



1 Web Services ReliableMessaging 2 (WS-Reliable Messaging)

3 Working Draft 087, January 4, 2006

4 Document identifier:

5 wsrn-1.1-spec-wd-087

6 Location:

7 Editors:

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

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

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

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

12 [Ümit Yalçınalp, SAP <umit.yalcinalp@sap.com>](mailto:umit.yalcinalp@sap.com)

13 TBD

14 Contributors:

15 TBD

16 Abstract:

17 This specification (WS-ReliableMessaging) describes a protocol that allows messages to be delivered
18 reliably between distributed applications in the presence of software component, system, or network
19 failures. The protocol is described in this specification in a transport-independent manner allowing it to be
20 implemented using different network technologies. To support interoperable Web services, a SOAP
21 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 \[WSDL 1.1\]](#) ~~SOAP [SOAP] and~~
26 ~~WSDL [WSDL]~~ extensibility model, SOAP-based and WSDL-based specifications are designed to be
27 composed with each other to define a rich Web services environment. As such, WS-ReliableMessaging
28 by itself does not define all the features required for a complete messaging solution. WS-
29 ReliableMessaging is a building block that is used in conjunction with other specifications and
30 application-specific protocols to accommodate a wide variety of protocols related to the operation of
31 distributed Web services.

32 Status:

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

35 **Table of Contents**

36	1 Introduction.....	4
37	1.1 Goals and Requirements.....	4
38	1.1.1 Requirements.....	4
39	1.2 Notational Conventions.....	4
40	1.3 Namespace.....	4
41	1.4 Compliance.....	5
42	2 Reliable Messaging Model.....	6
43	2.1 Glossary.....	6
44	2.2 Protocol Preconditions.....	7
45	2.3 Protocol Invariants.....	7
46	2.4 Example Message Exchange.....	7
47	3 RM Protocol Elements.....	10
48	3.1 Sequence Creation.....	10
49	3.2 Closing A Sequence.....	13
50	3.3 Sequence Termination.....	14
51	3.4 Sequences.....	15
52	3.5 Request Acknowledgement.....	16
53	3.6 Sequence Acknowledgement.....	17
54	4 Faults.....	20
55	4.1 SequenceFault Element.....	21
56	4.2 Sequence Terminated.....	22
57	4.3 Unknown Sequence.....	22
58	4.4 Invalid Acknowledgement.....	22
59	4.5 Message Number Rollover.....	23
60	4.6 Create Sequence Refused.....	23
61	4.7 Sequence Closed.....	23
62	5 Security Considerations.....	24
63	6 References.....	26
64	6.1 Normative.....	26
65	6.2 Non-Normative.....	26
66	A. Schema.....	28
67	B. Message Examples.....	32
68	B.1 Create Sequence.....	32
69	B.2 Initial Transmission.....	32
70	B.3 First Acknowledgement.....	34
71	B.4 Retransmission.....	34
72	B.5 Termination.....	35

73	C. WSDL.....	37
74	D. Acknowledgments.....	39
75	E. Revision History.....	40
76	F. Notices.....	43

77 1 Introduction

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

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

86 1.1 Goals and Requirements

87 1.1.1 Requirements

88 1.2 Notational Conventions

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

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

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

109 1.3 Namespace

110 The XML namespace [[XML-ns](#)] URI that MUST be used by implementations of this specification is:

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

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

114 The following namespaces are used in this document:

115 *Table 1*

Prefix	Namespace
S	http://www.w3.org/2003/05/soap-envelope
S11	http://schemas.xmlsoap.org/soap/envelope/
wsrn	http://docs.oasis-open.org/ws-rx/wsrn/200510
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing
xs	http://www.w3.org/2001/XMLSchema
Prefix	Namespace
S	http://www.w3.org/2003/05/soap-envelope
S11	http://schemas.xmlsoap.org/soap/envelope/
wsrn	http://docs.oasis-open.org/ws-rx/wsrn/200510
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
xs	http://www.w3.org/2001/XMLSchema

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

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

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

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

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

123 1.4 Compliance

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

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

2 Reliable Messaging Model

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

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

140 ~~The protocol supports reliability features which include ordered delivery, duplicate elimination, and~~
141 ~~guaranteed receipt for the RMD. It is expected that the AD and RMD will implement as many of these or~~
142 ~~as few of these characteristics as necessary to implement the AD. In any case the wire protocol does not~~
143 ~~change. In addition, the protocol allows the RM Source and RM Destination to provide their respective~~
144 ~~Application Source and Application Destination a guarantee that a message that is sent by an Application~~
145 ~~Source will be delivered to the Application Destination.~~

146 ~~Figure 1 below illustrates the entities and events in a simple reliable exchange of messages. First, the~~
147 ~~Application Source Sends a message for reliable delivery. The Reliable Messaging (RM) Source accepts~~
148 ~~the message and Transmits it one or more times. After receiving the message, the RM Destination~~
149 ~~Acknowledges it. Finally, the RM Destination delivers the message to the Application Destination. The~~
150 ~~exact roles the entities play and the complete meaning of the events will be defined throughout. This~~
151 ~~guarantee is specified as a delivery assurance. It is the responsibility of the RM Source and RM~~
152 ~~Destination to fulfill the delivery assurances on behalf of their respective Application counterparts, or raise~~
153 ~~an error. The protocol defined here allows endpoints to meet this guarantee for the delivery assurances~~
154 ~~defined below. However, the means by which these delivery assurances are manifested by either the RM~~
155 ~~Source or RM Destination roles is an implementation concern, and is out of scope of this specification.~~

156 ~~Note that the underlying protocol defined in this specification remains the same regardless of the delivery~~
157 ~~assurance.~~

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

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

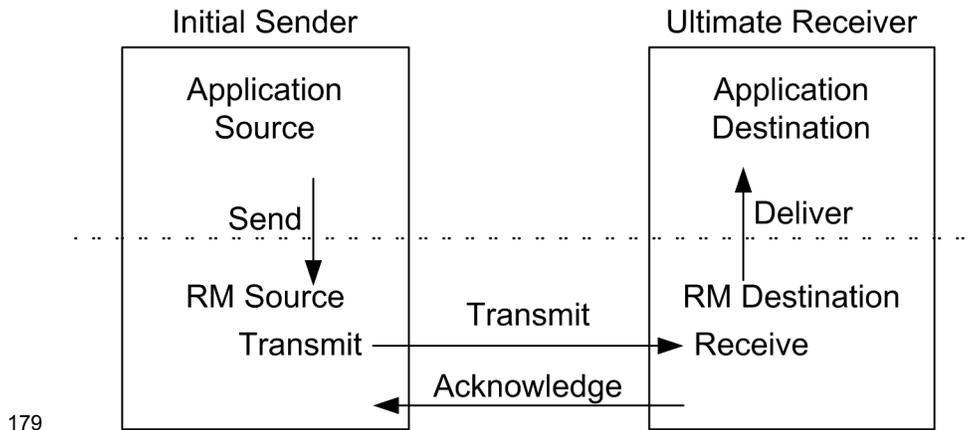
162 ~~**AtMostOnce** Messages will be delivered at most once without duplication or an error will be raised on at~~
163 ~~least one endpoint. It is possible that some messages in a sequence may not be delivered.~~

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

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

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

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



180 Figure 1: Reliable Messaging Model

181 2.1 Glossary

182 The following definitions are used throughout this specification:

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

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

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

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

189 ~~**Delivery Assurance:** The guarantee that the messaging infrastructure provides on the delivery of a~~
 190 ~~message.~~

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

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

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

197 ~~**RM Source:** The endpoint that transmits the message.~~

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

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

201 2.2 Protocol Preconditions

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

- 204 • For any single message exchange the RM Source MUST have an endpoint reference that uniquely
205 identifies the RM Destination endpoint.
- 206 • The RM Source MUST have knowledge of the destination's policies, if any, and the RM Source
207 MUST be capable of formulating messages that adhere to this policy.

208 If a secure exchange of messages is required, then the RM Source and RM Destination MUST have a
209 security context.

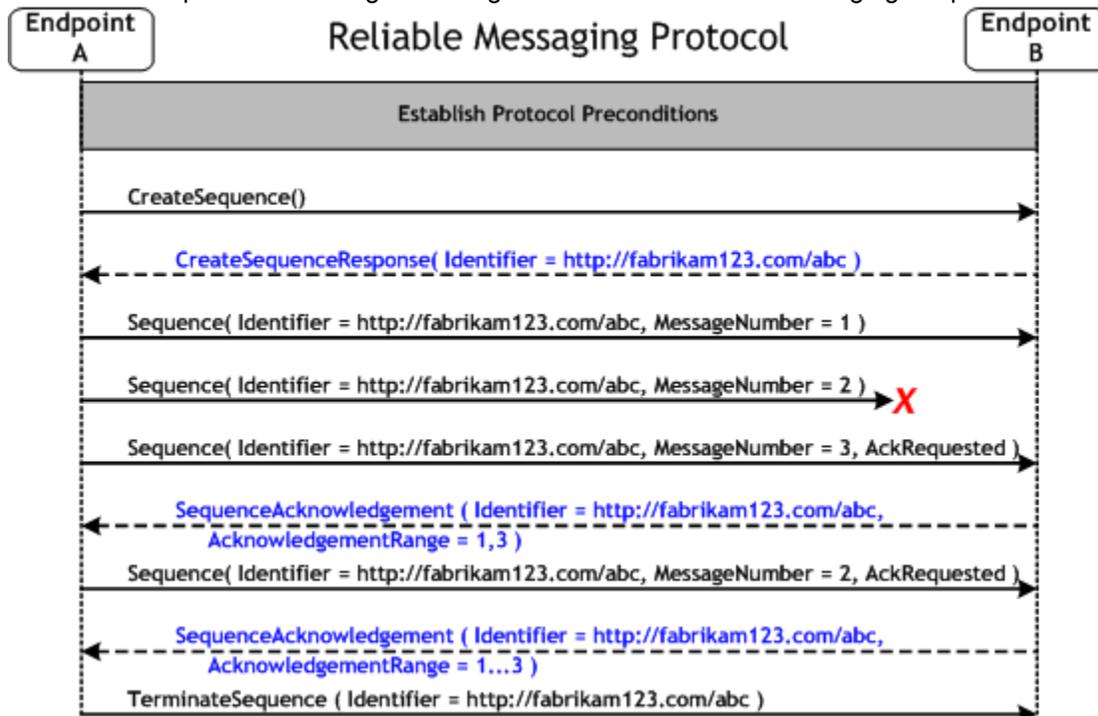
210 2.3 Protocol Invariants

211 During the lifetime of ~~a Sequence~~the protocol, two invariants are REQUIRED for correctness:

- 212 • The RM Source MUST assign each ~~message within a Sequence~~ a message number (defined
213 below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers
214 MUST be assigned in the same order in which messages are sent by the Application Source
215 ~~message~~ a sequence number (defined below) beginning at 1 and increasing by exactly 1 for each
216 subsequent reliable message.
- 217 • Every acknowledgement issued by the RM Destination MUST include within an acknowledgement
218 range or ranges the sequence number of every message successfully received by the RM
219 Destination and MUST exclude sequence numbers of any messages not yet received.

220 2.4 Example Message Exchange

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



222 Figure 2: The WS-ReliableMessaging Protocol

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

264 3 RM Protocol Elements

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

269 3.1 Sequence Creation

270 The RM Source MUST request creation of an outbound Sequence by sending a
271 `<wsrm:CreateSequence>` element in the body of a message to the RM Destination which in turn
272 responds either with a `<wsrm:CreateSequenceResponse>` or a `CreateSequenceRefused` fault in
273 the body of the response message. `<wsrm:CreateSequence>` MAY carry an offer to create an inbound
274 sequence which is either accepted or rejected in the `<wsrm:CreateSequenceResponse>`. Note,
275 offering a Sequence within the `<wsrm:CreateSequence>` element is simply a protocol optimization.
276 There is no semantic difference between offering a Sequence, and choosing not to offer one and
277 subsequently creating a new Sequence to carry messages from the RM Destination to the RM Source.

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

```
279 <wsrm:CreateSequence ...>  
280   <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>  
281   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
282   <wsrm:Offer ...>  
283     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
284     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
285     ...  
286   </wsrm:Offer> ?  
287   ...  
288 </wsrm:CreateSequence>
```

289 `/wsrm:CreateSequence`

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

294 `/wsrm:CreateSequence/wsrm:AcksTo`

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

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

301 `/wsrm:CreateSequence/wsrm:Expires`

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

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

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

309 /wsmr:CreateSequence/wsmr:Offer

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

312 /wsmr:CreateSequence/wsmr:Offer/wsmr:Identifier

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

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

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

318 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

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

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

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

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

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

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

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

331 /wsmr:CreateSequence/{any}

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

334 /wsmr:CreateSequence/@{any}

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

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

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

```
342 <wsmr:CreateSequenceResponse ...>  
343   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
344   <wsmr:Expires> xs:duration </wsmr:Expires> ?  
345   <wsmr:Accept ...>  
346     <wsmr:AcksTo ...> wsa:EndpointReferenceType </wsmr:AcksTo>
```

347
348
349
350

```
    ...  
  </wsrm:Accept> ?  
  ...  
</wsrm:CreateSequenceResponse>
```

351 /wsrm:CreateSequenceResponse

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

355 /wsrm:CreateSequenceResponse/wsrm:Identifier

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

358 /wsrm:CreateSequenceResponse/wsrm:Identifier/@{any}

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

361 /wsrm:CreateSequenceResponse/wsrm:Expires

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

366 /wsrm:CreateSequenceResponse/wsrm:Expires/@{any}

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

369 /wsrm:CreateSequenceResponse/wsrm:Accept

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

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

375 /wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo

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

379 /wsrm:CreateSequenceResponse/wsrm:Accept/{any}

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

382 /wsrm:CreateSequenceResponse/wsrm:Accept/@{any}

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

385 /wsrm:CreateSequenceResponse/{any}

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

388 /wsrm:CreateSequenceResponse/@{any}

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

391 **3.2 Closing A Sequence**

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

397 If the RM Source wishes to close the Sequence then it sends a `<wsrm:CloseSequence>` element, in the
398 body of a message, to the RM Destination. This message indicates that RM Destination MUST NOT
399 receive any new messages for the specified sequence, other than those already received at the time the
400 `<wsrm:CloseSequence>` element is interpreted by the RMD. Upon receipt of this message, or
401 subsequent to the RM Destination closing the Sequence of its own volition, the RM Destination MUST
402 include a final SequenceAcknowledgement (that MUST include the `<wsrm:Final>` element) header block
403 on each message destined to the RM Source, including the CloseSequenceResponse message and on
404 any Sequence Fault transmitted to the RMS. ~~the RM Destination MUST include a~~
405 SequenceAcknowledgement header block in the CloseSequenceResponse message. Note, this
406 SequenceAcknowledgement MUST include the `<wsrm:Final>` element.

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

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

415 The following exemplar defines the CloseSequence syntax:

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

417 /wsrm:CloseSequence

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

421 /wsrm:CloseSequence@Identifier

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

424 /wsrm:CloseSequence/{any}

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

427 /wsrm:CloseSequence@{any}

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

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

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

```
434 /wsrm:CloseSequenceResponse
```

```
435 /wsrm:CloseSequenceResponse
```

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

```
439 /wsrm:CloseSequenceResponse/{any}
```

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

```
442 /wsrm:CloseSequenceResponse@{any}
```

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

445 3.3 Sequence Termination

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

453 The following exemplar defines the TerminateSequence syntax:

```
454 <wsrm:TerminateSequence ...>  
455   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
456   ...  
457 </wsrm:TerminateSequence>
```

```
458 /wsrm:TerminateSequence
```

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

```
463 /wsrm:TerminateSequence/wsrm:Identifier
```

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

```
466 /wsrm:TerminateSequence/wsrm:Identifier/@{any}
```

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

```
469 /wsrm:TerminateSequence/{any}
```

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

472 /wsm:TerminateSequence/@{any}

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

475 3.4 Sequences

476 The RM protocol uses a <wsm:Sequence> header block to track and manage the reliable delivery of
477 messages. Messages for which ~~a reliable delivery is required~~~~the delivery assurance applies~~ MUST
478 contain a <wsm:Sequence> header block. Each Sequence MUST have a unique
479 <wsm:Identifier> element and each message within a Sequence MUST have a
480 <wsm:MessageNumber> element that increments by 1 from an initial value of 1. These values are
481 contained within a <wsm:Sequence> header block accompanying each message being delivered in the
482 context of a Sequence.

483 There MUST be no more than one <wsm:Sequence> header block in any message.

484 A following exemplar defines its syntax:

```
485 <wsm:Sequence ...>  
486   <wsm:Identifier ...> xs:anyURI </wsm:Identifier>  
487   <wsm:MessageNumber> xs:unsignedLong </wsm:MessageNumber>  
488   ...  
489 </wsm:Sequence>
```

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

491 /wsm:Sequence

492 This is the element containing Sequence information for WS-ReliableMessaging. The <wsm:Sequence>
493 element MUST be understood by the RM Destination. The <wsm:Sequence> element MUST have a
494 mustUnderstand attribute with a value 1/true from the namespace corresponding to the version of
495 SOAP to which the <wsm:Sequence> SOAP header block is bound.

496 /wsm:Sequence/wsm:Identifier

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

499 /wsm:Sequence/wsm:Identifier/@{any}

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

502 /wsm:Sequence/wsm:MessageNumber

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

508 /wsm:Sequence/{any}

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

511 /wsrm:Sequence/@{any}

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

514 The following example illustrates a Sequence header block.

```
515 <wsrm:Sequence>  
516   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
517   <wsrm:MessageNumber>10</wsrm:MessageNumber>  
518 </wsrm:Sequence>
```

519 **3.5 Request Acknowledgement**

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

522 At any time, the RM Source may request an acknowledgement message from the RM Destination using
523 an <wsrm:AckRequested> header block.

524 The RM Source may request an acknowledgement message from the RM Destination at any time~~requests-~~
525 ~~this acknowledgement~~ by including an <wsrm:AckRequested> header block in the message. An RM
526 Destination that receives a message that contains an <wsrm:AckRequested> header block MUST
527 respond with a message containing a <wsrm:SequenceAcknowledgement> header block. If a non-
528 mustUnderstand fault occurs when processing an RM Header that was piggy-backed on another
529 message, a fault MUST be generated, but the processing of the original message MUST NOT be
530 affected.

531 The following exemplar defines its syntax:

```
532 <wsrm:AckRequested ...>  
533   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
  
534   ...  
535 </wsrm:AckRequested>
```

536 /wsrm:AckRequested

537 This element requests an acknowledgement for the identified sequence.

538 /wsrm:AckRequested/wsrm:Identifier

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

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

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

544 /wsrm:AckRequested/{any}

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

547 /wsrm:AckRequested/@{any}

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

550 3.6 Sequence Acknowledgement

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

560 The following exemplar defines its syntax:

```
561 <wsrm:SequenceAcknowledgement ...>
562   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
563   [ [ <wsrm:AcknowledgementRange ...
564     Upper="xs:unsignedLong"
565     Lower="xs:unsignedLong"/> +
566
567     | <wsrm:None/> ]
568     <wsrm:Final/> ?
569     | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> + ]
570   ...
571 </wsrm:SequenceAcknowledgement>
```

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

573 `/wsrm:SequenceAcknowledgement`

574 This element contains the Sequence acknowledgement information.

575 `/wsrm:SequenceAcknowledgement/wsrm:Identifier`

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

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

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

582 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange`

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

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

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

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

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

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

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

596 /wsmr:SequenceAcknowledgement/wsmr:Final

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

603 /wsmr:SequenceAcknowledgement/wsmr:Nack

604 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the
605 <wsmr:MessageNumber> of an unreceived message in a Sequence. This element permits the gap
606 analysis of the <wsmr:AcknowledgementRange> elements to be performed at the RM Destination
607 rather than at the RM Source which may yield performance benefits in certain environments. The
608 <wsmr:Nack> element MUST NOT be present if a sibling <wsmr:AcknowledgementRange> or
609 <wsmr:None> elements are also present as a child of <wsmr:SequenceAcknowledgement>. Upon the
610 receipt of a Nack, an RM Source SHOULD retransmit the message identified by the Nack. The RM
611 Destination MUST NOT issue a <wsmr:SequenceAcknowledgement> containing a <wsmr:Nack> for
612 a message that it has previously acknowledged within a <wsmr:AcknowledgementRange>. The RM
613 Source SHOULD ignore a <wsmr:SequenceAcknowledgement> containing a <wsmr:Nack> for a
614 message that has previously been acknowledged within a <wsmr:AcknowledgementRange>.

615 /wsmr:SequenceAcknowledgement/wsmr:None

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

620 /wsmr:SequenceAcknowledgement/{any}

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

623 /wsmr:SequenceAcknowledgement/@{any}

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

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

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

```
628 <wsmr:SequenceAcknowledgement>  
629     <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>  
630     <wsmr:AcknowledgementRange Upper="10" Lower="1"/>  
631 </wsmr:SequenceAcknowledgement>
```

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

```
634 <wsmr:SequenceAcknowledgement>  
635     <wsmr:Identifier>http://example.com/abc</wsmr:Identifier>
```

```
636     <wsrm:AcknowledgementRange Upper="2" Lower="1"/>
637     <wsrm:AcknowledgementRange Upper="6" Lower="4"/>
638     <wsrm:AcknowledgementRange Upper="10" Lower="8"/>
639 </wsrm:SequenceAcknowledgement>
```

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

```
641 <wsrm:SequenceAcknowledgement>
642     <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
643     <wsrm:Nack>3</wsrm:Nack>
644 </wsrm:SequenceAcknowledgement>
```

645 4 Faults

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

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

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

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

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

663 The definitions of faults use the following properties:

664 [Code] The fault code.

665 [Subcode] The fault subcode.

666 [Reason] The English language reason element.

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

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

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

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

```
671 <S:Envelope>  
672   <S:Header>  
673     <wsa:Action>  
674       http://schemas.xmlsoap.org/ws/2004/08/addressing/fault  
675     </wsa:Action>  
676     <!-- Headers elided for clarity. -->  
677   </S:Header>  
678   <S:Body>  
679     <S:Fault>  
680       <S:Code>  
681         <S:Value> [Code] </S:Value>  
682         <S:Subcode>  
683           <S:Value> [Subcode] </S:Value>  
684         </S:Subcode>  
685       </S:Code>  
686       <S:Reason>
```

```

687     <S:Text xml:lang="en"> [Reason] </S:Text>
688   </S:Reason>
689   <S:Detail>
690     [Detail]
691     ...
692   </S:Detail>
693 </S:Fault>
694 </S:Body>
695 </S:Envelope>

```

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

```

698 <S11:Envelope>
699   <S11:Header>
700     <wsrm:SequenceFault>
701       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
702       ...
703     </wsrm:SequenceFault>
704     <!-- Headers elided for clarity. -->
705   </S11:Header>
706   <S11:Body>
707     <S11:Fault>
708       <faultcode> [Code] </faultcode>
709       <faultstring> [Reason] </faultstring>
710     </S11:Fault>
711   </S11:Body>
712 </S11:Envelope>

```

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

```

715 <S11:Envelope>
716   <S11:Body>
717     <S11:Fault>
718       <faultcode> [Subcode] </faultcode>
719       <faultstring xml:lang="en"> [Reason] </faultstring>
720     </S11:Fault>
721   </S11:Body>
722 </S11:Envelope>

```

723 4.1 SequenceFault Element

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

728 The following exemplar defines its syntax:

```

729 <wsrm:SequenceFault ...>
730   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
731   ...
732 </wsrm:SequenceFault>

```

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

734 /wsrm:SequenceFault

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

736 /wsm:SequenceFault/wsm:FaultCode

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

738 /wsm:SequenceFault/{any}

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

741 /wsm:SequenceFault/@{any}

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

744 **4.2 Sequence Terminated**

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

749 Properties:

750 [Code] Sender or Receiver

751 [Subcode] wsm:SequenceTerminated

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

753 [Detail]

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

755 **4.3 Unknown Sequence**

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

758 Properties:

759 [Code] Sender

760 [Subcode] wsm:UnknownSequence

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

762 [Detail]

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

764 **4.4 Invalid Acknowledgement**

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

768 [Code] Sender

769 [Subcode] wsm:InvalidAcknowledgement

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

771 [Detail]

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

773 **4.5 Message Number Rollover**

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

775 Properties:

776 [Code] Sender

777 [Subcode] wsrm:MessageNumberRollover

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

779 [Detail]

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

781 **4.6 Create Sequence Refused**

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

783 Properties:

784 [Code] Sender

785 [Subcode] wsrm:CreateSequenceRefused

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

787 [Detail]

```
788 xs:any
```

789 **4.7 Sequence Closed**

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

793 Properties:

794 [Code] Sender

795 [Subcode] wsrm:SequenceClosed

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

797 [Detail]

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

799 5 Security Considerations

800 It is strongly recommended that the communication between services be secured using the mechanisms
801 described in WS-Security. In order to properly secure messages, the body and all relevant headers need
802 to be included in the signature. Specifically, the <wsrm:Sequence> header needs to be signed with the
803 body in order to "bind" the two together. In order to properly secure messages, the body and all relevant
804 headers need to be included in the signature. Specifically, the <wsrm:Sequence> header needs to be
805 signed with the body in order to "bind" the two together. The <wsrm:SequenceAcknowledgement>
806 header may be signed independently because a reply independent of the message is not a security
807 concern.

808 Because Sequences are expected to exchange a number of messages, it is recommended that a security
809 context be established using the mechanisms described in WS-Trust and WS-SecureConversation
810 [SecureConversation]. If a Sequence is bound to a specific destination, then the security context needs to
811 be established or shared with the destination servicing the Sequence. While the context can be
812 established at any time, it is critical that the messages establishing the Sequence be secured even if they
813 precede security context establishment. However, it is recommended that the security context be
814 established first. Security contexts are independent of reliable messaging Sequences. Consequently,
815 security contexts can come and go independent of the lifetime of the Sequence. If a Sequence is bound to
816 a specific destination, then the security context needs to be established or shared with the destination
817 servicing the Sequence. While the context can be established at any time, it is critical that the messages
818 establishing the Sequence be secured even if they precede security context establishment. However, it is
819 recommended that the security context be established first. Security contexts are independent of reliable
820 messaging Sequences. Consequently, security contexts can come and go independent of the lifetime of
821 the Sequence. In fact, it is recommended that the lifetime of a security context be less than the lifetime of
822 the Sequence unless the Sequence is very short-lived.

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

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

833 The security context may be re-established using the mechanisms described in WS-Trust and WS-
834 SecureConversation. Similarly, secrets can be exchanged using the mechanisms described in WS-Trust.
835 Note, however, that the current shared secret should not be used to encrypt the new shared secret.
836 Similarly, secrets can be exchanged using the mechanisms described in WS-Trust. Note, however, that
837 the current shared secret should not be used to encrypt the new shared secret. Derived keys, the preferred
838 solution from this list, can be specified using the mechanisms described in WS-SecureConversation.

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

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

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

- 855 • **Message alteration** – Alteration is prevented by including signatures of the message information
856 using WS-Security.
- 857 • **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-Security.
- 858 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by comparing
859 secured policies – see WS-Policy and WS-SecurityPolicy).
- 860 • **Authentication** – Authentication is established using the mechanisms described in WS-Security
861 and WS-Trust. Each message is authenticated using the mechanisms described in WS-Security.
- 862 • **Accountability** – Accountability is a function of the type of and string of the key and algorithms
863 being used. ~~In many cases, a strong symmetric key provides sufficient accountability. In many~~
864 ~~cases, a strong symmetric key provides sufficient accountability.~~ However, in some environments,
865 strong PKI signatures are required.
- 866 • **Availability** – All reliable messaging services are subject to a variety of availability attacks. ~~Replay~~
867 ~~detection is a common attack and it is recommended that this be addressed by the mechanisms~~
868 ~~described in WS-Security. (Note that because of legitimate message replays, detection should~~
869 ~~include a differentiator besides message id such as a timestamp). Other attacks, such as network-~~
870 ~~level denial of service attacks are harder to avoid and are outside the scope of this specification.~~
871 ~~Replay detection is a common attack and it is recommended that this be addressed by the~~
872 ~~mechanisms described in WS-Security. (Note that because of legitimate message replays, detection~~
873 ~~should include a differentiator besides message id such as a timestamp). Other attacks, such as~~
874 ~~network-level denial of service attacks are harder to avoid and are outside the scope of this~~
875 ~~specification.~~ That said, care should be taken to ensure that minimal state is saved prior to any
876 authenticating sequences.

877 **6 References**

878 **6.1 Normative**

879 **[KEYWORDS]**

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

882 **[SOAP 1.1]**

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

884 **[SOAP 1.2]**

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

886 **[URI]**

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

889 **[XML]**

890 [W3C Recommendation, "Extensible Markup Language \(XML\) 1.0 \(Second Edition\)", October 2000.](#)

891 **[XML-ns]**

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

893 **[XML-Schema Part1]**

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

895 **[XML-Schema Part2]**

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

897 **[WSDL 1.1Security]**

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

900 **[RTTM]**

901 ~~V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC-1323, May 1992.~~

902 **[WSDL]**

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

904 **[WS-Addressing]**

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

906 **6.2 Non-Normative**

907 **[WS-Policy]**

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

909 **[WS-PolicyAttachment]**

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

911 **[WSecurity]**

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

914 **[RTTM]**

915 [V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, May
916 1992.](#)

917 **[SecurityPolicy]**

918 [G. Della-Libra, et. al. "Web Services Security Policy Language \(WS-SecurityPolicy\)". July 2005"~~Web-~~
919 ~~Services Security Policy Language \(WS-SecurityPolicy\),"~~ December 2002:](#)

920 **[SecureConversation]**

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

922 A. Schema

923 The normative schema [that is defined for WS-ReliableMessaging using \[XML-Schema Part1\] and \[XML-](#)
924 [Schema Part2\]for WS-ReliableMessaging](#) is located at:

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

926 The following copy is provided for reference.

```
927 <?xml version="1.0" encoding="UTF-8"?>
928 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
929 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
930 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
931 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"
932 elementFormDefault="qualified" attributeFormDefault="unqualified">
933 <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
934 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
935 <!-- Protocol Elements -->
936 <xs:complexType name="SequenceType">
937 <xs:sequence>
938 <xs:element ref="wsrm:Identifier"/>
939 <xs:element name="MessageNumber" type="xs:unsignedLong"/>
940 <xs:any namespace="##other" processContents="lax" minOccurs="0"
941 maxOccurs="unbounded"/>
942 </xs:sequence>
943 <xs:anyAttribute namespace="##other" processContents="lax"/>
944 </xs:complexType>
945 <xs:element name="Sequence" type="wsrm:SequenceType"/>
946 <xs:element name="SequenceAcknowledgement">
947 <xs:complexType>
948 <xs:sequence>
949 <xs:element ref="wsrm:Identifier"/>
950 <xs:choice>
951 <xs:sequence>
952 <xs:choice>
953 <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
954 <xs:complexType>
955 <xs:sequence/>
956 <xs:attribute name="Upper" type="xs:unsignedLong"
957 use="required"/>
958 <xs:attribute name="Lower" type="xs:unsignedLong"
959 use="required"/>
960 <xs:anyAttribute namespace="##other" processContents="lax"/>
961 </xs:complexType>
962 </xs:element>
963 <xs:element name="None" minOccurs="0">
964 <xs:complexType>
965 <xs:sequence/>
966 </xs:complexType>
967 </xs:element>
968 </xs:choice>
969 <xs:element name="Final" minOccurs="0">
970 <xs:complexType>
971 <xs:sequence/>
972 </xs:complexType>
973 </xs:element>
974 </xs:sequence>
975 <xs:element name="Nack" type="xs:unsignedLong"
976 maxOccurs="unbounded"/>
977 </xs:choice>
```

```

978         <xs:any namespace="##other" processContents="lax" minOccurs="0"
979 maxOccurs="unbounded"/>
980     </xs:sequence>
981     <xs:anyAttribute namespace="##other" processContents="lax"/>
982 </xs:complexType>
983 </xs:element>
984 <xs:complexType name="AckRequestedType">
985     <xs:sequence>
986         <xs:element ref="wsrm:Identifier"/>
987         <xs:any namespace="##other" processContents="lax" minOccurs="0"
988 maxOccurs="unbounded"/>
989     </xs:sequence>
990     <xs:anyAttribute namespace="##other" processContents="lax"/>
991 </xs:complexType>
992 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
993 <xs:element name="Identifier">
994     <xs:complexType>
995         <xs:annotation>
996             <xs:documentation>
997                 This type is for elements whose [children] is an anyURI and can have
998                 arbitrary attributes.
999             </xs:documentation>
1000         </xs:annotation>
1001         <xs:simpleContent>
1002             <xs:extension base="xs:anyURI">
1003                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1004             </xs:extension>
1005         </xs:simpleContent>
1006     </xs:complexType>
1007 </xs:element>
1008 <!-- Fault Container and Codes -->
1009 <xs:simpleType name="FaultCodes">
1010     <xs:restriction base="xs:QName">
1011         <xs:enumeration value="wsrm:UnknownSequence"/>
1012         <xs:enumeration value="wsrm:SequenceTerminated"/>
1013         <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1014         <xs:enumeration value="wsrm:MessageNumberRollover"/>
1015         <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1016     </xs:restriction>
1017 </xs:simpleType>
1018 <xs:complexType name="SequenceFaultType">
1019     <xs:sequence>
1020         <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1021         <xs:any namespace="##any" processContents="lax" minOccurs="0"
1022 maxOccurs="unbounded"/>
1023     </xs:sequence>
1024     <xs:anyAttribute namespace="##any" processContents="lax"/>
1025 </xs:complexType>
1026 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1027 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1028 <xs:element name="CreateSequenceResponse"
1029 type="wsrm:CreateSequenceResponseType"/>
1030 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1031 <xs:element name="CloseSequenceResponse"
1032 type="wsrm:CloseSequenceResponseType"/>
1033 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1034 <xs:complexType name="CreateSequenceType">
1035     <xs:sequence>
1036         <xs:element ref="wsrm:AcksTo"/>
1037         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1038         <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1039         <xs:any namespace="##other" processContents="lax" minOccurs="0"

```

```

1040 maxOccurs="unbounded">
1041     <xs:annotation>
1042         <xs:documentation>
1043             It is the authors intent that this extensibility be used to
1044 transfer a Security Token Reference as defined in WS-Security.
1045         </xs:documentation>
1046     </xs:annotation>
1047 </xs:any>
1048 </xs:sequence>
1049 <xs:anyAttribute namespace="##other" processContents="lax"/>
1050 </xs:complexType>
1051 <xs:complexType name="CreateSequenceResponseType">
1052     <xs:sequence>
1053         <xs:element ref="wsrm:Identifier"/>
1054         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1055         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1056         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1057 maxOccurs="unbounded"/>
1058     </xs:sequence>
1059     <xs:anyAttribute namespace="##other" processContents="lax"/>
1060 </xs:complexType>
1061 <xs:complexType name="CloseSequenceType">
1062     <xs:sequence>
1063         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1064 maxOccurs="unbounded"/>
1065     </xs:sequence>
1066     <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1067     <xs:anyAttribute namespace="##other" processContents="lax"/>
1068 </xs:complexType>
1069 <xs:complexType name="CloseSequenceResponseType">
1070     <xs:sequence>
1071         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1072 maxOccurs="unbounded"/>
1073     </xs:sequence>
1074     <xs:anyAttribute namespace="##other" processContents="lax"/>
1075 </xs:complexType>
1076 <xs:complexType name="TerminateSequenceType">
1077     <xs:sequence>
1078         <xs:element ref="wsrm:Identifier"/>
1079         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1080 maxOccurs="unbounded"/>
1081     </xs:sequence>
1082     <xs:anyAttribute namespace="##other" processContents="lax"/>
1083 </xs:complexType>
1084 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1085 <xs:complexType name="OfferType">
1086     <xs:sequence>
1087         <xs:element ref="wsrm:Identifier"/>
1088         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1089         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1090 maxOccurs="unbounded"/>
1091     </xs:sequence>
1092     <xs:anyAttribute namespace="##other" processContents="lax"/>
1093 </xs:complexType>
1094 <xs:complexType name="AcceptType">
1095     <xs:sequence>
1096         <xs:element ref="wsrm:AcksTo"/>
1097         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1098 maxOccurs="unbounded"/>
1099     </xs:sequence>
1100     <xs:anyAttribute namespace="##other" processContents="lax"/>
1101 </xs:complexType>

```

1102
1103
1104
1105
1106
1107
1108
1109
1110
1111

```
<xs:element name="Expires">  
  <xs:complexType>  
    <xs:simpleContent>  
      <xs:extension base="xs:duration">  
        <xs:anyAttribute namespace="##other" processContents="lax"/>  
      </xs:extension>  
    </xs:simpleContent>  
  </xs:complexType>  
</xs:element>  
</xs:schema>
```

1112
1113
1114
1115
1116
1117
1118
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1160
1161
1162

```
<xs:schema targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"  
  xmlns:xs="http://www.w3.org/2001/XMLSchema"  
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"  
  xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"  
  elementFormDefault="qualified" attributeFormDefault="unqualified"  
  <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"  
    schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing/">  
  <!-- Protocol Elements -->  
  <xs:complexType name="SequenceType">  
    <xs:sequence>  
      <xs:element ref="wsrm:Identifier"/>  
      <xs:element name="MessageNumber" type="xs:unsignedLong"/>  
      <xs:any namespace="##other" processContents="lax" minOccurs="0"  
maxOccurs="unbounded"/>  
    </xs:sequence>  
    <xs:anyAttribute namespace="##other" processContents="lax"/>  
  </xs:complexType>  
  <xs:element name="Sequence" type="wsrm:SequenceType"/>  
  <xs:element name="SequenceAcknowledgement">  
    <xs:complexType>  
      <xs:sequence>  
        <xs:element ref="wsrm:Identifier"/>  
        <xs:choice>  
          <xs:sequence>  
            <xs:choice>  
              <xs:element name="AcknowledgementRange" maxOccurs="unbounded">  
                <xs:complexType>  
                  <xs:sequence/>  
                  <xs:attribute name="Upper" type="xs:unsignedLong"  
use="required"/>  
                  <xs:attribute name="Lower" type="xs:unsignedLong"  
use="required"/>  
                <xs:anyAttribute namespace="##other" processContents="lax"/>  
              </xs:complexType>  
            </xs:element>  
            <xs:element name="None" minOccurs="0">  
              <xs:complexType>  
                <xs:sequence/>  
              </xs:complexType>  
            </xs:element>  
          </xs:choice>  
        <xs:element name="Final" minOccurs="0">  
          <xs:complexType>  
            <xs:sequence/>  
          </xs:complexType>  
        </xs:element>  
      </xs:sequence>  
      <xs:element name="Nack" type="xs:unsignedLong"  
maxOccurs="unbounded"/>  
    </xs:complexType>  
  </xs:element>  
</xs:choice>
```

```

1163 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1164 maxOccurs="unbounded"/>
1165 </xs:sequence>
1166 <xs:anyAttribute namespace="##other" processContents="lax"/>
1167 </xs:complexType>
1168 </xs:element>
1169 <xs:complexType name="AckRequestedType">
1170 <xs:sequence>
1171 <xs:element ref="wsrm:Identifier"/>
1172 -
1173 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1174 maxOccurs="unbounded"/>
1175 </xs:sequence>
1176 <xs:anyAttribute namespace="##other" processContents="lax"/>
1177 </xs:complexType>
1178 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1179 <xs:element name="Identifier">
1180 <xs:complexType>
1181 <xs:annotation>
1182 <xs:documentation>
1183 This type is for elements whose [children] is an anyURI and can have arbitrary
1184 attributes.
1185 </xs:documentation>
1186 </xs:annotation>
1187 <xs:simpleContent>
1188 <xs:extension base="xs:anyURI">
1189 <xs:anyAttribute namespace="##other" processContents="lax"/>
1190 </xs:extension>
1191 </xs:simpleContent>
1192 </xs:complexType>
1193 </xs:element>
1194 <!-- Fault Container and Codes -->
1195 <xs:simpleType name="FaultCodes">
1196 <xs:restriction base="xs:QName">
1197 <xs:enumeration value="wsrm:UnknownSequence"/>
1198 <xs:enumeration value="wsrm:SequenceTerminated"/>
1199 <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1200 <xs:enumeration value="wsrm:MessageNumberRollover"/>
1201 <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1202 </xs:restriction>
1203 </xs:simpleType>
1204 <xs:complexType name="SequenceFaultType">
1205 <xs:sequence>
1206 <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1207 <xs:any namespace="##any" processContents="lax" minOccurs="0"
1208 maxOccurs="unbounded"/>
1209 </xs:sequence>
1210 <xs:anyAttribute namespace="##any" processContents="lax"/>
1211 </xs:complexType>
1212 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1213 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1214 <xs:element name="CreateSequenceResponse"
1215 type="wsrm:CreateSequenceResponseType"/>
1216 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1217 <xs:element name="CloseSequenceResponse"
1218 type="wsrm:CloseSequenceResponseType"/>
1219 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1220 <xs:complexType name="CreateSequenceType">
1221 <xs:sequence>
1222 <xs:element ref="wsrm:AcksTo"/>
1223 <xs:element ref="wsrm:Expires" minOccurs="0"/>
1224 <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>

```

```

1225 -----<xs:any namespace="##other" processContents="lax" minOccurs="0"
1226 maxOccurs="unbounded">
1227 -----<xs:annotation>
1228 -----<xs:documentation>
1229 It is the authors intent that this extensibility be used to transfer a
1230 Security Token Reference as defined in WS-Security.
1231 </xs:documentation>
1232 </xs:annotation>
1233 </xs:any>
1234 </xs:sequence>
1235 <xs:anyAttribute namespace="##other" processContents="lax"/>
1236 </xs:complexType>
1237 <xs:complexType name="CreateSequenceResponseType">
1238 <xs:sequence>
1239 <xs:element ref="wsrm:Identifier"/>
1240 <xs:element ref="wsrm:Expires" minOccurs="0"/>
1241 <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1242 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1243 maxOccurs="unbounded"/>
1244 </xs:sequence>
1245 <xs:anyAttribute namespace="##other" processContents="lax"/>
1246 </xs:complexType>
1247 <xs:complexType name="CloseSequenceType">
1248 <xs:sequence>
1249 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1250 maxOccurs="unbounded"/>
1251 </xs:sequence>
1252 <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1253 <xs:anyAttribute namespace="##other" processContents="lax"/>
1254 </xs:complexType>
1255 <xs:complexType name="CloseSequenceResponseType">
1256 <xs:sequence>
1257 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1258 maxOccurs="unbounded"/>
1259 </xs:sequence>
1260 <xs:anyAttribute namespace="##other" processContents="lax"/>
1261 </xs:complexType>
1262 <xs:complexType name="TerminateSequenceType">
1263 <xs:sequence>
1264 <xs:element ref="wsrm:Identifier"/>
1265 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1266 maxOccurs="unbounded"/>
1267 </xs:sequence>
1268 <xs:anyAttribute namespace="##other" processContents="lax"/>
1269 </xs:complexType>
1270 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1271 <xs:complexType name="OfferType">
1272 <xs:sequence>
1273 <xs:element ref="wsrm:Identifier"/>
1274 <xs:element ref="wsrm:Expires" minOccurs="0"/>
1275 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1276 maxOccurs="unbounded"/>
1277 </xs:sequence>
1278 <xs:anyAttribute namespace="##other" processContents="lax"/>
1279 </xs:complexType>
1280 <xs:complexType name="AcceptType">
1281 <xs:sequence>
1282 <xs:element ref="wsrm:AcksTo"/>
1283 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1284 maxOccurs="unbounded"/>
1285 </xs:sequence>
1286 <xs:anyAttribute namespace="##other" processContents="lax"/>
1287 </xs:complexType>

```

```
1288 <xs:element name="Expires">
1289 <xs:complexType>
1290 <xs:simpleContent>
1291 <xs:extension base="xs:duration">
1292 <xs:anyAttribute namespace="##other" processContents="lax"/>
1293 </xs:extension>
1294 </xs:simpleContent>
1295 </xs:complexType>
1296 </xs:element>
1297 </xs:schema>
```

1298 B. Message Examples

1299 B.1 Create Sequence

1300 Create Sequence

```
1301 <?xml version="1.0" encoding="UTF-8"?>
1302 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1303 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1304 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1305   <S:Header>
1306     <wsa:MessageID>
1307       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1308     </wsa:MessageID>
1309     <wsa:To>http://example.com/serviceB/123</wsa:To>
1310     <wsa:Action>http://docs.oasis-open.org/ws-
1311 rx/wsmr/200510/CreateSequence</wsa:Action>
1312     <wsa:ReplyTo>
1313       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1314     </wsa:ReplyTo>
1315   </S:Header>
1316   <S:Body>
1317     <wsmr:CreateSequence>
1318       <wsa:AcksTo>
1319         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1320       </wsa:AcksTo>
1321     </wsmr:CreateSequence>
1322   </S:Body>
1323 </S:Envelope>
```

1324 Create Sequence Response

```
1325 <?xml version="1.0" encoding="UTF-8"?>
1326 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1327 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1328 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1329   <S:Header>
1330     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1331     <wsa:RelatesTo>
1332       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1333     </wsa:RelatesTo>
1334     <wsa:Action>
1335       http://docs.oasis-open.org/ws-rx/wsmr/200510/CreateSequenceResponse
1336     </wsa:Action>
1337   </S:Header>
1338   <S:Body>
1339     <wsmr:CreateSequenceResponse>
1340       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1341     </wsmr:CreateSequenceResponse>
1342   </S:Body>
1343 </S:Envelope>
```

1344 B.2 Initial Transmission

1345 The following example WS-ReliableMessaging headers illustrate the message exchange in the above
1346 figure. The three messages have the following headers; the third message is identified as the last
1347 message in the sequence:

1348 **Message 1**

```
1349 <?xml version="1.0" encoding="UTF-8"?>
1350 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1351 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1352 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1353   <S:Header>
1354     <wsa:MessageID>
1355       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1356     </wsa:MessageID>
1357     <wsa:To>http://example.com/serviceB/123</wsa:To>
1358     <wsa:From>
1359       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1360     </wsa:From>
1361     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1362     <wsmr:Sequence>
1363       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1364       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1365     </wsmr:Sequence>
1366   </S:Header>
1367   <S:Body>
1368     <!-- Some Application Data -->
1369   </S:Body>
1370 </S:Envelope>
```

1371 **Message 2**

```
1372 <?xml version="1.0" encoding="UTF-8"?>
1373 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1374 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1375 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1376   <S:Header>
1377     <wsa:MessageID>
1378       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1379     </wsa:MessageID>
1380     <wsa:To>http://example.com/serviceB/123</wsa:To>
1381     <wsa:From>
1382       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1383     </wsa:From>
1384     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1385     <wsmr:Sequence>
1386       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1387       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1388     </wsmr:Sequence>
1389   </S:Header>
1390   <S:Body>
1391     <!-- Some Application Data -->
1392   </S:Body>
1393 </S:Envelope>
```

1394 **Message 3**

```
1395 <?xml version="1.0" encoding="UTF-8"?>
1396 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1397 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1398 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1399   <S:Header>
1400     <wsa:MessageID>
1401       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1402     </wsa:MessageID>
1403     <wsa:To>http://example.com/serviceB/123</wsa:To>
1404     <wsa:From>
1405       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1406 </wsa:From>
1407 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1408 <wsrm:Sequence>
1409 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1410 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1411 </wsrm:Sequence>
1412 <wsrm:AckRequested>
1413 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1414 </wsrm:AckRequested>
1415 </S:Header>
1416 <S:Body>
1417 <!-- Some Application Data -->
1418 </S:Body>
1419 </S:Envelope>

```

1420 B.3 First Acknowledgement

1421 Message number 2 has not been received by the RM Destination due to some transmission error so it
1422 responds with an acknowledgement for messages 1 and 3:

```

1423 <?xml version="1.0" encoding="UTF-8"?>
1424 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1425 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1426 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1427 <S:Header>
1428 <wsa:MessageID>
1429 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1430 </wsa:MessageID>
1431 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1432 <wsa:From>
1433 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1434 </wsa:From>
1435 <wsa:Action>
1436 http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement
1437 </wsa:Action>
1438 <wsrm:SequenceAcknowledgement>
1439 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1440 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1441 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1442 </wsrm:SequenceAcknowledgement>
1443 </S:Header>
1444 <S:Body/>
1445 </S:Envelope>

```

1446 B.4 Retransmission

1447 The RM Sourcediscovers that message number 2 was not received so it resends the message and
1448 requests an acknowledgement:

```

1449 <?xml version="1.0" encoding="UTF-8"?>
1450 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1451 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1452 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1453 <S:Header>
1454 <wsa:MessageID>
1455 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1456 </wsa:MessageID>
1457 <wsa:To>http://example.com/serviceB/123</wsa:To>
1458 <wsa:From>
1459 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1460 </wsa:From>

```

```

1461 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1462 <wsrm:Sequence>
1463   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1464   <wsrm:MessageNumber>2</wsrm:MessageNumber>
1465 </wsrm:Sequence>
1466 <wsrm:AckRequested>
1467   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1468 </wsrm:AckRequested>
1469 </S:Header>
1470 <S:Body>
1471   <!-- Some Application Data -->
1472 </S:Body>
1473 </S:Envelope>

```

1474 B.5 Termination

1475 The RM Destination now responds with an acknowledgement for the complete sequence which can then
1476 be terminated:

```

1477 <?xml version="1.0" encoding="UTF-8"?>
1478 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1479 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1480 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1481   <S:Header>
1482     <wsa:MessageID>
1483       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1484     </wsa:MessageID>
1485     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1486     <wsa:From>
1487       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1488     </wsa:From>
1489     <wsa:Action>
1490       http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement
1491     </wsa:Action>
1492     <wsrm:SequenceAcknowledgement>
1493       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1494       <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1495     </wsrm:SequenceAcknowledgement>
1496   </S:Header>
1497   <S:Body/>
1498 </S:Envelope>

```

1499 Terminate Sequence

```

1500 <?xml version="1.0" encoding="UTF-8"?>
1501 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1502 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1503 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1504   <S:Header>
1505     <wsa:MessageID>
1506       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1507     </wsa:MessageID>
1508     <wsa:To>http://example.com/serviceB/123</wsa:To>
1509     <wsa:Action>
1510       http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence
1511     </wsa:Action>
1512     <wsa:From>
1513       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1514     </wsa:From>
1515   </S:Header>
1516   <S:Body>
1517     <wsrm:TerminateSequence>

```

```
1518     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1519     </wsrm:TerminateSequence>
1520     </S:Body>
1521 </S:Envelope>
```

1522 C. WSDL

1523 The non-normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1524 <http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd/wsrn-1.1-wsd-200510.wsd>

1525 The following non-normative copy is provided for reference.

```
1526 <?xml version="1.0" encoding="utf-8"?>
1527 <wSDL:definitions xmlns:wSDL="http://schemas.xmlsoap.org/wSDL/"
1528 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1529 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1530 xmlns:rm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1531 xmlns:tns="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd"
1532 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd">
1533   <wSDL:types>
1534     <xs:schema
1535       <xs:import namespace="http://docs.oasis-open.org/ws-
1536 rx/wsrn/200510" schemaLocation="http://docs.oasis-open.org/ws-
1537 rx/wsrn/200510/wsrn-1.1-schema-200510.xsd"/>
1538     </xs:schema>
1539   </wSDL:types>
1540   <wSDL:message name="CreateSequence">
1541     <wSDL:part name="create" element="rm:CreateSequence"/>
1542   </wSDL:message>
1543   <wSDL:message name="CreateSequenceResponse">
1544     <wSDL:part name="createResponse"
1545 element="rm:CreateSequenceResponse"/>
1546   </wSDL:message>
1547   <wSDL:message name="CloseSequence">
1548     <wSDL:part name="close" element="rm:CloseSequence"/>
1549   </wSDL:message>
1550   <wSDL:message name="CloseSequenceResponse">
1551     <wSDL:part name="closeResponse"
1552 element="rm:CloseSequenceResponse"/>
1553   </wSDL:message>
1554   <wSDL:message name="TerminateSequence">
1555     <wSDL:part name="terminate" element="rm:TerminateSequence"/>
1556   </wSDL:message>
1557   <wSDL:portType name="SequenceAbstractPortType">
1558     <wSDL:operation name="CreateSequence">
1559       <wSDL:input message="tns:CreateSequence"
1560 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/CreateSequence"/>
1561       <wSDL:output message="tns:CreateSequenceResponse"
1562 wsa:Action="http://docs.oasis-open.org/ws-
1563 rx/wsrn/200510/CreateSequenceResponse"/>
1564     </wSDL:operation>
1565     <wSDL:operation name="CloseSequence">
1566       <wSDL:input message="tns:CloseSequence"
1567 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/CloseSequence"/>
1568       <wSDL:output message="tns:CloseSequenceResponse"
1569 wsa:Action="http://docs.oasis-open.org/ws-
1570 rx/wsrn/200510/CloseSequenceResponse"/>
1571     </wSDL:operation>
1572     <wSDL:operation name="TerminateSequence">
1573       <wSDL:input message="tns:TerminateSequence"
1574 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence"/>
1575     </wSDL:operation>
1576   </wSDL:portType>
```

```

1577 </wsdl:definitions>
1578 <del><wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1579 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1580 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1581 xmlns:rm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1582 xmlns:tns="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsdl"
1583 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsdl">
1584 <del><wsdl:types>
1585 <del><xs:schema
1586 <del><xs:import namespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1587 schemaLocation="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1.xsd"/>
1588 <del></xs:schema>
1589 <del></wsdl:types>
1590 <del><wsdl:message name="CreateSequence">
1591 <del><wsdl:part name="create" element="rm:CreateSequence"/>
1592 <del></wsdl:message>
1593 <del><wsdl:message name="CreateSequenceResponse">
1594 <del><wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1595 <del></wsdl:message>
1596 <del><wsdl:message name="CloseSequence">
1597 <del><wsdl:part name="close" element="rm:CloseSequence"/>
1598 <del></wsdl:message>
1599 <del><wsdl:message name="CloseSequenceResponse">
1600 <del><wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1601 <del></wsdl:message>
1602 <del><wsdl:message name="TerminateSequence">
1603 <del><wsdl:part name="terminate" element="rm:TerminateSequence"/>
1604 <del></wsdl:message>
1605 <del><wsdl:portType name="SequenceAbstractPortType">
1606 <del><wsdl:operation name="CreateSequence">
1607 <del><wsdl:input message="tns:CreateSequence" wsa:Action="http://docs.oasis-
1608 open.org/ws-rx/wsrn/200510/CreateSequence"/>
1609 <del><wsdl:output message="tns:CreateSequenceResponse"
1610 wsa:Action="http://docs.oasis-open.org/ws-
1611 rx/wsrn/200510/CreateSequenceResponse"/>
1612 <del></wsdl:operation>
1613 <del><wsdl:operation name="CloseSequence">
1614 <del><wsdl:input name="tns:CloseSequence" wsa:Action="http://docs.oasis-
1615 open.org/ws-rx/wsrn/200510/CloseSequence"/>
1616 <del><wsdl:output name="tns:CloseSequenceResponse"
1617 wsa:Action="http://docs.oasis-open.org/ws-
1618 rx/wsrn/200510/CloseSequenceResponse"/>
1619 <del></wsdl:operation>
1620 <del><wsdl:operation name="TerminateSequence">
1621 <del><wsdl:input message="tns:TerminateSequence"
1622 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence"/>
1623 <del></wsdl:operation>
1624 <del></wsdl:portType>
1625 <del></wsdl:definitions>

```

1626 **D. Acknowledgments**

1627 This document is based on initial contribution to OASIS WS-RX Technical Committee by the following
1628 authors:

1629 Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft, Doug Davis, IBM,
1630 Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM, Mary Ann Hondo,
1631 IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David Langworthy, Microsoft
1632 (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft, Steve Lucco, Microsoft,
1633 Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham, BEA, David Orchard,
1634 BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John Shewchuk, Microsoft, Tony
1635 Storey, IBM.

1636 The following individuals have provided invaluable input into the initial contribution:

1637 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,
1638 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,
1639 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,
1640 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin
1641 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,
1642 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger
1643 Wolter, Microsoft.

1644 The following individuals were members of the committee during the development of this specification:

1645 *TBD*

E. Revision History

Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.
ws-02	2005-07-21	Doug Davis	I011 (PTOS) added
wd-02	2005-08-16	Anish Karmarkar	Trivial editorial changes
ws-03	2005-09-15	Doug Davis	I019 and i028 (CloseSeq) added
wd-05	2005-09-26	Gilbert Pilz	i005 (Source resend of nacks messages when ack already received) added.
wd-05	2005-09-27	Doug Davis	i027 (InOrder delivery assurance spanning multiple sequences) added
wd-05	2005-09-27	Doug Davis	i020 (Semantics of "At most once" Delivery Assurance) added
wd-05	2005-09-27	Doug Davis	i034 (Fault while processing a piggy-backed RM header) added
wd-05	2005-09-27	Doug Davis	i033 (Processing model of NACKs) added
wd-05	2005-09-27	Doug Davis	i031 (AckRequested schema inconsistency) added
wd-05	2005-09-27	Doug Davis	i025 (SeqAck/None) added
wd-05	2005-09-27	Doug Davis	i029 (Remove dependency on WS-Security) added
wd-05	2005-09-27	Doug Davis	i039 (What does 'have a mU attribute' mean) added
wd-05	2005-09-27	Doug Davis	i040 (Change 'optiona'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrn/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrn:FaultCode element)
wd-06	2005-11-02	Gilbert Pilz	Start wd-06 by changing title page from cd-01.
wd-06	2005-11-03	Gilbert Pilz	i047 (Reorder spec sections)
wd-07	2005-11-17	Gilbert Pilz	Start wd-07
wd-07	2005-11-28	Doug Davis	i071 – except for period in Appendix headings
wd-07	2005-11-28	Doug Davis	i10
wd-07	2005-11-28	Doug Davis	i030
wd-07	2005-11-28	Doug Davis	i037
wd-07	2005-11-28	Doug Davis	i038
wd-07	2005-11-28	Doug Davis	i041
wd-07	2005-11-28	Doug Davis	i043
wd-07	2005-11-28	Doug Davis	i044

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.
Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.
ws-02	2005-07-21	Doug Davis	i011 (PTOS) added
wd-02	2005-08-16	Anish Karmarkar	Trivial editorial changes
ws-03	2005-09-15	Doug Davis	i019 and i028 (CloseSeq) added
wd-05	2005-09-26	Gilbert Pilz	i005 (Source resend of nacks messages when ack already received) added.
wd-05	2005-09-27	Doug Davis	i027 (InOrder delivery assurance spanning multiple sequences) added
wd-05	2005-09-27	Doug Davis	i020 (Semantics of "At most once" Delivery Assurance) added
wd-05	2005-09-27	Doug Davis	i034 (Fault while processing a piggy-backed RM header) added
wd-05	2005-09-27	Doug Davis	i033 (Processing model of NACKs) added
wd-05	2005-09-27	Doug Davis	i031 (AckRequested schema inconsistency) added
wd-05	2005-09-27	Doug Davis	i025 (SeqAck/None) added
wd-05	2005-09-27	Doug Davis	i029 (Remove dependency on WS-Security) added
wd-05	2005-09-27	Doug Davis	i039 (What does 'have a mU attribute' mean) added
wd-05	2005-09-27	Doug Davis	i040 (Change 'optional'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrm/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrm: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

Rev	Date	By Whom	What
wd-07	2005-11-28	Doug Davis	i041
wd-07	2005-11-28	Doug Davis	i043
wd-07	2005-11-28	Doug Davis	i044
wd-07	2005-11-28	Doug Davis	i048
wd-07	2005-11-28	Doug Davis	i051
wd-07	2005-11-28	Doug Davis	i053
wd-07	2005-11-28	Doug Davis	i059
wd-07	2005-11-28	Doug Davis	i062
wd-07	2005-11-28	Doug Davis	i063
wd-07	2005-11-28	Doug Davis	i065
wd-07	2005-11-28	Doug Davis	i067
wd-07	2005-11-28	Doug Davis	i068
wd-07	2005-11-28	Doug Davis	i069
wd-07	2005-11-28	Doug Davis	Fix bulleted list (#2) in section 2.3
wd-07	2005-11-29	Gilbert Pilz	i074 (Use of [tcShortName] in artifact locations namespaces, etc)
wd-07	2005-11-29	Gilbert Pilz	i071 — Fixed styles and formatting for TOC. Fixed styles of the appendix headings.
wd-07	2005-11-30	Doug Davis	Removed dup definition of "Receive"
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

1647 F. Notices

1648 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
1649 might be claimed to pertain to the implementation or use of the technology described in this document or
1650 the extent to which any license under such rights might or might not be available; neither does it represent
1651 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to
1652 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made
1653 available for publication and any assurances of licenses to be made available, or the result of an attempt
1654 made to obtain a general license or permission for the use of such proprietary rights by implementors or
1655 users of this specification, can be obtained from the OASIS Executive Director.

1656 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or
1657 other proprietary rights which may cover technology that may be required to implement this specification.
1658 Please address the information to the OASIS Executive Director.

1659 Copyright (C) OASIS Open (2005). All Rights Reserved.

1660 This document and translations of it may be copied and furnished to others, and derivative works that
1661 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and
1662 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and
1663 this paragraph are included on all such copies and derivative works. However, this document itself may
1664 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as
1665 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights
1666 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it
1667 into languages other than English.

1668 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
1669 or assigns.

1670 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1671 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
1672 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR
1673 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.