



# 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

42	1 Introduction.....	4
43	1.1 Notational Conventions.....	4
44	1.2 Namespace.....	5
45	1.3 Conformance.....	5
46	2 Reliable Messaging Model.....	6
47	2.1 Glossary.....	6
48	2.2 Protocol Preconditions.....	7
49	2.3 Protocol Invariants.....	8
50	2.4 Example Message Exchange.....	8
51	3 RM Protocol Elements.....	10
52	3.1 Considerations on the Use of Extensibility Points.....	10
53	3.2 Considerations on the Use of "Piggy-Backing".....	10
54	3.3 Composition with WS-Addressing.....	10
55	3.4 Sequence Creation.....	10
56	3.5 Closing A Sequence.....	15
57	3.6 Sequence Termination.....	16
58	3.7 Sequences.....	18
59	3.8 Request Acknowledgement.....	19
60	3.9 Sequence Acknowledgement.....	20
61	4 Faults.....	23
62	4.1 SequenceFault Element.....	24
63	4.2 Sequence Terminated.....	25
64	4.3 Unknown Sequence.....	25
65	4.4 Invalid Acknowledgement.....	26
66	4.5 Message Number Rollover.....	26
67	4.6 Create Sequence Refused.....	27
68	4.7 Sequence Closed.....	27
69	4.8 WSRM Required.....	28
70	5 Security Threats and Countermeasures.....	29
71	5.1 Threats and Countermeasures.....	29
72	5.1.1 Integrity Threats.....	29
73	5.1.1.1 Countermeasures.....	29
74	5.1.2 Resource Consumption Threats.....	30
75	5.1.2.1 Countermeasures.....	30
76	5.1.3 Sequence Spoofing Threats.....	30
77	5.1.3.1 Sequence Hijacking.....	30
78	5.1.3.2 Countermeasures.....	30

79	5.2 Security Solutions and Technologies.....	31
80	5.2.1 Transport Layer Security.....	31
81	5.2.1.1 Model.....	31
82	5.2.1.2 Countermeasure Implementation.....	32
83	5.2.2 SOAP Message Security.....	33
84	5.2.2.1 Model.....	33
85	5.2.2.2 Countermeasure Implementation.....	33
86	6 Securing Sequences.....	35
87	6.1 Securing Sequences Using WS-Security.....	35
88	6.2 Securing Sequences Using SSL/TLS.....	36
89	7 References.....	38
90	7.1 Normative.....	38
91	7.2 Non-Normative.....	39
92	Appendix A. Schema.....	41
93	Appendix B. WSDL.....	46
94	Appendix C. Message Examples.....	48
95	Appendix C.1 Create Sequence.....	48
96	Appendix C.2 Initial Transmission.....	48
97	Appendix C.3 First Acknowledgement.....	50
98	Appendix C.4 Retransmission.....	50
99	Appendix C.5 Termination.....	51
100	Appendix D. State Tables.....	53
101	Appendix E. Acknowledgments.....	57
102	Appendix F. Revision History.....	58
103	Appendix G. Notices.....	64

# 104 **1 Introduction**

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

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

## 114 **1.1 Notational Conventions**

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

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

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

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

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

## 142 1.2 Namespace

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

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

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

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

149 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>

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

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

## 153 1.3 Conformance

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

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

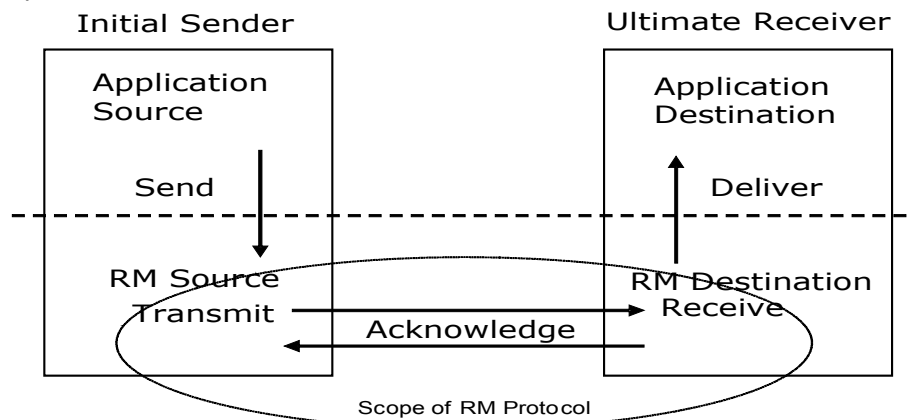
## 160 2 Reliable Messaging Model

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

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

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

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



186 Figure 1: Reliable Messaging Model

### 187 2.1 Glossary

188 The following definitions are used throughout this specification:

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

## 217 **2.2 Protocol Preconditions**

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

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

## 227 2.3 Protocol Invariants

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

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

## 239 2.4 Example Message Exchange

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

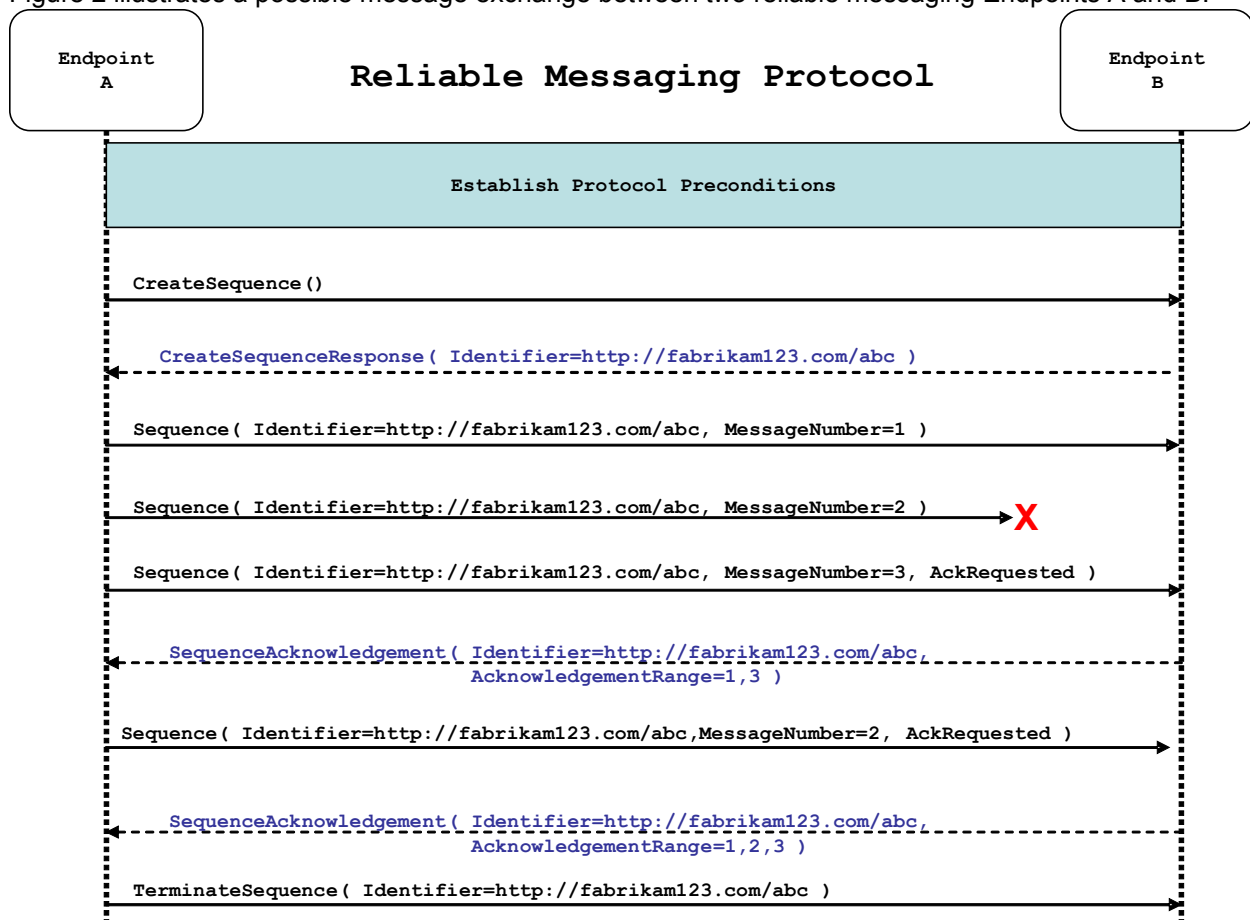


Figure 2: The WS-ReliableMessaging Protocol

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



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

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

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

## 280 **3 RM Protocol Elements**

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

### 283 **3.1 Considerations on the Use of Extensibility Points**

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

### 288 **3.2 Considerations on the Use of "Piggy-Backing"**

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

### 294 **3.3 Composition with WS-Addressing**

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

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

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

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

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

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

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

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

### 312 **3.4 Sequence Creation**

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

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

320 The following exemplar defines the `CreateSequence` syntax:

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

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

336 `/wsrm:CreateSequence`

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

341 `/wsrm:CreateSequence/wsrm:AcksTo`

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

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

351 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

359 `/wsrm:CreateSequence/wsrm:Offer`

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

362 `/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier`

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

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

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

368 /wsmr:CreateSequence/wsmr:Offer/wsmr:Endpoint

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

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

380 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

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

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

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

387 /wsmr:CreateSequence/wsmr:Offer/wsmr:IncompleteSequenceBehavior

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

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

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

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

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

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

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

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

404 /wsmr:CreateSequence/{any}

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

407 /wsmr:CreateSequence/@{any}

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

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

414 The following exemplar defines the `CreateSequenceResponse` syntax:

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

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

428 /wsmr:CreateSequenceResponse

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

432 /wsmr:CreateSequenceResponse/wsmr:Identifier

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

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

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

439 /wsmr:CreateSequenceResponse/wsmr:Expires

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

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

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

451 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`

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

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

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

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

462 `/wsrm:CreateSequenceResponse/wsrm:Accept`

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

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

468 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

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

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

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

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

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

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

483 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

### 489 3.5 Closing A Sequence

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

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

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

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

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

517 The following exemplar defines the `CloseSequence` syntax:

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

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

523 `/wsmr:CloseSequence`

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

527 `/wsmr:CloseSequence/wsmr:Identifier`

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

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

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

534 /wsmr:CloseSequence/{any}

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

537 /wsmr:CloseSequence@{any}

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

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

543 The following exemplar defines the `CloseSequenceResponse` syntax:

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

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

549 /wsmr:CloseSequenceResponse

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

553 /wsmr:CloseSequenceResponse/wsmr:Identifier

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

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

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

560 /wsmr:CloseSequenceResponse/{any}

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

563 /wsmr:CloseSequenceResponse@{any}

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

## 566 3.6 Sequence Termination

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



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

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

578 The following exemplar defines the TerminateSequence syntax:

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

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

584 /wsrm:TerminateSequence

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

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

595 /wsrm:TerminateSequence/wsrm:Identifier

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

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

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

602 /wsrm:TerminateSequence/{any}

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

605 /wsrm:TerminateSequence/@{any}

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

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

611 The following exemplar defines the TerminateSequenceResponse syntax:

```
612 <wsrm:TerminateSequenceResponse ...>
```

```
613 <wsm:Identifier ...> xs:anyURI </wsm:Identifier>
614 ...
615 </wsm:TerminateSequenceResponse>
```

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

617 `/wsm:TerminateSequenceResponse`

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

622 `/wsm:TerminateSequenceResponse/wsm:Identifier`

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

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

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

630 `/wsm:TerminateSequenceResponse/{any}`

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

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

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

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

### 639 3.7 Sequences

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

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

647 A following exemplar defines its syntax:

```
648 <wsm:Sequence ...>
649   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>
650   <wsm:MessageNumber> wsm:MessageNumberType </wsm:MessageNumber>
651   ...
652 </wsm:Sequence>
```

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

654 /wsmr:Sequence

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

661 /wsmr:Sequence/wsmr:Identifier

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

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

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

668 /wsmr:Sequence/wsmr:MessageNumber

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

673 /wsmr:Sequence/{any}

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

676 /wsmr:Sequence/@{any}

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

679 The following example illustrates a Sequence header block.

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

### 684 3.8 Request Acknowledgement

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

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

695 Destination return a `AcknowledgementRange` or `None` element instead of a `Nack` element (see Section  
696 3.9).

697 The following exemplar defines its syntax:

```
698 <wsm:AckRequested ...>  
699   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>  
700   ...  
701 </wsm:AckRequested>
```

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

703 `/wsm:AckRequested`

704 This element requests an Acknowledgement for the identified Sequence.

705 `/wsm:AckRequested/wsm:Identifier`

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

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

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

712 `/wsm:AckRequested/{any}`

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

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

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

### 718 **3.9 Sequence Acknowledgement**

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

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

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

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

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

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

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

726 affected.

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

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

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

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

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

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

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

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

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

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

739 The following exemplar defines its syntax:

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

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

752 `/wsm:SequenceAcknowledgement`

753 This element contains the Sequence Acknowledgement information.

754 `/wsm:SequenceAcknowledgement/wsm:Identifier`

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

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

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

763 `/wsm:SequenceAcknowledgement/wsm:AcknowledgementRange`

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

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

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

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

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

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

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

778 `/wsm:SequenceAcknowledgement/wsm:None`

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

783 `/wsrm:SequenceAcknowledgement/wsrm:Final`

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

790 `/wsrm:SequenceAcknowledgement/wsrm:Nack`

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

800 `/wsrm:SequenceAcknowledgement/{any}`

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

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

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

806 The following examples illustrate `SequenceAcknowledgement` elements:

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

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

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

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

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

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

823  
824

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

## 825 4 Faults

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

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

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

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

839 The definitions of faults use the following properties:

840 [Code] The fault code.

841 [Subcode] The fault subcode.

842 [Reason] The English language reason element.

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

846 Entities that generate WS-ReliableMessaging faults MUST set the [Code] property to either "Sender" or  
847 "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

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

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



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

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

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

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

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

## 902 4.1 SequenceFault Element

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

908 The following exemplar defines its syntax:

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

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

915 `/wsrm:SequenceFault`

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

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

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

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

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

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

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

## 932 4.2 Sequence Terminated

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

935 Properties:

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

940 `<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.

## 941 4.3 Unknown Sequence

942 Properties:

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

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

946 [Detail]

947 `<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.

#### 948 4.4 Invalid Acknowledgement

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

951 [Code] Sender

952 [Subcode] wsrn:InvalidAcknowledgement

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

954 [Detail]

955 `<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.

#### 956 4.5 Message Number Rollover

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

958 Properties:

959 [Code] Sender

960 [Subcode] wsrn:MessageNumberRollover

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

962 [Detail]

```

963 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
964 <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.

## 965 4.6 Create Sequence Refused

966 Properties:

967 [Code] Sender or Receiver

968 [Subcode] wsrm:CreateSequenceRefused

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

970 [Detail]

```

971 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.

## 972 4.7 Sequence Closed

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

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

976 Properties:

977 [Code] Sender

978 [Subcode] wsrm:SequenceClosed

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

980 [Detail]

981 `<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.

## 982 **4.8 WSRM Required**

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

985 Properties:

986 [Code] Sender

987 [Subcode] wsrm:WSRMRequired

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

989 [Detail]

990 `xs:any`

## 991 **5 Security Threats and Countermeasures**

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

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

998 There are many other security concerns that one may need to consider when implementing or using this  
999 protocol. The material below should not be considered as a "check list". Implementers and users of this  
1000 protocol are urged to perform a security analysis to determine their particular threat profile and the  
1001 appropriate responses to those threats.

1002 Implementers are also advised that there is a core tension between security and reliable messaging that  
1003 can be problematic if not addressed by implementations; one aspect of security is to prevent message  
1004 replay but one of the invariants of this protocol is to resend messages until they are acknowledged.  
1005 Consequently, if the security sub-system processes a message but a failure occurs before the reliable  
1006 messaging sub-system Receives that message, then it is possible (and likely) that the security sub-system  
1007 will treat subsequent copies as replays and discard them. At the same time, the reliable messaging sub-  
1008 system will likely continue to expect and even solicit the missing message(s). Care should be taken to  
1009 avoid and prevent this condition.

### 1010 **5.1 Threats and Countermeasures**

1011 The primary security requirement of this protocol is to protect the specified semantics and protocol  
1012 invariants against various threats. The following sections describe several threats to the integrity and  
1013 operation of this protocol and provide some general outlines of countermeasures to those threats.  
1014 Implementers and users of this protocol should keep in mind that all threats are not necessarily applicable  
1015 to all operational contexts.

#### 1016 **5.1.1 Integrity Threats**

1017 In general, any mechanism which allows an attacker to alter the information in a Sequence Traffic  
1018 Message, Sequence Lifecycle Message, Acknowledgement Messages, Acknowledgement Request, or  
1019 Sequence-related fault, or which allows an attacker to alter the correlation of a RM Protocol Header Block  
1020 to its intended message represents a threat to the WS-RM protocol.

1021 For example, if an attacker is able to swap `Sequence` headers on messages in transit between the RM  
1022 Source and RM Destination then they have undermined the implementation's ability to guarantee the first  
1023 invariant described in Section 2.3. The result is that there is no way of guaranteeing that messages will be  
1024 Delivered to the Application Destination in the same order that they were sent by the Application Source.

##### 1025 **5.1.1.1 Countermeasures**

1026 Integrity threats are generally countered via the use of digital signatures some level of the communication  
1027 protocol stack. Note that, in order to counter header swapping attacks, the signature SHOULD include  
1028 both the SOAP body and any relevant SOAP headers (e.g. `Sequence` header). Because some headers  
1029 (`AckRequested`, `SequenceAcknowledgement`) are independent of the body of the SOAP message in which  
1030 they occur, implementations MUST allow for signatures that cover only these headers.

## 1031 **5.1.2 Resource Consumption Threats**

1032 The creation of a Sequence with an RM Destination consumes various resources on the systems used to  
1033 implement that RM Destination. These resources can include network connections, database tables,  
1034 message queues, etc. This behavior can be exploited to conduct denial of service attacks against an RM  
1035 Destination. For example, a simple attack is to repeatedly send `CreateSequence` messages to an RM  
1036 Destination. Another attack is to create a Sequence for a service that is known to require in-order  
1037 message Delivery and use this Sequence to send a stream of very large messages to that service,  
1038 making sure to omit message number “1” from that stream.

### 1039 **5.1.2.1 Countermeasures**

1040 There are a number of countermeasures against the described resource consumption threats. The  
1041 technique advocated by this specification is for the RM Destination to restrict the ability to create a  
1042 Sequence to a specific set of entities/principals. This reduces the number of potential attackers and, in  
1043 some cases, allows the identity of any attackers to be determined.

1044 The ability to restrict Sequence creation depends, in turn, upon the RM Destination's ability identify and  
1045 authenticate the RM Source that issued the `CreateSequence` message.

## 1046 **5.1.3 Sequence Spoofing Threats**

1047 Sequence spoofing is a class of threats in which the attacker uses knowledge of the `Identifier` for a  
1048 particular Sequence to forge Sequence Lifecycle or Traffic Messages. For example the attacker creates a  
1049 fake `TerminateSequence` message that references the target Sequence and sends this message to the  
1050 appropriate RM Destination. Some sequence spoofing attacks also require up-to-date knowledge of the  
1051 current `MessageNumber` for their target Sequence.

1052 In general any Sequence Lifecycle Message, RM Protocol Header Block, or sequence-correlated SOAP  
1053 fault (e.g. `InvalidAcknowledgement`) can be used by someone with knowledge of the Sequence identifier  
1054 to attack the Sequence. These attacks are “two-way” in that an attacker may choose to target the RM  
1055 Source by, for example, inserting a fake `SequenceAcknowledgement` header into a message that it sends  
1056 to the `AcksTo` EPR of an RM Source.

### 1057 **5.1.3.1 Sequence Hijacking**

1058 Sequence hijacking is a specific case of a sequence spoofing attack. The attacker attempts to inject  
1059 Sequence Traffic Messages into an existing Sequence by inserting fake `Sequence` headers into those  
1060 messages.

1061 Note that “sequence hijacking” should not be equated with “security session hijacking”. Although a  
1062 Sequence may be bound to some form of a security session in order to counter the threats described in  
1063 this section, applications MUST NOT rely on WS-RM-related information to make determinations about  
1064 the identity of the entity that created a message; applications SHOULD rely only upon information that is  
1065 established by the security infrastructure to make such determinations. Failure to observe this rule  
1066 creates, among other problems, a situation in which the absence of WS-RM may deprive an application of  
1067 the ability to authenticate its peers even though the necessary security processing has taken place.

### 1068 **5.1.3.2 Countermeasures**

1069 There are a number of countermeasures against sequence spoofing threats. The technique advocated by  
1070 this specification is to consider the Sequence to be a shared resource that is jointly owned by the RM

1071 Source that initiated its creation (i.e. that sent the `CreateSequence` message) and the RM Destination that  
1072 serves as its terminus (i.e. that sent the `CreateSequenceResponse` message). To counter sequence  
1073 spoofing attempts the RM Destination SHOULD ensure that every message or fault that it Receives that  
1074 refers to a particular Sequence originated from the RM Source that jointly owns the referenced Sequence.  
1075 For its part the RM Source SHOULD ensure that every message or fault that it Receives that refers to a  
1076 particular Sequence originated from the RM Destination that jointly owns the referenced Sequence.

1077 For the RM Destination to be able to identify its sequence peer it MUST be able to identify and  
1078 authenticate the entity that sent the `CreateSequence` message. Similarly for the RM Source to identify its  
1079 sequence peer it MUST be able to identify and authenticate the entity that sent the  
1080 `CreateSequenceResponse` message. For either the RM Destination or the RM Source to determine if a  
1081 message was sent by its sequence peer it MUST be able to identify and authenticate the initiator of that  
1082 message and, if necessary, correlate this identity with the sequence peer identity established at sequence  
1083 creation time.

## 1084 **5.2 Security Solutions and Technologies**

1085 The security threats described in the previous sections are neither new nor unique. The solutions that  
1086 have been developed to secure other SOAP-based protocols can be used to secure WS-RM as well. This  
1087 section maps the facilities provided by common web services security solutions against countermeasures  
1088 described in the previous sections.

1089 Before continuing this discussion, however, some examination of the underlying requirements of the  
1090 previously described countermeasures is necessary. Specifically it should be noted that the technique  
1091 described in Section 5.1.2.1 has two components. Firstly, the RM Destination identifies and authenticates  
1092 the issuer of a `CreateSequence` message. Secondly, the RM Destination performs an authorization check  
1093 against this authenticated identity and determines if the RM Source is permitted to create Sequences with  
1094 the RM Destination. Since the facilities for performing this authorization check (runtime infrastructure,  
1095 policy frameworks, etc.) lie completely within the domain of individual implementations, any discussion of  
1096 such facilities is considered to be beyond the scope of this specification.

### 1097 **5.2.1 Transport Layer Security**

1098 This section describes how the facilities provided by SSL/TLS [RFC 4346] can be used to implement the  
1099 countermeasures described in the previous sections. The use of SSL/TLS is subject to the constraints  
1100 defined in Section 4 of the Basic Security Profile 1.0 [BSP 1.0].

1101 The description provided here is general in nature and is not intended to serve as a complete definition on  
1102 the use of SSL/TLS to protect WS-RM. In order to interoperate implementations need to agree on the  
1103 choice of features as well as the manner in which they will be used. The mechanisms described in the  
1104 Web Services Security Policy Language [SecurityPolicy] MAY be used by services to describe the  
1105 requirements and constraints of the use of SSL/TLS.

#### 1106 **5.2.1.1 Model**

1107 The basic model for using SSL/TLS is as follows:

- 1108 1. The RM Source establishes an SSL/TLS session with the RM Destination.
- 1109 2. The RM Source uses this SSL/TLS session to send a `CreateSequence` message to the RM  
1110 Destination.



- 1111 3. The RM Destination establishes an SSL/TLS session with the RM Source and sends an  
1112 asynchronous `CreateSequenceResponse` using this session. Alternately it may respond with a  
1113 synchronous `CreateSequenceResponse` using the session established in (1).
- 1114 4. For the lifetime of the Sequence the RM Source uses the SSL/TLS session from (1) to Transmit  
1115 any and all messages or faults that refer to that Sequence.
- 1116 5. For the lifetime of the Sequence the RM Destination either uses the SSL/TLS session established  
1117 in (3) to Transmit any and all messages or faults that refer to that Sequence or, for synchronous  
1118 exchanges, the RM Destination uses the SSL/TLS session established in (1).

### 1119 5.2.1.2 Countermeasure Implementation

1120 Used in its simplest fashion (without relying upon any authentication mechanisms), SSL/TLS provides the  
1121 necessary integrity qualities to counter the threats described in Section 5.1.1. Note, however, that the  
1122 nature of SSL/TLS limits the scope of this integrity protection to a single transport level session. If  
1123 SSL/TLS is the only mechanism used to provide integrity, any intermediaries between the RM Source and  
1124 the RM Destination MUST be trusted to preserve the integrity of the messages that flow through them.

1125 As noted, the technique described in Sections 5.1.2.1 involves the use of authentication. This specification  
1126 advocates either of two mechanisms for authenticating entities using SSL/TLS. In both of these methods  
1127 the SSL/TLS server (the party accepting the SSL/TLS connection) authenticates itself to the SSL/TLS  
1128 client using an X.509 certificate that is exchanged during the SSL/TLS handshake.

- 1129 • **HTTP Basic Authentication:** This method of authentication presupposes that a SOAP/HTTP  
1130 binding is being used as part of the protocol stack beneath WS-RM. Subsequent to the  
1131 establishment of the SSL/TLS session, the sending party authenticates itself to the receiving party  
1132 using HTTP Basic Authentication [RFC 2617]. For example, a RM Source might authenticate itself  
1133 to a RM Destination (e.g. when transmitting a Sequence Traffic Message) using BasicAuth.  
1134 Similarly the RM Destination might authenticate itself to the RM Source (e.g. when sending an  
1135 Acknowledgement) using BasicAuth.
- 1136 • **SSL/TLS Client Authentication:** In this method of authentication, the party initiating the  
1137 connection authenticates itself to the party accepting the connection using an X.509 certificate  
1138 that is exchanged during the SSL/TLS handshake.

1139 To implement the countermeasures described in section 5.1.2.1 the RM Source must authenticate itself  
1140 using one the above mechanisms. The authenticated identity can then be used to determine if the RM  
1141 Source is authorized to create a Sequence with the RM Destination.

1142 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring  
1143 an RM node's Sequence peer to be equivalent to their SSL/TLS session peer. This allows the  
1144 authorization decisions described in section 5.1.3.2 to be based on SSL/TLS session identity rather than  
1145 on authentication information. For example, an RM Destination can determine that a Sequence Traffic  
1146 Message rightfully belongs to its referenced Sequence if that message arrived over the same SSL/TLS  
1147 session that was used to carry the `CreateSequence` message for that Sequence. Note that requiring a  
1148 one-to-one relationship between SSL/TLS session peer and Sequence peer constrains the lifetime of a  
1149 SSL/TLS-protected Sequence to be less than or equal to the lifetime of the SSL/TLS session that is used  
1150 to protect that Sequence.

1151 This specification does not preclude the use of other methods of using SSL/TLS to implement the  
1152 countermeasures (such as associating specific authentication information with a Sequence) although such  
1153 methods are not covered by this document.

1154 Issues specific to the life-cycle management of SSL/TLS sessions (such as the resumption of a SSL/TLS  
1155 session) are outside the scope of this specification.

## 1156 **5.2.2 SOAP Message Security**

1157 The mechanisms described in WS-Security may be used in various ways to implement the  
1158 countermeasures described in the previous sections. This specification advocates using the protocol  
1159 described by WS-SecureConversation [[SecureConversation](#)] (optionally in conjunction with WS-Trust  
1160 [[Trust](#)]) as a mechanism for protecting Sequences. The use of WS-Security (as an underlying component  
1161 of WS-SecureConversation) is subject to the constraints defined in the Basic Security Profile 1.0.

1162 The description provided here is general in nature and is not intended to serve as a complete definition on  
1163 the use of WS-SecureConversation/WS-Trust to protect WS-RM. In order to interoperate implementations  
1164 need to agree on the choice of features as well as the manner in which they will be used. The  
1165 mechanisms described in the Web Services Security Policy Language MAY be used by services to  
1166 describe the requirements and constraints of the use of WS-SecureConversation.

### 1167 **5.2.2.1 Model**

1168 The basic model for using WS-SecureConversation is as follows:

- 1169 1. The RM Source and the RM Destination create a WS-SecureConversation security context. This  
1170 may involve the participation of third parties such as a security token service. The tokens  
1171 exchanged may contain authentication claims (e.g. X.509 certificates or Kerberos service tickets).
- 1172 2. During the `CreateSequence` exchange, the RM Source SHOULD explicitly identify the security  
1173 context that will be used to protect the Sequence. This is done so that, in cases where the  
1174 `CreateSequence` message is signed by more than one security context, the RM Source can  
1175 indicate which security context should be used to protect the newly created Sequence.
- 1176 3. For the lifetime of the Sequence the RM Source and the RM Destination use the session key(s)  
1177 associated with the security context to sign (as defined by WS-Security) at least the body and any  
1178 relevant WS-RM-defined headers of any and all messages or faults that refer to that Sequence.

### 1179 **5.2.2.2 Countermeasure Implementation**

1180 Without relying upon any authentication information, the per-message signatures provide the necessary  
1181 integrity qualities to counter the threats described in Section 5.1.1.

1182 To implement the countermeasures described in section 5.1.2.1 some mutually agreed upon form of  
1183 authentication claims must be provided by the RM Source to the RM Destination during the establishment  
1184 of the Security Context. These claims can then be used to determine if the RM Source is authorized to  
1185 create a Sequence with the RM Destination.

1186 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring  
1187 an RM node's Sequence peer to be equivalent to their security context session peer. This allows the  
1188 authorization decisions described in section 5.1.3.2 to be based on the identity of the message's security  
1189 context rather than on any authentication claims that may have been established during security context  
1190 initiation. Note that other methods of using WS-SecureConversation to implement the countermeasures  
1191 (such as associating specific authentication claims to a Sequence) are possible but not covered by this  
1192 document.

1193 As with transport security, the requisite equivalence of a security context peer and with a Sequence peer  
1194 limits the lifetime of a Sequence to the lifetime of the protecting security context. Unlike transport security,

1195 the association between a Sequence and its protecting security context cannot always be established  
1196 implicitly at Sequence creation time. This is due to the fact that the `CreateSequence` and  
1197 `CreateSequenceResponse` messages may be signed by more than one security context.

1198 Issues specific to the life-cycle management of WS-SecureConversation security contexts (such as  
1199 amending or renewing contexts) are outside the scope of this specification.

## 1200 6 Securing Sequences

1201 As noted in Section 5, the RM Source and RM Destination should be able to protect their shared  
1202 Sequences against the threat of Sequence Spoofing attacks. There are a number of OPTIONAL means of  
1203 achieving this objective depending upon the underlying security infrastructure.

### 1204 6.1 Securing Sequences Using WS-Security

1205 One mechanism for protecting a Sequence is to include a security token using a  
1206 `wsse:SecurityTokenReference` element from WS-Security (see section 9 in WS-  
1207 SecureConversation) in the `CreateSequence` element. This establishes an association between the  
1208 created (and, if present, offered) Sequence(s) and the referenced security token, such that the RM Source  
1209 and Destination MUST use the security token as the basis for authorization of all subsequent interactions  
1210 related to the Sequence(s). The `wsse:SecurityTokenReference` explicitly identifies the token as  
1211 there may be more than one token on a `CreateSequence` message or inferred from the communication  
1212 context (e.g. transport protection).

1213 It is RECOMMENDED that a message independent referencing mechanism be used to identify the token,  
1214 if the token being referenced supports such mechanism.

1215 The following exemplar defines the `CreateSequence` syntax when extended to include a  
1216 `wsse:SecurityTokenReference`:

```
1217 <wsrm:CreateSequence ...>  
1218   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>  
1219   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1220   <wsrm:Offer ...>  
1221     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
1222     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>  
1223     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1224     <wsrm:IncompleteSequenceBehavior>  
1225       wsrml:IncompleteSequenceBehaviorType  
1226     </wsrm:IncompleteSequenceBehavior> ?  
1227     ...  
1228   </wsrm:Offer> ?  
1229   ...  
1230   <wsse:SecurityTokenReference>  
1231     ...  
1232   </wsse:SecurityTokenReference> ?  
1233   ...  
1234 </wsrm:CreateSequence>
```

1235 The following describes the content model of the additional `CreateSequence` elements.

1236 `/wsrm:CreateSequence/wsse:SecurityTokenReference`

1237 This element uses the extensibility mechanism defined for the `CreateSequence` element (defined in  
1238 section 3.4) to communicate an explicit reference to the security token, using a  
1239 `wsse:SecurityTokenReference` as documented in WS-Security, that the RM Source and Destination  
1240 MUST use to authorize messages for the created (and, if present, the offered) Sequence(s). All  
1241 subsequent messages related to the created (and, if present, the offered) Sequence(s) MUST  
1242 demonstrate proof-of-possession of the secret associated with the token (e.g., by using or deriving from a  
1243 private or secret key).

1244 When a RM Source transmits a `CreateSequence` that has been extended to include a  
1245 `wsse:SecurityTokenReference` it SHOULD ensure that the RM Destination both understands and  
1246 will conform to the requirements listed above. In order to achieve this, the RM Source SHOULD include

1247 the `UsesSequenceSTR` element as a SOAP header block within the `CreateSequence` message. This  
1248 element MUST include a `soap:mustUnderstand` attribute with a value of 'true'. Thus the RM Source  
1249 can be assured that a RM Destination that responds with a `CreateSequenceResponse` understands  
1250 and conforms with the requirements listed above. Note that an RM Destination understanding this header  
1251 does not mean that it has processed and understood any WS-Security headers, the fault behavior defined  
1252 in WS-Security still applies.

1253 The following exemplar defines the `UsesSequenceSTR` syntax:

```
1254 <wsm:UsesSequenceSTR ... />
```

1255 The following describes the content model of the `UsesSequenceSTR` header block.

1256 `/wsm:UsesSequenceSTR`

1257 This element SHOULD be included as a SOAP header block in `CreateSequence` messages that use the  
1258 extensibility mechanism described above in this section. The `soap:mustUnderstand` attribute value  
1259 MUST be 'true'. The receiving RM Destination MUST understand and correctly implement the extension  
1260 described above or else generate a `soap:MustUnderstand` fault, thus aborting the requested  
1261 Sequence creation.

1262 The following is an example of a `CreateSequence` message using the

1263 `wsse:SecurityTokenReference` extension and the `UsesSequenceSTR` header block:

```
1264 <soap:Envelope ...>  
1265   <soap:Header>  
1266     ...  
1267     <wsm:UsesSequenceSTR soap:mustUnderstand='true' />  
1268     ...  
1269   </soap:Header>  
1270   <soap:Body>  
1271     <wsm:CreateSequence>  
1272       <wsm:AcksTo>  
1273         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>  
1274       </wsm:AcksTo>  
1275       <wsse:SecurityTokenReference>  
1276         ...  
1277       </wsse:SecurityTokenReference>  
1278     </wsm:CreateSequence>  
1279   </soap:Body>  
1280 </soap:Envelope>
```

## 1281 6.2 Securing Sequences Using SSL/TLS

1282 One mechanism for protecting a Sequence is to bind the Sequence to the underlying SSL/TLS session(s).  
1283 The RM Source indicates to the RM Destination that a Sequence is to be bound to the underlying  
1284 SSL/TLS session(s) via the `UsesSequenceSSL` header block. If the RM Source wishes to bind a  
1285 Sequence to the underlying SSL/TLS sessions(s) it MUST include the `UsesSequenceSSL` element as a  
1286 SOAP header block within the `CreateSequence` message.

1287 The following exemplar defines the `UsesSequenceSSL` syntax:

```
1288 <wsm:UsesSequenceSSL soap:mustUnderstand="true" ... />
```

1289 The following describes the content model of the `UsesSequenceSSL` header block.

1290 `/wsm:UsesSequenceSSL`

1291 The RM Source MAY include this element as a SOAP header block of a `CreateSequence` message to  
1292 indicate to the RM Destination that the resulting Sequence is to be bound to the SSL/TLS session that was

1293 used to carry the `CreateSequence` message. If included, the RM Source MUST mark this header with a  
1294 `soap:mustUnderstand` attribute with a value of 'true'. The receiving RM Destination MUST understand  
1295 and correctly implement the functionality described in Section 5.2.1 or else generate a  
1296 `soap:MustUnderstand` fault, thus aborting the requested Sequence creation.

1297 Note that the use inclusion of the above header by the RM Source implies that all Sequence-related  
1298 information (Sequence Lifecycle or Acknowledgment messages or Sequence-related faults) flowing from  
1299 the RM Destination to the RM Source will be bound to the SSL/TLS session that is used to carry the  
1300 `CreateSequenceResponse` message.

## 1301 **7 References**

### 1302 **7.1 Normative**

#### 1303 **[KEYWORDS]**

1304 S. Bradner, "Key words for use in RFCs to Indicate Requirement Levels," RFC 2119, Harvard University,  
1305 March 1997

1306 <http://www.ietf.org/rfc/rfc2119.txt>

#### 1307 **[WS-RM Policy]**

1308 OASIS WS-RX Technical Committee Draft, "Web Services ReliableMessaging Policy Assertion( WS-RM  
1309 Policy)" October 2006

1310 <http://docs.oasis-open.org/ws-rx/wsrmp/200608/wsrmp-1.1-spec-wd-11.pdf>

#### 1311 **[SOAP 1.1]**

1312 W3C Note, "SOAP: Simple Object Access Protocol 1.1," 08 May 2000.

1313 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>

#### 1314 **[SOAP 1.2]**

1315 W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging Framework" June 2003.

1316 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

#### 1317 **[URI]**

1318 T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 3986,  
1319 MIT/LCS, U.C. Irvine, Xerox Corporation, January 2005.

1320 <http://ietf.org/rfc/rfc3986>

#### 1321 **[UUID]**

1322 P. Leach, M. Mealling, R. Salz, "A Universally Unique IDentifier (UUID) URN Namespace," RFC 4122,  
1323 Microsoft, Refactored Networks - LLC, DataPower Technology Inc, July 2005

1324 <http://www.ietf.org/rfc/rfc4122.txt>

#### 1325 **[XML]**

1326 W3C Recommendation, "Extensible Markup Language (XML) 1.0 (Fourth Edition)", September 2006.

1327 <http://www.w3.org/TR/REC-xml/>

#### 1328 **[XML-ns]**

1329 W3C Recommendation, "Namespaces in XML," 14 January 1999.

1330 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>

#### 1331 **[XML-Schema Part1]**

1332 W3C Recommendation, "XML Schema Part 1: Structures," October 2004.

1333 <http://www.w3.org/TR/xmlschema-1/>

1334 **[XML-Schema Part2]**

1335 W3C Recommendation, "XML Schema Part 2: Datatypes," October 2004.

1336 <http://www.w3.org/TR/xmlschema-2/>

1337 **[XPath 1.0]**

1338 W3C Recommendation, "XML Path Language (XPath) Version 1.0," 16 November 1999.

1339 <http://www.w3.org/TR/xpath>

1340 **[WSDL 1.1]**

1341 W3C Note, "Web Services Description Language (WSDL 1.1)," 15 March 2001.

1342 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

1343 **[WS-Addressing]**

1344 W3C Recommendation, "Web Services Addressing 1.0 - Core", May 2006.

1345 <http://www.w3.org/TR/2006/REC-ws-addr-core-20060509/>

1346 W3C Recommendation, "Web Services Addressing 1.0 – SOAP Binding", May 2006.

1347 <http://www.w3.org/TR/2006/REC-ws-addr-soap-20060509/>

1348 **7.2 Non-Normative**

1349 **[BSP 1.0]**

1350 WS-I Working Group Draft. "Basic Security Profile Version 1.0," August 2006

1351 <http://www.ws-i.org/Profiles/BasicSecurityProfile-1.0.html>

1352 **[RDDL 2.0]**

1353 Jonathan Borden, Tim Bray, eds. "Resource Directory Description Language (RDDL) 2.0," January 2004

1354 <http://www.openhealth.org/RDDL/20040118/rddl-20040118.html>

1355 **[RFC 2617]**

1356 J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Loutonen, L. Stewart, "HTTP  
1357 Authentication: Basic and Digest Access Authentication," June 1999.

1358 <http://www.ietf.org/rfc/rfc2617.txt>

1359 **[RFC 4346]**

1360 T. Dierks, E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.1," April 2006.

1361 <http://www.ietf.org/rfc/rfc4346.txt>

1362 **[WS-Policy]**

1363 W3C Member Submission, "Web Services Policy Framework (WS-Policy)," April 2006.

1364 <http://www.w3.org/Submission/2006/SUBM-WS-Policy-20060425/>

1365 **[WS-PolicyAttachment]**

1366 W3C Member Submission, "Web Services Policy Attachment (WS-PolicyAttachment)," April 2006.

1367 [http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-  
1368 20060425/](http://www.w3.org/Submission/2006/SUBM-WS-PolicyAttachment-20060425/)



1369 **[WS-Security]**

1370 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:  
1371 SOAP Message Security 1.0 (WS-Security 2004)", OASIS Standard 200401, March 2004.

1372 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

1373 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:  
1374 SOAP Message Security 1.1 (WS-Security 2004)", OASIS Standard 200602, February 2006.

1375 <http://www.oasis-open.org/committees/download.php/16790/wss-v1.1-spec-os-SOAPMessageSecurity.pdf>

1376 **[RTTM]**

1377 V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, May  
1378 1992.

1379 <http://www.rfc-editor.org/rfc/rfc1323.txt>

1380 **[SecurityPolicy]**

1381 G. Della-Libra, et. al. "Web Services Security Policy Language (WS-SecurityPolicy)", July 2005

1382 <http://specs.xmlsoap.org/ws/2005/07/securitypolicy/ws-securitypolicy.pdf>

1383 **[SecureConversation]**

1384 S. Anderson, et al, "Web Services Secure Conversation Language (WS-SecureConversation)," February  
1385 2005.

1386 <http://schemas.xmlsoap.org/ws/2004/04/sc/>

1387 **[Trust]**

1388 S. Anderson, et al, "Web Services Trust Language (WS-Trust)," February 2005.

1389 <http://schemas.xmlsoap.org/ws/2005/02/trust>

## 1390 Appendix A. Schema

1391 The normative schema that is defined for WS-ReliableMessaging using [XML-Schema Part1] and [XML-  
1392 Schema Part2] is located at:

1393 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-200608.xsd>

1394 The following copy is provided for reference.

```
1395 <?xml version="1.0" encoding="UTF-8"?>
1396 <!--
1397 OASIS takes no position regarding the validity or scope of any intellectual
1398 property or other rights that might be claimed to pertain to the
1399 implementation or use of the technology described in this document or the
1400 extent to which any license under such rights might or might not be available;
1401 neither does it represent that it has made any effort to identify any such
1402 rights. Information on OASIS's procedures with respect to rights in OASIS
1403 specifications can be found at the OASIS website. Copies of claims of rights
1404 made available for publication and any assurances of licenses to be made
1405 available, or the result of an attempt made to obtain a general license or
1406 permission for the use of such proprietary rights by implementors or users of
1407 this specification, can be obtained from the OASIS Executive Director.
1408 OASIS invites any interested party to bring to its attention any copyrights,
1409 patents or patent applications, or other proprietary rights which may cover
1410 technology that may be required to implement this specification. Please
1411 address the information to the OASIS Executive Director.
1412 Copyright © OASIS Open 2002-2006. All Rights Reserved.
1413 This document and translations of it may be copied and furnished to others,
1414 and derivative works that comment on or otherwise explain it or assist in its
1415 implementation may be prepared, copied, published and distributed, in whole or
1416 in part, without restriction of any kind, provided that the above copyright
1417 notice and this paragraph are included on all such copies and derivative
1418 works. However, this document itself does not be modified in any way, such as
1419 by removing the copyright notice or references to OASIS, except as needed for
1420 the purpose of developing OASIS specifications, in which case the procedures
1421 for copyrights defined in the OASIS Intellectual Property Rights document must
1422 be followed, or as required to translate it into languages other than English.
1423 The limited permissions granted above are perpetual and will not be revoked by
1424 OASIS or its successors or assigns.
1425 This document and the information contained herein is provided on an "AS IS"
1426 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1427 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1428 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1429 FOR A PARTICULAR PURPOSE.
1430 -->
1431 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
1432 xmlns:wsa="http://www.w3.org/2005/08/addressing"
1433 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1434 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1435 elementFormDefault="qualified" attributeFormDefault="unqualified">
1436   <xs:import namespace="http://www.w3.org/2005/08/addressing"
1437   schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd"/>
1438   <!-- Protocol Elements -->
1439   <xs:complexType name="SequenceType">
1440     <xs:sequence>
1441       <xs:element ref="wsrm:Identifier"/>
1442       <xs:element name="MessageNumber" type="wsrm:MessageNumberType"/>
1443       <xs:any namespace="##other" processContents="lax" minOccurs="0"
1444 maxOccurs="unbounded"/>
1445     </xs:sequence>
```

```

1446     <xs:anyAttribute namespace="##other" processContents="lax"/>
1447 </xs:complexType>
1448 <xs:element name="Sequence" type="wsrm:SequenceType"/>
1449 <xs:element name="SequenceAcknowledgement">
1450   <xs:complexType>
1451     <xs:sequence>
1452       <xs:element ref="wsrm:Identifier"/>
1453       <xs:choice>
1454         <xs:sequence>
1455           <xs:choice>
1456             <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
1457               <xs:complexType>
1458                 <xs:sequence/>
1459                 <xs:attribute name="Upper" type="xs:unsignedLong"
1460 use="required"/>
1461                 <xs:attribute name="Lower" type="xs:unsignedLong"
1462 use="required"/>
1463               <xs:anyAttribute namespace="##other" processContents="lax"/>
1464             </xs:complexType>
1465           </xs:element>
1466           <xs:element name="None">
1467             <xs:complexType>
1468               <xs:sequence/>
1469             </xs:complexType>
1470           </xs:element>
1471         </xs:choice>
1472         <xs:element name="Final" minOccurs="0">
1473           <xs:complexType>
1474             <xs:sequence/>
1475           </xs:complexType>
1476         </xs:element>
1477       </xs:sequence>
1478       <xs:element name="Nack" type="xs:unsignedLong"
1479 maxOccurs="unbounded"/>
1480     </xs:choice>
1481     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1482 maxOccurs="unbounded"/>
1483   </xs:sequence>
1484   <xs:anyAttribute namespace="##other" processContents="lax"/>
1485 </xs:complexType>
1486 </xs:element>
1487 <xs:complexType name="AckRequestedType">
1488   <xs:sequence>
1489     <xs:element ref="wsrm:Identifier"/>
1490     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1491 maxOccurs="unbounded"/>
1492   </xs:sequence>
1493   <xs:anyAttribute namespace="##other" processContents="lax"/>
1494 </xs:complexType>
1495 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1496 <xs:element name="Identifier">
1497   <xs:complexType>
1498     <xs:annotation>
1499       <xs:documentation>
1500         This type is for elements whose [children] is an anyURI and can have
1501 arbitrary attributes.
1502       </xs:documentation>
1503     </xs:annotation>
1504     <xs:simpleContent>
1505       <xs:extension base="xs:anyURI">
1506         <xs:anyAttribute namespace="##other" processContents="lax"/>
1507       </xs:extension>
1508     </xs:simpleContent>

```

```

1509     </xs:complexType>
1510 </xs:element>
1511 <xs:element name="Address">
1512   <xs:complexType>
1513     <xs:simpleContent>
1514       <xs:extension base="xs:anyURI">
1515         <xs:anyAttribute namespace="##other" processContents="lax"/>
1516       </xs:extension>
1517     </xs:simpleContent>
1518   </xs:complexType>
1519 </xs:element>
1520 <xs:simpleType name="MessageNumberType">
1521   <xs:restriction base="xs:unsignedLong">
1522     <xs:minInclusive value="1"/>
1523     <xs:maxInclusive value="9223372036854775807"/>
1524   </xs:restriction>
1525 </xs:simpleType>
1526 <!-- Fault Container and Codes -->
1527 <xs:simpleType name="FaultCodes">
1528   <xs:restriction base="xs:QName">
1529     <xs:enumeration value="wsrm:SequenceTerminated"/>
1530     <xs:enumeration value="wsrm:UnknownSequence"/>
1531     <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1532     <xs:enumeration value="wsrm:MessageNumberRollover"/>
1533     <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1534     <xs:enumeration value="wsrm:SequenceClosed"/>
1535     <xs:enumeration value="wsrm:WSRMRequired"/>
1536     <xs:enumeration value="wsrm:UnsupportedSelection"/>
1537   </xs:restriction>
1538 </xs:simpleType>
1539 <xs:complexType name="SequenceFaultType">
1540   <xs:sequence>
1541     <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1542     <xs:element name="Detail" type="wsrm:DetailType" minOccurs="0"/>
1543     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1544 maxOccurs="unbounded"/>
1545   </xs:sequence>
1546   <xs:anyAttribute namespace="##other" processContents="lax"/>
1547 </xs:complexType>
1548 <xs:complexType name="DetailType">
1549   <xs:sequence>
1550     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1551 maxOccurs="unbounded"/>
1552   </xs:sequence>
1553   <xs:anyAttribute namespace="##other" processContents="lax"/>
1554 </xs:complexType>
1555 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1556 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1557 <xs:element name="CreateSequenceResponse"
1558 type="wsrm:CreateSequenceResponseType"/>
1559 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1560 <xs:element name="CloseSequenceResponse"
1561 type="wsrm:CloseSequenceResponseType"/>
1562 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1563 <xs:element name="TerminateSequenceResponse"
1564 type="wsrm:TerminateSequenceResponseType"/>
1565 <xs:complexType name="CreateSequenceType">
1566   <xs:sequence>
1567     <xs:element ref="wsrm:AcksTo"/>
1568     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1569     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1570     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1571 maxOccurs="unbounded"/>

```

```

1572     <xs:annotation>
1573         <xs:documentation>
1574             It is the authors intent that this extensibility be used to
1575 transfer a Security Token Reference as defined in WS-Security.
1576         </xs:documentation>
1577     </xs:annotation>
1578 </xs:any>
1579 </xs:sequence>
1580 <xs:anyAttribute namespace="##other" processContents="lax"/>
1581 </xs:complexType>
1582 <xs:complexType name="CreateSequenceResponseType">
1583     <xs:sequence>
1584         <xs:element ref="wsrm:Identifier"/>
1585         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1586         <xs:element name="IncompleteSequenceBehavior"
1587 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1588         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1589         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1590 maxOccurs="unbounded"/>
1591     </xs:sequence>
1592     <xs:anyAttribute namespace="##other" processContents="lax"/>
1593 </xs:complexType>
1594 <xs:complexType name="CloseSequenceType">
1595     <xs:sequence>
1596         <xs:element ref="wsrm:Identifier"/>
1597         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1598 maxOccurs="unbounded"/>
1599     </xs:sequence>
1600     <xs:anyAttribute namespace="##other" processContents="lax"/>
1601 </xs:complexType>
1602 <xs:complexType name="CloseSequenceResponseType">
1603     <xs:sequence>
1604         <xs:element ref="wsrm:Identifier"/>
1605         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1606 maxOccurs="unbounded"/>
1607     </xs:sequence>
1608     <xs:anyAttribute namespace="##other" processContents="lax"/>
1609 </xs:complexType>
1610 <xs:complexType name="TerminateSequenceType">
1611     <xs:sequence>
1612         <xs:element ref="wsrm:Identifier"/>
1613         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1614 maxOccurs="unbounded"/>
1615     </xs:sequence>
1616     <xs:anyAttribute namespace="##other" processContents="lax"/>
1617 </xs:complexType>
1618 <xs:complexType name="TerminateSequenceResponseType">
1619     <xs:sequence>
1620         <xs:element ref="wsrm:Identifier"/>
1621         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1622 maxOccurs="unbounded"/>
1623     </xs:sequence>
1624     <xs:anyAttribute namespace="##other" processContents="lax"/>
1625 </xs:complexType>
1626 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1627 <xs:complexType name="OfferType">
1628     <xs:sequence>
1629         <xs:element ref="wsrm:Identifier"/>
1630         <xs:element name="Endpoint" type="wsa:EndpointReferenceType"/>
1631         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1632         <xs:element name="IncompleteSequenceBehavior"
1633 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1634         <xs:any namespace="##other" processContents="lax" minOccurs="0"

```

```

1635 maxOccurs="unbounded"/>
1636     </xs:sequence>
1637     <xs:anyAttribute namespace="##other" processContents="lax"/>
1638 </xs:complexType>
1639 <xs:complexType name="AcceptType">
1640     <xs:sequence>
1641         <xs:element ref="wsrm:AcksTo"/>
1642         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1643 maxOccurs="unbounded"/>
1644     </xs:sequence>
1645     <xs:anyAttribute namespace="##other" processContents="lax"/>
1646 </xs:complexType>
1647 <xs:element name="Expires">
1648     <xs:complexType>
1649         <xs:simpleContent>
1650             <xs:extension base="xs:duration">
1651                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1652             </xs:extension>
1653         </xs:simpleContent>
1654     </xs:complexType>
1655 </xs:element>
1656 <xs:simpleType name="IncompleteSequenceBehaviorType">
1657     <xs:restriction base="xs:string">
1658         <xs:enumeration value="DiscardEntireSequence"/>
1659         <xs:enumeration value="DiscardFollowingFirstGap"/>
1660         <xs:enumeration value="NoDiscard"/>
1661     </xs:restriction>
1662 </xs:simpleType>
1663 <xs:element name="UsesSequenceSTR">
1664     <xs:complexType>
1665         <xs:sequence/>
1666         <xs:anyAttribute namespace="##other" processContents="lax"/>
1667     </xs:complexType>
1668 </xs:element>
1669 <xs:element name="UsesSequenceSSL">
1670     <xs:complexType>
1671         <xs:sequence/>
1672         <xs:anyAttribute namespace="##other" processContents="lax"/>
1673     </xs:complexType>
1674 </xs:element>
1675 <xs:element name="UnsupportedElement">
1676     <xs:simpleType>
1677         <xs:restriction base="xs:QName"/>
1678     </xs:simpleType>
1679 </xs:element>
1680 </xs:schema>

```

## 1681 Appendix B. WSDL

1682 This WSDL describes the WS-RM protocol from the point of view of an RM Destination. In the case where  
1683 an endpoint acts both as an RM Destination and an RM Source, note that additional messages may be  
1684 present in exchanges with that endpoint.

1685 Also note that this WSDL is intended to describe the internal structure of the WS-RM protocol, and will not  
1686 generally appear in a description of a WS-RM-capable Web service. See WS-RM Policy [WS-RM Policy]  
1687 for a higher-level mechanism to indicate that WS-RM is engaged.

1688 The normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1689 <http://docs.oasis-open.org/ws-rx/wsr/200608/wsd/wsr-1.1-wsd-200608.wsd>

1690 The following non-normative copy is provided for reference.

```
1691 <?xml version="1.0" encoding="utf-8"?>
1692 <!--
1693 OASIS takes no position regarding the validity or scope of any intellectual
1694 property or other rights that might be claimed to pertain to the
1695 implementation or use of the technology described in this document or the
1696 extent to which any license under such rights might or might not be available;
1697 neither does it represent that it has made any effort to identify any such
1698 rights. Information on OASIS's procedures with respect to rights in OASIS
1699 specifications can be found at the OASIS website. Copies of claims of rights
1700 made available for publication and any assurances of licenses to be made
1701 available, or the result of an attempt made to obtain a general license or
1702 permission for the use of such proprietary rights by implementors or users of
1703 this specification, can be obtained from the OASIS Executive Director.
1704 OASIS invites any interested party to bring to its attention any copyrights,
1705 patents or patent applications, or other proprietary rights which may cover
1706 technology that may be required to implement this specification. Please
1707 address the information to the OASIS Executive Director.
1708 Copyright (c) OASIS Open 2002-2006. All Rights Reserved.
1709 This document and translations of it may be copied and furnished to others,
1710 and derivative works that comment on or otherwise explain it or assist in its
1711 implementation may be prepared, copied, published and distributed, in whole or
1712 in part, without restriction of any kind, provided that the above copyright
1713 notice and this paragraph are included on all such copies and derivative
1714 works. However, this document itself does not be modified in any way, such as
1715 by removing the copyright notice or references to OASIS, except as needed for
1716 the purpose of developing OASIS specifications, in which case the procedures
1717 for copyrights defined in the OASIS Intellectual Property Rights document must
1718 be followed, or as required to translate it into languages other than English.
1719 The limited permissions granted above are perpetual and will not be revoked by
1720 OASIS or its successors or assigns.
1721 This document and the information contained herein is provided on an "AS IS"
1722 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1723 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1724 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1725 FOR A PARTICULAR PURPOSE.
1726 -->
1727 <wsc:definitions xmlns:wsc="http://schemas.xmlsoap.org/wsc/"
1728 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1729 xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:rm="http://docs.oasis-
1730 open.org/ws-rx/wsr/200608" xmlns:tns="http://docs.oasis-open.org/ws-
1731 rx/wsr/200608/wsd" targetNamespace="http://docs.oasis-open.org/ws-
1732 rx/wsr/200608/wsd">
1733 <wsc:types>
```

```

1734     <xs:schema>
1735         <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1736 schemaLocation="http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-
1737 200608.xsd"/>
1738     </xs:schema>
1739 </wsdl:types>

1740 <wsdl:message name="CreateSequence">
1741     <wsdl:part name="create" element="rm:CreateSequence"/>
1742 </wsdl:message>
1743 <wsdl:message name="CreateSequenceResponse">
1744     <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1745 </wsdl:message>
1746 <wsdl:message name="CloseSequence">
1747     <wsdl:part name="close" element="rm:CloseSequence"/>
1748 </wsdl:message>
1749 <wsdl:message name="CloseSequenceResponse">
1750     <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1751 </wsdl:message>
1752 <wsdl:message name="TerminateSequence">
1753     <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1754 </wsdl:message>
1755 <wsdl:message name="TerminateSequenceResponse">
1756     <wsdl:part name="terminateResponse"
1757 element="rm:TerminateSequenceResponse"/>
1758 </wsdl:message>

1759 <wsdl:portType name="SequenceAbstractPortType">
1760     <wsdl:operation name="CreateSequence">
1761         <wsdl:input message="tns:CreateSequence" wsaw:Action="http://docs.oasis-
1762 open.org/ws-rx/wsrn/200608/CreateSequence"/>
1763         <wsdl:output message="tns:CreateSequenceResponse"
1764 wsaw:Action="http://docs.oasis-open.org/ws-
1765 rx/wsrn/200608/CreateSequenceResponse"/>
1766     </wsdl:operation>
1767     <wsdl:operation name="CloseSequence">
1768         <wsdl:input message="tns:CloseSequence" wsaw:Action="http://docs.oasis-
1769 open.org/ws-rx/wsrn/200608/CloseSequence"/>
1770         <wsdl:output message="tns:CloseSequenceResponse"
1771 wsaw:Action="http://docs.oasis-open.org/ws-
1772 rx/wsrn/200608/CloseSequenceResponse"/>
1773     </wsdl:operation>
1774     <wsdl:operation name="TerminateSequence">
1775         <wsdl:input message="tns:TerminateSequence"
1776 wsaw:Action="http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence"/>
1777         <wsdl:output message="tns:TerminateSequenceResponse"
1778 wsaw:Action="http://docs.oasis-open.org/ws-
1779 rx/wsrn/200608/TerminateSequenceResponse"/>
1780     </wsdl:operation>
1781 </wsdl:portType>

1782 </wsdl:definitions>

```



## 1783 Appendix C. Message Examples

### 1784 Appendix C.1 Create Sequence

#### 1785 Create Sequence

```
1786 <?xml version="1.0" encoding="UTF-8"?>
1787 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1788 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1789 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1790   <S:Header>
1791     <wsa:MessageID>
1792       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1793     </wsa:MessageID>
1794     <wsa:To>http://example.com/serviceB/123</wsa:To>
1795     <wsa:Action>http://docs.oasis-open.org/ws-
1796 rx/wsmr/200608/CreateSequence</wsa:Action>
1797     <wsa:ReplyTo>
1798       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1799     </wsa:ReplyTo>
1800   </S:Header>
1801   <S:Body>
1802     <wsmr:CreateSequence>
1803       <wsmr:AcksTo>
1804         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1805       </wsmr:AcksTo>
1806     </wsmr:CreateSequence>
1807   </S:Body>
1808 </S:Envelope>
```

#### 1809 Create Sequence Response

```
1810 <?xml version="1.0" encoding="UTF-8"?>
1811 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1812 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1813 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1814   <S:Header>
1815     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1816     <wsa:RelatesTo>
1817       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1818     </wsa:RelatesTo>
1819     <wsa:Action>
1820       http://docs.oasis-open.org/ws-rx/wsmr/200608/CreateSequenceResponse
1821     </wsa:Action>
1822   </S:Header>
1823   <S:Body>
1824     <wsmr:CreateSequenceResponse>
1825       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1826     </wsmr:CreateSequenceResponse>
1827   </S:Body>
1828 </S:Envelope>
```

### 1829 Appendix C.2 Initial Transmission

1830 The following example WS-ReliableMessaging headers illustrate the message exchange in the above  
1831 figure. The three messages have the following headers; the third message is identified as the last  
1832 message in the Sequence:

1833 **Message 1**

```
1834 <?xml version="1.0" encoding="UTF-8"?>
1835 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1836 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1837 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1838   <S:Header>
1839     <wsa:MessageID>
1840       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1841     </wsa:MessageID>
1842     <wsa:To>http://example.com/serviceB/123</wsa:To>
1843     <wsa:From>
1844       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1845     </wsa:From>
1846     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1847     <wsmr:Sequence>
1848       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1849       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1850     </wsmr:Sequence>
1851   </S:Header>
1852   <S:Body>
1853     <!-- Some Application Data -->
1854   </S:Body>
1855 </S:Envelope>
```

1856 **Message 2**

```
1857 <?xml version="1.0" encoding="UTF-8"?>
1858 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1859 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1860 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1861   <S:Header>
1862     <wsa:MessageID>
1863       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1864     </wsa:MessageID>
1865     <wsa:To>http://example.com/serviceB/123</wsa:To>
1866     <wsa:From>
1867       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1868     </wsa:From>
1869     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1870     <wsmr:Sequence>
1871       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1872       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1873     </wsmr:Sequence>
1874   </S:Header>
1875   <S:Body>
1876     <!-- Some Application Data -->
1877   </S:Body>
1878 </S:Envelope>
```

1879 **Message 3**

```
1880 <?xml version="1.0" encoding="UTF-8"?>
1881 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1882 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1883 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1884   <S:Header>
1885     <wsa:MessageID>
1886       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1887     </wsa:MessageID>
1888     <wsa:To>http://example.com/serviceB/123</wsa:To>
1889     <wsa:From>
1890       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1891 </wsa:From>
1892 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1893 <wsrm:Sequence>
1894 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1895 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1896 </wsrm:Sequence>
1897 <wsrm:AckRequested>
1898 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1899 </wsrm:AckRequested>
1900 </S:Header>
1901 <S:Body>
1902 <!-- Some Application Data -->
1903 </S:Body>
1904 </S:Envelope>

```

### 1905 **Appendix C.3 First Acknowledgement**

1906 Message number 2 has not been accepted by the RM Destination due to some transmission error so it  
1907 responds with an Acknowledgement for messages 1 and 3:

```

1908 <?xml version="1.0" encoding="UTF-8"?>
1909 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1910 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1911 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1912 <S:Header>
1913 <wsa:MessageID>
1914 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1915 </wsa:MessageID>
1916 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1917 <wsa:From>
1918 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1919 </wsa:From>
1920 <wsa:Action>
1921 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1922 </wsa:Action>
1923 <wsrm:SequenceAcknowledgement>
1924 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1925 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1926 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1927 </wsrm:SequenceAcknowledgement>
1928 </S:Header>
1929 <S:Body/>
1930 </S:Envelope>

```

### 1931 **Appendix C.4 Retransmission**

1932 The RM Sourcediscovers that message number 2 was not accepted so it resends the message and  
1933 requests an Acknowledgement:

```

1934 <?xml version="1.0" encoding="UTF-8"?>
1935 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1936 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1937 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1938 <S:Header>
1939 <wsa:MessageID>
1940 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1941 </wsa:MessageID>
1942 <wsa:To>http://example.com/serviceB/123</wsa:To>
1943 <wsa:From>
1944 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1945 </wsa:From>

```

```

1946 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1947 <wsrm:Sequence>
1948 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1949 <wsrm:MessageNumber>2</wsrm:MessageNumber>
1950 </wsrm:Sequence>
1951 <wsrm:AckRequested>
1952 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1953 </wsrm:AckRequested>
1954 </S:Header>
1955 <S:Body>
1956 <!-- Some Application Data -->
1957 </S:Body>
1958 </S:Envelope>

```

## 1959 Appendix C.5 Termination

1960 The RM Destination now responds with an Acknowledgement for the complete Sequence which can then  
1961 be terminated:

```

1962 <?xml version="1.0" encoding="UTF-8"?>
1963 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1964 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1965 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1966 <S:Header>
1967 <wsa:MessageID>
1968 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1969 </wsa:MessageID>
1970 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1971 <wsa:From>
1972 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1973 </wsa:From>
1974 <wsa:Action>
1975 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
1976 </wsa:Action>
1977 <wsrm:SequenceAcknowledgement>
1978 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1979 <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1980 </wsrm:SequenceAcknowledgement>
1981 </S:Header>
1982 <S:Body/>
1983 </S:Envelope>

```

## 1984 Terminate Sequence

```

1985 <?xml version="1.0" encoding="UTF-8"?>
1986 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1987 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1988 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1989 <S:Header>
1990 <wsa:MessageID>
1991 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1992 </wsa:MessageID>
1993 <wsa:To>http://example.com/serviceB/123</wsa:To>
1994 <wsa:Action>
1995 http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence
1996 </wsa:Action>
1997 <wsa:From>
1998 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1999 </wsa:From>
2000 </S:Header>
2001 <S:Body>
2002 <wsrm:TerminateSequence>

```

```
2003     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2004     </wsrm:TerminateSequence>
2005     </S:Body>
2006     </S:Envelope>
```

## 2007 Terminate Sequence Response

```
2008     <?xml version="1.0" encoding="UTF-8"?>
2009     <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2010     xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
2011     xmlns:wsa="http://www.w3.org/2005/08/addressing">
2012     <S:Header>
2013     <wsa:MessageID>
2014     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
2015     </wsa:MessageID>
2016     <wsa:To>http://example.com/serviceA/789</wsa:To>
2017     <wsa:Action>
2018     http://docs.oasis-open.org/ws-rx/wsmr/200608/TerminateSequenceResponse
2019     </wsa:Action>
2020     <wsa:RelatesTo>
2021     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
2022     </wsa:RelatesTo>
2023     <wsa:From>
2024     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2025     </wsa:From>
2026     </S:Header>
2027     <S:Body>
2028     <wsrm:TerminateSequenceResponse>
2029     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2030     </wsrm:TerminateSequenceResponse>
2031     </S:Body>
2032     </S:Envelope>
```

## 2033 Appendix D. State Tables

2034 This appendix specifies the non-normative state transition tables for RM Source and RM Destination.

2035 The state tables describe the lifetime of a sequence in both the RM Source and the RM Destination

2036 Legend:

2037 The first column of these tables contains the motivating event and has the following format:

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

2038 Where:

- 2039 ● Event Name: indicates the name of the event. Event Names surrounded by "<>" are optional as  
2040 described by the specification.
- 2041 ● [source]: indicates the source of the event; one of:
  - 2042 ● [msg] a Received message
  - 2043 ● [int]: an internal event such as the firing of a timer
  - 2044 ● [app]: the application
  - 2045 ● [unspec]: the source is unspecified

2046 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}

2047 Where:

- 2048 ● action to take: indicates that the state machine performs the following action. Actions surrounded  
2049 by "<>" are optional as described by the specification. "Xmit" is used as a short form for the word  
2050 "Transmit"
  - 2051 ● [next state]: indicates the state to which the state machine will advance upon the performance of  
2052 the action. For ease of reading the next state "same" indicates that the state does not change.
  - 2053 ● {ref} is a reference to the document section describing the behavior in this cell
- 2054 "N/A" in a cell indicates a state / event combination self-inconsistent with the state machine; should these  
2055 conditions occur, it would indicate an implementation error. A blank cell indicates that the behavior is not  
2056 described in this specification and does not indicate normal protocol operation. Implementations MAY  
2057 generate a Sequence Terminated fault (see section 4.2) in these circumstances. Robust implementations  
2058 MUST be able to operate in a stable manner despite the occurrence of unspecified event / state  
2059 combinations.

2060 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]	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]	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>CloseSequence</b> [msg] {3.5}	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Xmit CloseSequence Response [Closed] {3.5}</u>	<u>Xmit CloseSequence Response [Closed] {3.5}</u>	<u>Xmit CloseSequence Response [Closed] {3.5}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>
<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}

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
<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] {4.7}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	No action [Closed] {4.7}	No action [Closed] {4.7}	No action [Same]	No action [Same]
<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>TerminateSequence</b> [msg] {3.6}	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>	<u>Xmit Terminate Sequence Response [None] {3.6}</u>	<u>Xmit Terminate Sequence Response [None] {3.6}</u>	<u>Xmit Terminate Sequence Response [None] {3.6}</u>	<u>Generate Unknown Sequence Fault [Same] {4.3}</u>
<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 Acknowledgement</b> [msg] {4.4}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Invalid Acknowledgement Fault [Same] {4.4}	Generate Invalid Acknowledgement Fault [Same] {4.4}	Generate Invalid Acknowledgement Fault [Same] {4.4}	Generate Invalid Acknowledgement Fault [Same] {4.4}

2060 Table 2 RM Destination Sequence State Transition Table

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



Events	Sequence States			
	None	Created	Closing	Closed
<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}	<u>Xmit SeqAck+Final</u> [Same] {3.9}	Xmit SeqAck+Final [Same] {3.9}
<b>CloseSequence</b> [msg] {3.5}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit CloseSequence Response with SeqAck+Final [Closed] {3.5}	<u>Xmit CloseSequenceResponse with SeqAck+Final</u> [Closed] {3.5}	Xmit CloseSequence Response with SeqAck+Final [Closed] {3.5}
<b>&lt;CloseSequence autonomously&gt;</b> [int]	N/A	<u>Xmit CloseSequence with SeqAck+Final</u> [Closing] {3.5}No-Action [Closed]		N/A
<b>CloseSequenceResponse</b> [msg] {3.5}	<u>Generate Unknown Sequence Fault</u> [Same] {4.3}		<u>No Action</u> [Closed] {3.5}	<u>No action</u> [Closed] {3.5}
<b>TerminateSequence</b> [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Terminate Sequence-Response [None] {3.6}	<u>Xmit TerminateSequenceResponse</u> [None] {3.6}	Xmit Terminate-Sequence Response [None] {3.6}
<b>&lt;TerminateSequence autonomously&gt;</b> [int]		<u>Xmit Terminate Sequence</u> [None] with SeqAck+Final {3.6}	<u>Xmit Terminate Sequence</u> [None] with SeqAck+Final {3.6}	<u>Xmit Terminate Sequence</u> [None] with SeqAck+Final {3.6}
<b>TerminateSequenceResponse</b> [msg]	<u>Generate Unknown Sequence Fault</u> [Same] {4.3}			
<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>	N/A			

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

## 2061 **Appendix E. Acknowledgments**

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

2061 Ruslan Bilorusets(BEA), Don Box(Microsoft), Luis Felipe Cabrera(Microsoft), Doug Davis(IBM),  
2062 Donald Ferguson(IBM), Christopher Ferris-Editor(BM), Tom Freund(IBM), Mary Ann Hondo(IBM),  
2063 John Ibbotson(IBM), Lei Jin(BEA), Chris Kaler(Microsoft), David Langworthy-Editor(Microsoft),  
2064 Amelia Lewis(TIBCO Software), Rodney Limprecht(Microsoft), Steve Lucco(Microsoft), Don  
2065 Mullen(TIBCO Software), Anthony Nadalin(IBM), Mark Nottingham(BEA), David Orchard(BEA),  
2066 Jamie Roots(IBM), Shivajee Samdarshi(TIBCO Software), John Shewchuk(Microsoft), Tony  
2067 Storey(IBM).

2061 The following individuals have provided invaluable input into the initial contribution:

2061 Keith Ballinger(Microsoft), Stefan Batres(Microsoft), Rebecca Bergersen(Iona), Allen  
2062 Brown(Microsoft), Michael Conner(IBM), George Copeland(Microsoft), Francisco Curbera(IBM),  
2063 Paul Fremantle(IBM), Steve Graham(IBM), Pat Helland(Microsoft), Rick Hill(Microsoft), Scott  
2064 Hinkelman(IBM), Tim Holloway(IBM), Efim Hudis(Microsoft), David Ingham(Microsoft), Gopal  
2065 Kakivaya(Microsoft), Johannes Klein(Microsoft), Frank Leymann(IBM), Martin Nally(IBM), Peter  
2066 Niblett(IBM), Jeffrey Schlimmer(Microsoft), James Snell(IBM), Keith Stobie(Microsoft), Satish  
2067 Thatte(Microsoft), Stephen Todd(IBM), Sanjiva Weerawarana(IBM), Roger Wolter(Microsoft).

2061 The following individuals were members of the committee during the development of this specification:

2061 Abbie Barbir(Nortel), Charlton Barreto(Adobe), Stefan Batres(Microsoft), Hamid Ben  
2062 Malek(Fujitsu), Andreas Bjarlestam(Ericsson), Toufic Boubez(Layer 7), Doug Bunting(Sun), Lloyd  
2063 Burch(Novell), Steve Carter(Novell), Martin Chapman(Oracle), Dave Chappell(Sonic), Paul  
2064 Cotton(Microsoft), Glen Daniels(Sonic), Doug Davis(IBM), Blake Dournaee(Intel), Jacques  
2065 Durand(Fujitsu), Colleen Evans(Microsoft), Christopher Ferris(IBM), Paul Fremantle(WSO2),  
2066 Robert Freund(Hitachi), Peter Furniss(Erebor), Marc Goodner(Microsoft), Alastair  
2067 Green(Choreology), Mike Grogan(Sun), Ondrej Hrebicek(Microsoft), Kazunori Iwasa(Fujitsu),  
2068 Chamikara Jayalath(WSO2), Lei Jin(BEA), Ian Jones(BTplc), Anish Karmarkar(Oracle), Paul  
2069 Knight(Nortel), Dan Leshchiner(Tibco), Mark Little(JBoss), Lily Liu(webMethods), Matt  
2070 Lovett(IBM), Ashok Malhotra(Oracle), Jonathan Marsh(Microsoft), Daniel Millwood(IBM), Jeff  
2071 Mischkinsky(Oracle), Nilo Mitra(Ericsson), Peter Niblett(IBM), Duane Nickull(Adobe), Eisaku  
2072 Nishiyama(Hitachi), Dave Orchard(BEA), Chouthri Palanisamy(NEC), Sanjay Patil(SAP), Gilbert  
2073 Pilz(BEA), Martin Raepfle(SAP), Eric Rajkovic(Oracle), Stefan Rossmannith(SAP), Tom  
2074 Rutt(Fujitsu), Rich Salz(IBM), Shivajee Samdarshi(Tibco), Vladimir Videlov(SAP), Claus von  
2075 Riegen(SAP), Pete Wenzel(Sun), Steve Winkler(SAP), Ümit Yalçinalp(SAP), Nobuyuki  
2076 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

## 2061 **Appendix G. Notices**

2062 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that  
2063 might be claimed to pertain to the implementation or use of the technology described in this document or  
2064 the extent to which any license under such rights might or might not be available; neither does it represent  
2065 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to  
2066 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made  
2067 available for publication and any assurances of licenses to be made available, or the result of an attempt  
2068 made to obtain a general license or permission for the use of such proprietary rights by implementors or  
2069 users of this specification, can be obtained from the OASIS Executive Director.

2070 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or  
2071 other proprietary rights which may cover technology that may be required to implement this specification.  
2072 Please address the information to the OASIS Executive Director.

2073 Copyright (C) OASIS Open (2006). All Rights Reserved.

2074 This document and translations of it may be copied and furnished to others, and derivative works that  
2075 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and  
2076 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and  
2077 this paragraph are included on all such copies and derivative works. However, this document itself may  
2078 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as  
2079 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights  
2080 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it  
2081 into languages other than English.

2082 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors  
2083 or assigns.

2084 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
2085 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY  
2086 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR  
2087 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.