099 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 <a href="http://docs.oasis-open.org/wsrn/200510/">http://docs.oasis-open.org/wsrn/200510/</a> )
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



## 2100 **Appendix G. Notices**

2101 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that  
2102 might be claimed to pertain to the implementation or use of the technology described in this document or  
2103 the extent to which any license under such rights might or might not be available; neither does it represent  
2104 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to  
2105 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made  
2106 available for publication and any assurances of licenses to be made available, or the result of an attempt  
2107 made to obtain a general license or permission for the use of such proprietary rights by implementors or  
2108 users of this specification, can be obtained from the OASIS Executive Director.

2109 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or  
2110 other proprietary rights which may cover technology that may be required to implement this specification.  
2111 Please address the information to the OASIS Executive Director.

2112 Copyright (C) OASIS Open (2006). All Rights Reserved.

2113 This document and translations of it may be copied and furnished to others, and derivative works that  
2114 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and  
2115 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and  
2116 this paragraph are included on all such copies and derivative works. However, this document itself may  
2117 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as  
2118 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights  
2119 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it  
2120 into languages other than English.

2121 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors  
2122 or assigns.

2123 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
2124 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY  
2125 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR  
2126 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.