



1 Web Services Reliable Messaging 2 (WS-ReliableMessaging)

3 Committee Draft 04, August 11, 2006

4 Document identifier:

5 wsrn-1.1-spec-cd-04

6 Location:

7 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-spec-cd-04.pdf>

8 Editors:

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

14 Contributors:

15 See the Acknowledgments (Appendix E).

16 Abstract:

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

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

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

31 Status:

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

42 Table of Contents

43	1 Introduction.....	4
44	1.1 Notational Conventions.....	4
45	1.2 Namespace.....	5
46	1.3 Compliance.....	5
47	2 Reliable Messaging Model.....	6
48	2.1 Glossary.....	6
49	2.2 Protocol Preconditions.....	7
50	2.3 Protocol Invariants.....	7
51	2.4 Example Message Exchange.....	8
52	3 RM Protocol Elements.....	10
53	3.1 Considerations on the Use of Extensibility Points.....	10
54	3.2 Considerations on the Use of "Piggy-Backing".....	10
55	3.3 Composition with WS-Addressing.....	10
56	3.4 Sequence Creation.....	10
57	3.5 Closing A Sequence.....	15
58	3.6 Sequence Termination.....	16
59	3.7 Sequences.....	18
60	3.8 Request Acknowledgement.....	19
61	3.9 Sequence Acknowledgement.....	20
62	3.10 MakeConnection.....	22
63	3.11 MessagePending.....	24
64	4 Faults.....	25
65	4.1 SequenceFault Element.....	26
66	4.2 Sequence Terminated.....	27
67	4.3 Unknown Sequence.....	27
68	4.4 Invalid Acknowledgement.....	28
69	4.5 Message Number Rollover.....	28
70	4.6 Create Sequence Refused.....	29
71	4.7 Sequence Closed.....	29
72	4.8 WSRM Required.....	30
73	4.9 Unsupported Selection.....	30
74	5 Security Threats and Countermeasures.....	32
75	5.1 Threats and Countermeasures.....	32
76	5.1.1 Integrity Threats.....	32
77	5.1.1.1 Countermeasures.....	32
78	5.1.2 Resource Consumption Threats.....	33
79	5.1.2.1 Countermeasures.....	33

80	5.1.3 Sequence Spoofing Threats.....	33
81	5.1.3.1 Sequence Hijacking.....	33
82	5.1.3.2 Countermeasures.....	33
83	5.2 Security Solutions and Technologies.....	34
84	5.2.1 Transport Layer Security.....	34
85	5.2.1.1 Model.....	34
86	5.2.1.2 Countermeasure Implementation.....	35
87	5.2.2 SOAP Message Security.....	36
88	5.2.2.1 Model.....	36
89	5.2.2.2 Countermeasure Implementation.....	36
90	6 Securing Sequences.....	38
91	6.1 Securing Sequences Using WS-Security.....	38
92	6.2 Securing Sequences Using SSL/TLS.....	39
93	7 References.....	41
94	7.1 Normative.....	41
95	7.2 Non-Normative.....	41
96	Appendix A. Schema.....	43
97	Appendix B. WSDL.....	48
98	Appendix C. Message Examples.....	50
99	Appendix C.1 Create Sequence.....	50
100	Appendix C.2 Initial Transmission.....	50
101	Appendix C.3 First Acknowledgement.....	52
102	Appendix C.4 Retransmission.....	52
103	Appendix C.5 Termination.....	53
104	Appendix C.6 MakeConnection.....	54
105	Appendix D. State Tables.....	58
106	Appendix E. Acknowledgments.....	63
107	Appendix F. Revision History.....	64
108	Appendix G. Notices.....	70

109 1 Introduction

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

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

119 1.1 Notational Conventions

120 The keywords "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD
121 NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described
122 in RFC 2119 [KEYWORDS].

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

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

138 Elements and Attributes defined by this specification are referred to in the text of this document using
139 XPath 1.0 [XPATH 1.0] expressions. Extensibility points are referred to using an extended version of this
140 syntax:

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

147 1.2 Namespace

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

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

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

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

154 Table 1

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

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

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

158 1.3 Compliance

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

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

2 Reliable Messaging Model

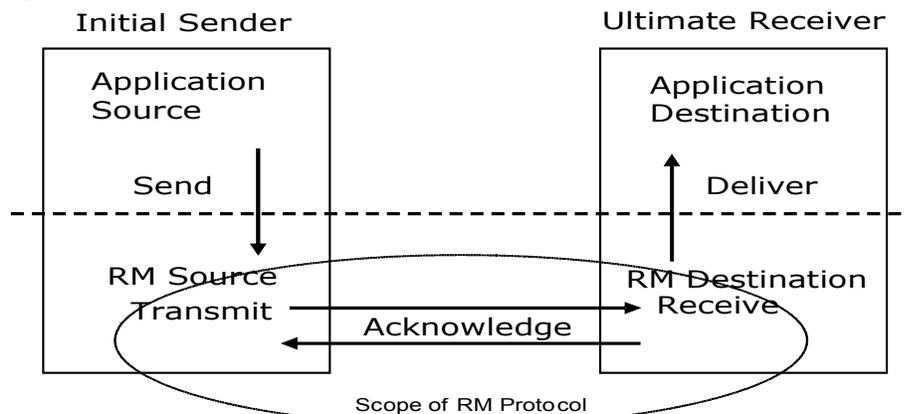
165

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

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

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

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



191 Figure 1: Reliable Messaging Model

2.1 Glossary

192

193 The following definitions are used throughout this specification:

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

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

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

200 **Acknowledgement Request:** A message containing a `AckRequested` header. Acknowledgement
201 Requests may or may not contain a SOAP body.

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

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

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

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

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

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

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

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

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

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

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

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

219 **2.2 Protocol Preconditions**

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

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

229 **2.3 Protocol Invariants**

230 During the lifetime of a `Sequence`, two invariants are **REQUIRED** for correctness:

- 231 • The RM Source MUST assign each message within a Sequence a message number (defined
- 232 below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers
- 233 MUST be assigned in the same order in which messages are sent by the Application Source.
- 234 • Within every Acknowledgement Message it issues, the RM Destination MUST include one or more
- 235 `AcknowledgementRange` child elements that contain, in their collective ranges, the message
- 236 number of every message accepted by the RM Destination. The RM Destination MUST exclude, in
- 237 the `AcknowledgementRange` elements, the message numbers of any messages it has not
- 238 accepted.

239 2.4 Example Message Exchange

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

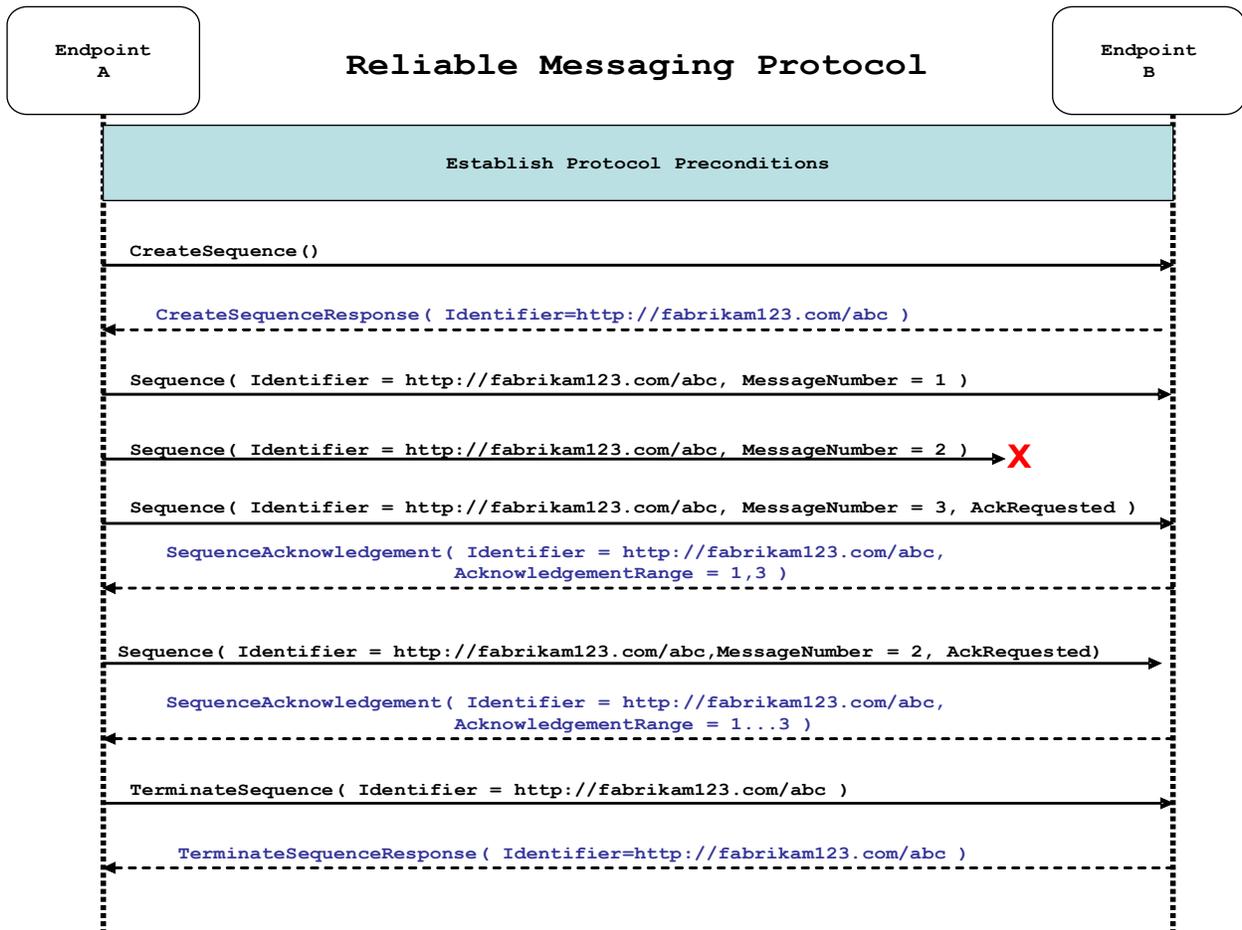


Figure 2: The WS-ReliableMessaging Protocol

- 241 1. The protocol preconditions are established. These include policy exchange, endpoint resolution,
- 242 and establishing trust.
- 243 2. The RM Source requests creation of a new Sequence.
- 244 3. The RM Destination creates a new Sequence and returns its unique identifier.
- 245 4. The RM Source begins Transmitting messages in the Sequence beginning with MessageNumber 1.
- 246 In the figure above, the RM Source sends 3 messages in the Sequence.

247 5. The 2nd message in the Sequence is lost in transit.

248 6. The 3rd message is the last in this Sequence and the RM Source includes an `AckRequested`
249 header to ensure that it gets a timely `SequenceAcknowledgement` for the Sequence.

250 7. The RM Destination acknowledges receipt of message numbers 1 and 3 as a result of receiving the
251 RM Source's `AckRequested` header.

252 8. The RM Source retransmits the unacknowledged message with `MessageNumber 2`. This is a new
253 message from the perspective of the underlying transport, but it has the same `Sequence Identifier`
254 and `MessageNumber` so the RM Destination can recognize it as a duplicate of the earlier message,
255 in case the original and retransmitted messages are both Received. The RM Source includes an
256 `AckRequested` header in the retransmitted message so the RM Destination will expedite an
257 acknowledgement.

258 9. The RM Destination Receives the second transmission of the message with `MessageNumber 2`
259 and acknowledges receipt of message numbers 1, 2, and 3.

260 10. The RM Source Receives this Acknowledgement and sends a `TerminateSequence` message to the
261 RM Destination indicating that the Sequence is completed. The `TerminateSequence` message
262 indicates that message number 3 is the last message in the Sequence. The RM Source then and
263 reclaims any resources associated with the Sequence.

264 11. The RM Destination Receives the `TerminateSequence` message indicating that the RM Source will
265 not be sending any more messages. The RM Destination sends a `TerminateSequenceResponse`
266 message to the RM Source and and reclaims any resources associated with the Sequence.

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

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

281 **3 RM Protocol Elements**

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

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

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

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

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

294 **3.3 Composition with WS-Addressing**

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

- 297 1. When an Endpoint generates a message that carries an RM protocol element, that is defined in
298 section 3 below, in the body of a SOAP envelope that Endpoint MUST include in that envelope a
299 `wsa:Action` SOAP header block whose value is an IRI that is a concatenation of the WS-RM
300 namespace URI, followed by a "/", followed by the value of the local name of the child element of
301 the SOAP body. For example, for a Sequence creation request message as described in section
302 3.1 below, the value of the `wsa:Action` IRI would be:

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

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

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

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

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

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

312 **3.4 Sequence Creation**

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

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

320 The following exemplar defines the CreateSequence syntax:

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

335 /wsrm:CreateSequence

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

340 /wsrm:CreateSequence/wsrm:AcksTo

341 The RM Source MUST include this element in any CreateSequence message it sends. This element is of
342 type wsa:EndpointReferenceType (as specified by WS-Addressing). It specifies the endpoint
343 reference to which messages containing SequenceAcknowledgement header blocks and faults related
344 to the created Sequence are to be sent, unless otherwise noted in this specification (for example, see
345 Section 3.2).

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

350 /wsrm:CreateSequence/wsrm:Expires

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

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

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

358 /wsrm:CreateSequence/wsrm:Offer

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

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

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

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

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

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

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

372 Implementations MUST NOT use an endpoint reference in the Endpoint element that would prevent the
373 sending of Sequence Lifecycle Message, Sequence Traffic Message, etc. For example, using the WS-
374 Addressing "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM
375 Destination to ever send Sequence Lifecycle Messages (e.g. `TerminateSequence`) to the RM Source
376 for the Offered Sequence. Implementations MAY use the WS-RM anonymous URI template and doing so
377 implies that messages will be retrieved using a mechanism such as the `MakeConnection` message (see
378 section 3.7).

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

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

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

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

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

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

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

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

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

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

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

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

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

403 /wsmr:CreateSequence/{any}

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

406 /wsmr:CreateSequence/@{any}

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

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

413 The following exemplar defines the `CreateSequenceResponse` syntax:

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

426 /wsmr:CreateSequenceResponse

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

430 /wsmr:CreateSequenceResponse/wsmr:Identifier

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

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

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

437 /wsmr:CreateSequenceResponse/wsmr:Expires

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

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

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

449 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`

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

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

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

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

460 `/wsrm:CreateSequenceResponse/wsrm:Accept`

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

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

466 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

467 The RM Destination MUST include this element, of type `wsa:EndpointReferenceType` (as specified
468 by WS-Addressing). It specifies the endpoint reference to which messages containing
469 `SequenceAcknowledgement` header blocks and faults related to the created Sequence are to be sent,
470 unless otherwise noted in this specification (for example, see Section 3.2).

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

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

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

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

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

481 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

487 3.5 Closing A Sequence

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

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

502 In order to allow the RM Destination to determine if it has received all of the messages in a Sequence, the
503 RM Source MUST include a `LastMsgNumber` element in the `CloseSequence` message. The
504 `LastMsgNumber` element specifies the highest assigned message number of all the Sequence Traffic
505 Messages for the Sequence being closed. The RM Destination can use this information, for example, to
506 implement the behavior indicated by
507 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`.

508 To successfully complete a Sequence, the RM Source MUST send either a `CloseSequence` or a
509 `TerminateSequence` message to the RM Destination.

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

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

519 The following exemplar defines the `CloseSequence` syntax:

```
520 <wsrm:CloseSequence ...>  
521   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
522   ...  
523 </wsrm:CloseSequence>
```

524 `/wsrm:CloseSequence`

525 This element is sent by an RM Source to indicate that the RM Destination MUST NOT accept any new
526 messages for this Sequence. A `SequenceClosed` fault MUST be generated by the RM Destination when it
527 Receives a message for a Sequence that is already closed.

528 `/wsrm:CloseSequence/wsrm:Identifier`

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

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

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

535 /wsmr:CloseSequence/wsmr:LastMsgNumber

536 The RM Source MUST include this element in any CloseSequence messages it sends. The RM Source
537 MUST set the value of this element to the highest assigned MessageNumber of any Sequence Traffic
538 Message for the Sequence identified in this CloseSequence message.

539 /wsmr:CloseSequence/{any}

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

542 /wsmr:CloseSequence@{any}

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

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

548 The following exemplar defines the CloseSequenceResponse syntax:

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

554 /wsmr:CloseSequenceResponse

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

557 /wsmr:CloseSequenceResponse/wsmr:Identifier

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

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

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

564 /wsmr:CloseSequenceResponse/{any}

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

567 /wsmr:CloseSequenceResponse@{any}

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

570 3.6 Sequence Termination

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

578 In order to allow the RM Destination to determine if it has received all of the messages in a Sequence, the
579 RM Source MUST include a `LastMsgNumber` element in the `TerminateSequence` message. The
580 `LastMsgNumber` element specifies the highest assigned message number of all the Sequence Traffic
581 Messages for the Sequence being terminated. The RM Destination can use this information, for example,
582 to implement the behavior indicated by

583 `/wsrm:CreateSequenceResponse/wsrm:IncompleteSequenceBehavior`. The value of the
584 `LastMsgNumber` element in the `TerminateSequence` message must be equal to the value of the
585 `LastMsgNumber` element in any `CloseSequence` message(s) sent for the same Sequence.

586 To successfully complete a Sequence, the RM Source MUST send either a `CloseSequence` or a
587 `TerminateSequence` message to the RM Destination.

588 The following exemplar defines the `TerminateSequence` syntax:

```
589 <wsrm:TerminateSequence ...>  
590   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
591   <wsrm>LastMsgNumber> wsrm:MessageNumberType </wsrm>LastMsgNumber>  
592   ...  
593 </wsrm:TerminateSequence>
```

594 `/wsrm:TerminateSequence`

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

600 `/wsrm:TerminateSequence/wsrm:Identifier`

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

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

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

607 `/wsrm:TerminateSequence/wsrm>LastMsgNumber`

608 The RM Source MUST include this element in any `TerminateSequence` messages it sends. The RM
609 Source MUST set the value of this element to the highest assigned `MessageNumber` of any Sequence
610 Traffic Message for the Sequence identified in this `TerminateSequence` message.

611 `/wsrm:TerminateSequence/{any}`

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

614 `/wsrm:TerminateSequence/@{any}`

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

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

620 The following exemplar defines the `TerminateSequenceResponse` syntax:

```
621 <wsrm:TerminateSequenceResponse ...>  
622   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
623   ...  
624 </wsrm:TerminateSequenceResponse>
```

625 `/wsrm:TerminateSequenceResponse`

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

629 `/wsrm:TerminateSequenceResponse/wsrm:Identifier`

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

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

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

636 `/wsrm:TerminateSequenceResponse/{any}`

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

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

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

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

645 **3.7 Sequences**

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

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

653 A following exemplar defines its syntax:

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

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

660 /wsm:Sequence

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

667 /wsm:Sequence/wsm:Identifier

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

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

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

674 /wsm:Sequence/wsm:MessageNumber

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

679 /wsm:Sequence/{any}

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

682 /wsm:Sequence/@{any}

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

685 The following example illustrates a Sequence header block.

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

690 3.8 Request Acknowledgement

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

693 The RM Source MAY request an Acknowledgement Message from the RM Destination at any time by
694 including an `AckRequested` header block in any message targeted to the RM Destination. An RM
695 Destination that Receives a message that contains an `AckRequested` header block MUST send a
696 message containing a `SequenceAcknowledgement` header block to the `AcksTo` endpoint reference
697 (see Section 3.1) for a known Sequence or else generate an `UnknownSequence` fault. If a non-
698 `mustUnderstand` fault occurs when processing an RM header that was piggy-backed on another
699 message, a fault MUST be generated, but the processing of the original message MUST NOT be
700 affected. It is RECOMMENDED that the RM Destination return a `AcknowledgementRange` or `None`
701 element instead of a `Nack` element (see Section 3.6).

702 The following exemplar defines its syntax:

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

707 `/wsm:AckRequested`

708 This element requests an Acknowledgement for the identified Sequence.

709 `/wsm:AckRequested/wsm:Identifier`

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

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

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

716 `/wsm:AckRequested/{any}`

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

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

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

722 **3.9 Sequence Acknowledgement**

723 The RM Destination informs the RM Source of successful message receipt using a
724 `SequenceAcknowledgement` header block. The RM Destination MAY Transmit the
725 `SequenceAcknowledgement` header block independently or it MAY include the
726 `SequenceAcknowledgement` header block on any message targeted to the `AcksTo` EPR.
727 Acknowledgements can be explicitly requested using the `AckRequested` directive (see Section 3.5). If a
728 non-`mustUnderstand` fault occurs when processing an RM header that was piggy-backed on another
729 message, a fault MUST be generated, but the processing of the original message MUST NOT be
730 affected.

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

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

735 anonymous IRI as the address of the `AcksTo` EPR, the RM Destination MUST Transmit any
736 `SequenceAcknowledgement` headers for the created Sequence in a SOAP envelope to be Transmitted
737 on the protocol binding-specific channel. Such a channel is provided by the context of a Received
738 message containing a SOAP envelope that contains a `Sequence` header block and/or a `AckRequested`
739 header block for that same Sequence identifier.

740 The following exemplar defines its syntax:

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

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

753 `/wsrm:SequenceAcknowledgement`

754 This element contains the Sequence Acknowledgement information.

755 `/wsrm:SequenceAcknowledgement/wsrm:Identifier`

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

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

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

764 `/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange`

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

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

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

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

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

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

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

779 /wsmr:SequenceAcknowledgement/wsmr:None

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

784 /wsmr:SequenceAcknowledgement/wsmr:Final

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

791 /wsmr:SequenceAcknowledgement/wsmr:Nack

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

801 /wsmr:SequenceAcknowledgement/{any}

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

804 /wsmr:SequenceAcknowledgement/@{any}

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

807 The following examples illustrate `SequenceAcknowledgement` elements:

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

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

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

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

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

```
822 <wsrm:SequenceAcknowledgement>  
823   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
824   <wsrm:Nack>3</wsrm:Nack>  
825 </wsrm:SequenceAcknowledgement>
```

826 3.10 MakeConnection

827 When an Endpoint is not directly addressable (e.g. behind a firewall or not able to allow incoming
828 connections), an anonymous URI in the EPR address property can indicate such an Endpoint. The WS-
829 Addressing anonymous URI is one such anonymous URI. This specification defines a URI template (the
830 WS-RM anonymous URI) which may be used to uniquely identify anonymous Endpoints.

```
831 http://docs.oasis-open.org/ws-rx/wsrn/200608/anonymous?id={uuid}
```

832 This URI template in an EPR indicates a protocol-specific back-channel will be established through a
833 mechanism such as `MakeConnection`, defined below. When using this URI template, “{uuid}” MUST be
834 replaced by a UUID value as defined by RFC4122[UUID]. This UUID value uniquely distinguishes the
835 Endpoint. A sending Endpoint SHOULD Transmit messages at Endpoints identified with the URI template
836 using a protocol-specific back-channel, including but not limited to those established with a
837 `MakeConnection` message. Note, this URI is semantically similar to the WS-Addressing anonymous
838 URI if a protocol-specific back-channel is available.

839 The `MakeConnection` is a one-way operation that establishes a contextualized back-channel for the
840 transmission of messages according to matching criteria (defined below). In the non-faulting case, if no
841 matching message is available then no SOAP envelopes will be returned on the back-channel. A common
842 usage will be a client RM Destination sending `MakeConnection` to a server RM Source for the purpose
843 of receiving asynchronous response messages.

844 The following exemplar defines the `MakeConnection` syntax:

```
845 <wsrm:MakeConnection ...>  
846   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier> ?  
847   <wsrm:Address ...> xs:anyURI </wsrm:Address> ?  
848   ...  
849 </wsrm:MakeConnection>
```

850 /wsrm:MakeConnection

851 This element allows the sender to create a transport-specific back-channel that can be used to return a
852 message that matches the selection criteria. Endpoints MUST NOT send this element as a header block.

853 /wsrm:MakeConnection/wsrm:Identifier

854 This element specifies the WS-RM Sequence Identifier that establishes the context for the transport-
855 specific back-channel. The Sequence Identifier should be compared with the Sequence Identifiers
856 associated with the messages held by the sending Endpoint, and if there is a matching message it will be
857 returned. If this element is omitted from the message then the `Address` MUST be included in the
858 message.

859 /wsrm:MakeConnection/wsrm:Identifier/@{any}

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

862 /wsrm:MakeConnection/wsrm:Address

863 This element specifies the URI (`wsa:Address`) of the initiating Endpoint. Endpoints MUST NOT return
864 messages on the transport-specific back-channel unless they have been addressed to this URI. This
865 `Address` property and a message's WS-Addressing destination property are considered identical when
866 they are exactly the same character-for-character. Note that URIs which are not identical in this sense
867 may in fact be functionally equivalent. Examples include URI references which differ only in case, or
868 which are in external entities which have different effective base URIs. If this element is omitted from the
869 message then the `Identifier` MUST be included in the message.

870 `/wsrm:MakeConnection/wsrm:Address/@{any}`

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

873 `/wsrm:MakeConnection/{any}`

874 This is an extensibility mechanism to allow different (extensible) types of information, based on a schema,
875 to be passed. This allows fine-tuning of the messages to be returned, additional selection criteria included
876 here are logically ANDed with the `Address` and/or `Identifier`. If an extension is not supported by the
877 Endpoint then it should return a `UnsupportedSelection` fault.

878 `/wsrm:MakeConnection/@{any}`

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

881 If both `Identifier` and `Address` are present, then the Endpoint processing the `MakeConnection`
882 message MUST insure that any SOAP Envelope flowing on the backchannel MUST be associated with
883 the given Sequence and MUST be addressed to the given URI.

884 The management of messages that are awaiting the establishment of a back-channel to their receiving
885 Endpoint is an implementation detail that is outside the scope of this specification. Note, however, that
886 these messages form a class of asynchronous messages that is not dissimilar from "ordinary"
887 asynchronous messages that are waiting for the establishment of a connection to their destination
888 Endpoints.

889 This specification places no constraint on the types of messages that can be returned on the transport-
890 specific back-channel. As in an asynchronous environment, it is up to the recipient of the
891 `MakeConnection` message to decide which messages are appropriate for transmission to any particular
892 Endpoint. However, the Endpoint processing the `MakeConnection` message MUST insure that the
893 messages match the selection criteria as specified by the child elements of the `MakeConnection`
894 element.

895 **3.11 MessagePending**

896 When `MakeConnection` is used, and a message is returned on the transport-specific back-channel, the
897 `MessagePending` header SHOULD be included on the returned message as an indicator whether there
898 are additional messages waiting to be retrieved using the same selection criteria that was specified in the
899 `MakeConnection` element.

900 The following exemplar defines the `MessagePending` syntax:

```
901 <wsrm:MessagePending pending="xs:boolean" ...>  
902   ...  
903 </wsrm:MessagePending>
```

904 `/wsrm:MessagePending`

905 This element indicates whether additional messages are waiting to be retrieved.

906 /wsmr:MessagePending@pending

907 This attribute, when set to "true", indicates that there is at least one message waiting to be retrieved.

908 When this attribute is set to "false" it indicates there are currently no messages waiting to be retrieved.

909 /wsmr:MessagePending/{any}

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

912 /wsmr:MessagePending/@{any}

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

915 The absence of the `MessagePending` header has no implication as to whether there are additional
916 messages waiting to be retrieved.

917 4 Faults

918 Faults for the `CreateSequence` message exchange are treated as defined in WS-Addressing. Create
919 Sequence Refused is a possible fault reply for this operation. Unknown Sequence is a fault generated by
920 Endpoints when messages carrying RM header blocks targeted at unrecognized or terminated Sequences
921 are detected. WSRM Required is a fault generated an RM Destination that requires the use of WS-RM on
922 a Received message that did not use the protocol. All other faults in this section relate to known
923 Sequences. RM Destinations that generate Sequence faults SHOULD send those faults to the same
924 [destination] as Acknowledgement Messages.

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

927 `http://docs.oasis-open.org/ws-rx/wsr/200608/fault`

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

930 The definitions of faults use the following properties:

931 [Code] The fault code.

932 [Subcode] The fault subcode.

933 [Reason] The English language reason element.

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

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

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

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

```
940 <S:Envelope>
941   <S:Header>
942     <wsa:Action>
943       http://docs.oasis-open.org/ws-rx/wsr/200608/fault
944     </wsa:Action>
945     <!-- Headers elided for clarity. -->
946   </S:Header>
947   <S:Body>
948     <S:Fault>
949       <S:Code>
950         <S:Value> [Code] </S:Value>
951         <S:Subcode>
952           <S:Value> [Subcode] </S:Value>
953         </S:Subcode>
954       </S:Code>
955       <S:Reason>
956         <S:Text xml:lang="en"> [Reason] </S:Text>
957       </S:Reason>
958       <S:Detail>
959         [Detail]
```

```

960     ...
961     </S:Detail>
962 </S:Fault>
963 </S:Body>
964 </S:Envelope>

```

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

```

967 <S11:Envelope>
968 <S11:Header>
969   <wsrm:SequenceFault>
970     <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
971     <wsrm:Detail> [Detail] </wsrm:Detail>
972     ...
973   </wsrm:SequenceFault>
974   <!-- Headers elided for clarity. -->
975 </S11:Header>
976 <S11:Body>
977   <S11:Fault>
978     <faultcode> [Code] </faultcode>
979     <faultstring> [Reason] </faultstring>
980   </S11:Fault>
981 </S11:Body>
982 </S11:Envelope>

```

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

```

985 <S11:Envelope>
986 <S11:Body>
987   <S11:Fault>
988     <faultcode> [Subcode] </faultcode>
989     <faultstring> [Reason] </faultstring>
990   </S11:Fault>
991 </S11:Body>
992 </S11:Envelope>

```

993 4.1 SequenceFault Element

994 The purpose of the `SequenceFault` element is to carry the specific details of a fault generated during
995 the reliable messaging specific processing of a message belonging to a Sequence. WS-
996 ReliableMessaging nodes MUST use the `SequenceFault` container only in conjunction with the SOAP
997 1.1 fault mechanism. WS-ReliableMessaging nodes MUST NOT use the `SequenceFault` container in
998 conjunction with the SOAP 1.2 binding.

999 The following exemplar defines its syntax:

```

1000 <wsrm:SequenceFault ...>
1001   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
1002   <wsrm:Detail> ... </wsrm:Detail> ?
1003   ...
1004 </wsrm:SequenceFault>

```

1005 The following describes the content model of the `SequenceFault` element.

1006 /wsrm:SequenceFault

1007 This is the element containing Sequence information for WS-ReliableMessaging

1008 /wsrm:SequenceFault/wsrm:FaultCode

1009 WS-ReliableMessaging nodes that generate a `SequenceFault` MUST set the value of this element to a
1010 qualified name from the set of fault [Subcodes] defined below.

1011 `/wsrm:SequenceFault/wsrm:Detail`

1012 This element, if present, carries application specific error information related to the fault being described.

1013 `/wsrm:SequenceFault/wsrm:Detail/{any}`

1014 The application specific error information related to the fault being described.

1015 `/wsrm:SequenceFault/wsrm:Detail/@{any}`

1016 The application specific error information related to the fault being described.

1017 `/wsrm:SequenceFault/{any}`

1018 This is an extensibility mechanism to allow different (extensible) types of information, based on a schema,
1019 to be passed.

1020 `/wsrm:SequenceFault/@{any}`

1021 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the
1022 element.

1023 4.2 Sequence Terminated

1024 The Endpoint that generates this fault SHOULD make every reasonable effort to notify the corresponding
1025 Endpoint of this decision.

1026 Properties:

1027 [Code] Sender or Receiver

1028 [Subcode] `wsrn:SequenceTerminated`

1029 [Reason] The Sequence has been terminated due to an unrecoverable error.

1030 [Detail]

1031 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

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

1032 4.3 Unknown Sequence

1033 Properties:

1034 [Code] Sender

1035 [Subcode] `wsrn:UnknownSequence`

1036 [Reason] The value of wsrn:Identifier is not a known Sequence identifier.

1037 [Detail]

1038 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

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

1039 **4.4 Invalid Acknowledgement**

1040 An example of when this fault is generated is when a message is Received by the RM Source containing
1041 a SequenceAcknowledgement covering messages that have not been sent.

1042 [Code] Sender

1043 [Subcode] wsrn:InvalidAcknowledgement

1044 [Reason] The SequenceAcknowledgement violates the cumulative Acknowledgement invariant.

1045 [Detail]

1046 `<wsrm:SequenceAcknowledgement ...> ... </wsrm:SequenceAcknowledgement>`

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Source.	In response to a SequenceAcknowledgement that violate the invariants stated in 2.3 or any of the requirements in 3.6 about valid combinations of AckRange, Nack and None in a single SequenceAcknowledgement element or with respect to already Received such elements.	Unspecified.	Unspecified.

1047 **4.5 Message Number Rollover**

1048 If the condition listed below is reached, the RM Destination MUST generate this fault.

1049 Properties:

1050 [Code] Sender

1051 [Subcode] wsrn:MessageNumberRollover

1052 [Reason] The maximum value for wsrn:MessageNumber has been exceeded.

1053 [Detail]

```
1054 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
1055 <wsrm:MaxMessageNumber> wsrm:MessageNumberType </wsrm:MaxMessageNumber>
```

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

1056 4.6 Create Sequence Refused

1057 Properties:

1058 [Code] Sender

1059 [Subcode] wsrm:CreateSequenceRefused

1060 [Reason] The create Sequence request has been refused by the RM Destination.

1061 [Detail]

```
1062 xs:any
```

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

1063 4.7 Sequence Closed

1064 This fault is generated by an RM Destination to indicate that the specified Sequence has been closed.

1065 This fault MUST be generated when an RM Destination is asked to accept a message for a Sequence that
1066 is closed or when an RM Destination is asked to close a Sequence that is already closed.

1067 Properties:

1068 [Code] Sender

1069 [Subcode] wsrm:SequenceClosed

1070 [Reason] The Sequence is closed and can not accept new messages.

1071 [Detail]

1072 `<wsrm:Identifier...> xs:anyURI </wsrm:Identifier>`

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

1073 **4.8 WSRM Required**

1074 If an RM Destination requires the use of WS-RM, this fault is generated when it Receives an incoming
1075 message that did not use this protocol.

1076 Properties:

1077 [Code] Sender

1078 [Subcode] wsrm:WSRMRequired

1079 [Reason] The RM Destination requires the use of WSRM.

1080 [Detail]

1081 `xs:any`

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Destination.	On receipt of a message that does not use this protocol and for which this protocol is required.	Unspecified.	Unspecified.

1082 **4.9 Unsupported Selection**

1083 The QName of the unsupported element(s) are included in the detail.

1084 Properties:

1085 [Code] Receiver

1086 [Subcode] wsrm:UnsupportedSelection

1087 [Reason] The extension element used in the message selection is not supported by the RM Source

1088 [Detail]

1089 `<wsrm:UnsupportedElement> xs:QName </wsrm:UnsupportedElement>+`

Generated by	Condition	Action Upon Generation	Action Upon Receipt
RM Source or RM Destination.	In response to a <code>MakeConnection</code> message containing a selection criteria in the extensibility section of the message that is not support.ed	Unspecified.	Unspecified.

1090 **5 Security Threats and Countermeasures**

1091 This specification considers two sets of security requirements, those of the applications that use the WS-
1092 RM protocol and those of the protocol itself.

1093 This specification makes no assumptions about the security requirements of the applications that use WS-
1094 RM. However, once those requirements have been satisfied within a given operational context, the
1095 addition of WS-RM to this operational context should not undermine the fulfillment of those requirements;
1096 the use of WS-RM should not create additional attack vectors within an otherwise secure system.

1097 There are many other security concerns that one may need to consider when implementing or using this
1098 protocol. The material below should not be considered as a "check list". Implementers and users of this
1099 protocol are urged to perform a security analysis to determine their particular threat profile and the
1100 appropriate responses to those threats.

1101 Implementers are also advised that there is a core tension between security and reliable messaging that
1102 can be problematic if not addressed by implementations; one aspect of security is to prevent message
1103 replay but one of the invariants of this protocol is to resend messages until they are acknowledged.
1104 Consequently, if the security sub-system processes a message but a failure occurs before the reliable
1105 messaging sub-system Receives that message, then it is possible (and likely) that the security sub-system
1106 will treat subsequent copies as replays and discard them. At the same time, the reliable messaging sub-
1107 system will likely continue to expect and even solicit the missing message(s). Care should be taken to
1108 avoid and prevent this condition.

1109 **5.1 Threats and Countermeasures**

1110 The primary security requirement of this protocol is to protect the specified semantics and protocol
1111 invariants against various threats. The following sections describe several threats to the integrity and
1112 operation of this protocol and provide some general outlines of countermeasures to those threats.
1113 Implementers and users of this protocol should keep in mind that all threats are not necessarily applicable
1114 to all operational contexts.

1115 **5.1.1 Integrity Threats**

1116 In general, any mechanism which allows an attacker to alter the information in a Sequence Traffic
1117 Message, Sequence Lifecycle Message, Acknowledgement Messages, Acknowledgement Request, or
1118 Sequence-related fault, or which allows an attacker to alter the correlation of a RM Protocol Header Block
1119 to its intended message represents a threat to the WS-RM protocol.

1120 For example, if an attacker is able to swap `Sequence` headers on messages in transit between the RM
1121 Source and RM Destination then they have undermined the implementation's ability to guarantee the first
1122 invariant described in Section 2.3. The result is that there is no way of guaranteeing that messages will be
1123 Delivered to the Application Destination in the same order that they were sent by the Application Source.

1124 **5.1.1.1 Countermeasures**

1125 Integrity threats are generally countered via the use of digital signatures some level of the communication
1126 protocol stack. Note that, in order to counter header swapping attacks, the signature SHOULD include
1127 both the SOAP body and any relevant SOAP headers (e.g. `Sequence` header). Because some headers
1128 (`AckRequested`, `SequenceAcknowledgement`) are independent of the body of the SOAP message in which
1129 they occur, implementations MUST allow for signatures that cover only these headers.

1130 **5.1.2 Resource Consumption Threats**

1131 The creation of a Sequence with an RM Destination consumes various resources on the systems used to
1132 implement that RM Destination. These resources can include network connections, database tables,
1133 message queues, etc. This behavior can be exploited to conduct denial of service attacks against an RM
1134 Destination. For example, a simple attack is to repeatedly send `CreateSequence` messages to an RM
1135 Destination. Another attack is to create a Sequence for a service that is known to require in-order
1136 message Delivery and use this Sequence to send a stream of very large messages to that service,
1137 making sure to omit message number “1” from that stream.

1138 **5.1.2.1 Countermeasures**

1139 There are a number of countermeasures against the described resource consumption threats. The
1140 technique advocated by this specification is for the RM Destination to restrict the ability to create a
1141 Sequence to a specific set of entities/principals. This reduces the number of potential attackers and, in
1142 some cases, allows the identity of any attackers to be determined.

1143 The ability to restrict Sequence creation depends, in turn, upon the RM Destination's ability identify and
1144 authenticate the RM Source that issued the `CreateSequence` message.

1145 **5.1.3 Sequence Spoofing Threats**

1146 Sequence spoofing is a class of threats in which the attacker uses knowledge of the `Identifier` for a
1147 particular Sequence to forge Sequence Lifecycle or Traffic Messages. For example the attacker creates a
1148 fake `TerminateSequence` message that references the target Sequence and sends this message to the
1149 appropriate RM Destination. Some sequence spoofing attacks also require up-to-date knowledge of the
1150 current `MessageNumber` for their target Sequence.

1151 In general any Sequence Lifecycle Message, RM Protocol Header Block, or sequence-correlated SOAP
1152 fault (e.g. `InvalidAcknowledgement`) can be used by someone with knowledge of the Sequence identifier
1153 to attack the Sequence. These attacks are “two-way” in that an attacker may choose to target the RM
1154 Source by, for example, inserting a fake `SequenceAcknowledgement` header into a message that it sends
1155 to the `AcksTo` EPR of an RM Source.

1156 **5.1.3.1 Sequence Hijacking**

1157 Sequence hijacking is a specific case of a sequence spoofing attack. The attacker attempts to inject
1158 Sequence Traffic Messages into an existing Sequence by inserting fake `Sequence` headers into those
1159 messages.

1160 Note that “sequence hijacking” should not be equated with “security session hijacking”. Although a
1161 Sequence may be bound to some form of a security session in order to counter the threats described in
1162 this section, applications MUST NOT rely on WS-RM-related information to make determinations about
1163 the identity of the entity that created a message; applications SHOULD rely only upon information that is
1164 established by the security infrastructure to make such determinations. Failure to observe this rule
1165 creates, among other problems, a situation in which the absence of WS-RM may deprive an application of
1166 the ability to authenticate its peers even though the necessary security processing has taken place.

1167 **5.1.3.2 Countermeasures**

1168 There are a number of countermeasures against sequence spoofing threats. The technique advocated by
1169 this specification is to consider the Sequence to be a shared resource that is jointly owned by the RM

1170 Source that initiated its creation (i.e. that sent the `CreateSequence` message) and the RM Destination that
1171 serves as its terminus (i.e. that sent the `CreateSequenceResponse` message). To counter sequence
1172 spoofing attempts the RM Destination SHOULD ensure that every message or fault that it Receives that
1173 refers to a particular Sequence originated from the RM Source that jointly owns the referenced Sequence.
1174 For its part the RM Source SHOULD ensure that every message or fault that it Receives that refers to a
1175 particular Sequence originated from the RM Destination that jointly owns the referenced Sequence.

1176 For the RM Destination to be able to identify its sequence peer it MUST be able to identify and
1177 authenticate the entity that sent the `CreateSequence` message. Similarly for the RM Source to identify its
1178 sequence peer it MUST be able to identify and authenticate the entity that sent the
1179 `CreateSequenceResponse` message. For either the RM Destination or the RM Source to determine if a
1180 message was sent by its sequence peer it MUST be able to identify and authenticate the initiator of that
1181 message and, if necessary, correlate this identity with the sequence peer identity established at sequence
1182 creation time.

1183 **5.2 Security Solutions and Technologies**

1184 The security threats described in the previous sections are neither new nor unique. The solutions that
1185 have been developed to secure other SOAP-based protocols can be used to secure WS-RM as well. This
1186 section maps the facilities provided by common web services security solutions against countermeasures
1187 described in the previous sections.

1188 Before continuing this discussion, however, some examination of the underlying requirements of the
1189 previously described countermeasures is necessary. Specifically it should be noted that the technique
1190 described in Section 5.1.2.1 has two components. Firstly, the RM Destination identifies and authenticates
1191 the issuer of a `CreateSequence` message. Secondly, the RM Destination to performs an authorization
1192 check against this authenticated identity and determines if the RM Source is permitted to create
1193 Sequences with the RM Destination. Since the facilities for performing this authorization check (runtime
1194 infrastructure, policy frameworks, etc.) lie completely within the domain of individual implementations, any
1195 discussion of such facilities is considered to be beyond the scope of this specification.

1196 **5.2.1 Transport Layer Security**

1197 This section describes how the the facilities provided by SSL/TLS [RFC 4346] can be used to implement
1198 the countermeasures described in the previous sections. The use of SSL/TLS is subject to the constraints
1199 defined in Section 4 of the Basic Security Profile 1.0 [BSP 1.0].

1200 The description provided here is general in nature and is not intended to serve as a complete definition on
1201 the use of SSL/TLS to protect WS-RM. In order to interoperate implementations need to agree on the
1202 choice of features as well as the manner in which they will be used. The mechanisms described in the
1203 Web Services Security Policy Language [SecurityPolicy] MAY be used by services to describe the
1204 requirements and constraints of the use of SSL/TLS.

1205 **5.2.1.1 Model**

1206 The basic model for using SSL/TLS is as follows:

- 1207 1. The RM Source establishes an SSL/TLS session with the RM Destination.
- 1208 2. The RM Source uses this SSL/TLS session to send a `CreateSequence` message to the RM
1209 Destination.

- 1210 3. The RM Destination establishes an SSL/TLS session with the RM Source and sends an
1211 asynchronous `CreateSequenceResponse` using this session. Alternately it may respond with a
1212 synchronous `CreateSequenceResponse` using the session established in (1).
- 1213 4. For the lifetime of the Sequence the RM Source uses the SSL/TLS session from (1) to Transmit
1214 any and all messages or faults that refer to that Sequence.
- 1215 5. For the lifetime of the Sequence the RM Destination either uses the SSL/TLS session established
1216 in (3) to Transmit any and all messages or faults that refer to that Sequence or, for synchronous
1217 exchanges, the RM Destination uses the SSL/TLS session established in (1).

1218 5.2.1.2 Countermeasure Implementation

1219 Used in its simplest fashion (without relying upon any authentication mechanisms), SSL/TLS provides the
1220 necessary integrity qualities to counter the threats described in Section 5.1.1. Note, however, that the
1221 nature of SSL/TLS limits the scope of this integrity protection to a single transport level session. If
1222 SSL/TLS is the only mechanism used to provide integrity, any intermediaries between the RM Source and
1223 the RM Destination MUST be trusted to preserve the integrity of the messages that flow through them.

1224 As noted, the technique described in Sections 5.1.2.1 involves the use of authentication. This specification
1225 advocates either of two mechanisms for authenticating entities using SSL/TLS. In both of these methods
1226 the SSL/TLS server (the party accepting the SSL/TLS connection) authenticates itself to the SSL/TLS
1227 client using an X.509 certificate that is exchanged during the SSL/TLS handshake.

- 1228 • **HTTP Basic Authentication:** This method of authentication presupposes that a SOAP/HTTP
1229 binding is being used as part of the protocol stack beneath WS-RM. Subsequent to the
1230 establishment of the the SSL/TLS session, the sending party authenticates itself to the receiving
1231 party using HTTP Basic Authentication [[RFC 2617](#)]. For example, a RM Source might
1232 authenticate itself to a RM Destination (e.g. when transmitting a Sequence Traffic Message) using
1233 BasicAuth. Similarly the RM Destination might authenticate itself to the RM Source (e.g. when
1234 sending an Acknowledgement) using BasicAuth.
- 1235 • **SSL/TLS Client Authentication:** In this method of authentication, the party initiating the
1236 connection authenticates itself to the party accepting the connection using an X.509 certificate
1237 that is exchanged during the SSL/TLS handshake.

1238 To implement the countermeasures described in section 5.1.2.1 the RM Source must authenticate itself
1239 using one the above mechanisms. The authenticated identity can then be used to determine if the RM
1240 Source is authorized to create a Sequence with the RM Destination.

1241 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring
1242 an RM node's Sequence peer to be equivalent to their SSL/TLS session peer. This allows the
1243 authorization decisions described in section 5.1.3.2 to be based on SSL/TLS session identity rather than
1244 on authentication information. For example, an RM Destination can determine that a Sequence Traffic
1245 Message rightfully belongs to its referenced Sequence if that message arrived over the same SSL/TLS
1246 session that was used to carry the `CreateSequence` message for that Sequence. Note that requiring a
1247 one-to-one relationship between SSL/TLS session peer and Sequence peer constrains the lifetime of a
1248 SSL/TLS-protected Sequence to be less than or equal to the lifetime of the SSL/TLS session that is used
1249 to protect that Sequence.

1250 This specification does not preclude the use of other methods of using SSL/TLS to implement the
1251 countermeasures (such as associating specific authentication information with a Sequence) although such
1252 methods are not covered by this document.

1253 Issues specific to the life-cycle management of SSL/TLS sessions (such as the resumption of a SSL/TLS
1254 session) are outside the scope of this specification.

1255 **5.2.2 SOAP Message Security**

1256 The mechanisms described in WS-Security may be used in various ways to implement the
1257 countermeasures described in the previous sections. This specification advocates using the protocol
1258 described by WS-SecureConversation [[SecureConversation](#)] (optionally in conjunction with WS-Trust
1259 [[Trust](#)]) as a mechanism for protecting Sequences. The use of WS-Security (as an underlying component
1260 of WS-SecureConversation) is subject to the constraints defined in the Basic Security Profile 1.0.

1261 The description provided here is general in nature and is not intended to serve as a complete definition on
1262 the use of WS-SecureConversation/WS-Trust to protect WS-RM. In order to interoperate implementations
1263 need to agree on the choice of features as well as the manner in which they will be used. The
1264 mechanisms described in the Web Services Security Policy Language MAY be used by services to
1265 describe the requirements and constraints of the use of WS-SecureConversation.

1266 **5.2.2.1 Model**

1267 The basic model for using WS-SecureConversation is as follows:

- 1268 1. The RM Source and the RM Destination create a WS-SecureConversation security context. This
1269 may involve the participation of third parties such as a security token service. The tokens
1270 exchanged may contain authentication claims (e.g. X.509 certificates or Kerberos service tickets).
- 1271 2. During the `CreateSequence` exchange, the RM Source SHOULD explicitly identify the security
1272 context that will be used to protect the Sequence. This is done so that, in cases where the
1273 `CreateSequence` message is signed by more than one security context, the RM Source can
1274 indicate which security context should be used to protect the newly created Sequence.
- 1275 3. For the lifetime of the Sequence the RM Source and the RM Destination use the session key(s)
1276 associated with the security context to sign (as defined by WS-Security) at least the body and any
1277 relevant WS-RM-defined headers of any and all messages or faults that refer to that Sequence.

1278 **5.2.2.2 Countermeasure Implementation**

1279 Without relying upon any authentication information, the per-message signatures provide the necessary
1280 integrity qualities to counter the threats described in Section 5.1.1.

1281 To implement the countermeasures described in section 5.1.2.1 some mutually agreed upon form of
1282 authentication claims must be provided by the RM Source to the RM Destination during the establishment
1283 of the Security Context. These claims can then be used to determine if the RM Source is authorized to
1284 create a Sequence with the RM Destination.

1285 This specification advocates implementing the countermeasures described in section 5.1.3.2 by requiring
1286 an RM node's Sequence peer to be equivalent to their security context session peer. This allows the
1287 authorization decisions described in section 5.1.3.2 to be based on the identity of the message's security
1288 context rather than on any authentication claims that may have been established during security context
1289 initiation. Note that other methods of using WS-SecurityConversation to implement the countermeasures
1290 (such as associating specific authentication claims to a Sequence) are possible but not covered by this
1291 document.

1292 As with transport security, the requisite equivalence of a security context peer and with a Sequence peer
1293 limits the lifetime of a Sequence to the lifetime of the protecting security context. Unlike transport security,

1294 the association between a Sequence and its protecting security context cannot always be established
1295 implicitly at Sequence creation time. This is due to the fact that the `CreateSequence` and
1296 `CreateSequenceResponse` messages may be signed by more than one security context.

1297 Issues specific to the life-cycle management of WS-SecurityConversation security contexts (such as
1298 amending or renewing contexts) are outside the scope of this specification.

1299 6 Securing Sequences

1300 As noted in Section 5, the RM Source and RM Destination should be able to protect their shared
1301 Sequences against the threat of Sequence Spoofing attacks. There are a number of OPTIONAL means of
1302 achieving this objective depending upon the underlying security infrastructure.

1303 6.1 Securing Sequences Using WS-Security

1304 One mechanism for protecting a Sequence is to include a security token using a
1305 `wsse:SecurityTokenReference` element from WS-Security (see section 9 in WS-
1306 SecureConversation) in the `CreateSequence` element. This establishes an association between the
1307 created (and, if present, offered) Sequence(s) and the referenced security token, such that the RM Source
1308 and Destination MUST use the security token as the basis for authorization of all subsequent interactions
1309 related to the Sequence(s). The `wsse:SecurityTokenReference` explicitly identifies the token as
1310 there may be more than one token on a `CreateSequence` message or inferred from the communication
1311 context (e.g. transport protection).

1312 It is RECOMMENDED that a message independent referencing mechanism be used to identify the token,
1313 if the token being referenced supports such mechanism.

1314 The following exemplar defines the `CreateSequence` syntax when extended to include a
1315 `wsse:SecurityTokenReference`:

```
1316 <wsrm:CreateSequence ...>  
1317   <wsrm:AcksTo> wsa:EndpointReferenceType </wsrm:AcksTo>  
1318   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1319   <wsrm:Offer ...>  
1320     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
1321     <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>  
1322     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?  
1323     <wsrm:IncompleteSequenceBehavior>  
1324       wsrml:IncompleteSequenceBehaviorType  
1325     </wsrm:IncompleteSequenceBehavior> ?  
1326     ...  
1327   </wsrm:Offer> ?  
1328   ...  
1329   <wsse:SecurityTokenReference>  
1330     ...  
1331   </wsse:SecurityTokenReference> ?  
1332   ...  
1333 </wsrm:CreateSequence>
```

1334 `/wsrm:CreateSequence/wsse:SecurityTokenReference`

1335 This element uses the extensibility mechanism defined for the `CreateSequence` element (defined in
1336 section 3.1) to communicate an explicit reference to the security token, using a
1337 `wsse:SecurityTokenReference` as documented in WS-Security, that the RM Source and Destination
1338 MUST use to authorize messages for the created (and, if present, the offered) Sequence(s). All
1339 subsequent messages related to the created (and, if present, the offered) Sequence(s) MUST
1340 demonstrate proof-of-possession of the secret associated with the token (e.g., by using or deriving from a
1341 private or secret key).

1342 When a RM Source Transmits a `CreateSequence` that has been extended to include a
1343 `wsse:SecurityTokenReference` it SHOULD ensure that the RM Destination both understands and
1344 will conform with the requirements listed above. In order to achieve this, the RM Source SHOULD include
1345 the `UsesSequenceSTR` element as a SOAP header block within the `CreateSequence` message. This
1346 element MUST include a `soap:mustUnderstand` attribute with a value of 'true'. Thus the RM Source

1347 can be assured that a RM Destination that responds with a `CreateSequenceResponse` understands
1348 and conforms with the requirements listed above. Note that an RM Destination understanding this header
1349 does not mean that it has processed and understood any WS-Security headers, the fault behavior defined
1350 in WS-Security still applies.

1351 The following exemplar defines the `UsesSequenceSTR` syntax:

```
1352 <wsrm:UsesSequenceSTR ... />
```

1353 /wsrm:UsesSequenceSTR

1354 This element SHOULD be included as a SOAP header block in `CreateSequence` messages that use the
1355 extensibility mechanism described above in this section. The `soap:mustUnderstand` attribute value
1356 MUST be 'true'. The receiving RM Destination MUST understand and correctly implement the extension
1357 described above or else generate a `soap:MustUnderstand` fault, thus aborting the requested
1358 Sequence creation.

1359 The following is an example of a `CreateSequence` message using the
1360 `wsse:SecurityTokenReference` extension and the `UsesSequenceSTR` header block:

```
1361 <soap:Envelope ...>  
1362   <soap:Header>  
1363     ...  
1364     <wsrm:UsesSequenceSTR soap:mustUnderstand='true' />  
1365     ...  
1366   </soap:Header>  
1367   <soap:Body>  
1368     <wsrm:CreateSequence>  
1369       <wsrm:AcksTo>  
1370         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>  
1371       </wsrm:AcksTo>  
1372       <wsse:SecurityTokenReference>  
1373         ...  
1374       </wsse:SecurityTokenReference>  
1375     </wsrm:CreateSequence>  
1376   </soap:Body>  
1377 </soap:Envelope>
```

1378 6.2 Securing Sequences Using SSL/TLS

1379 One mechanism for protecting a Sequence is to bind the Sequence to the underlying SSL/TLS session(s).
1380 The RM Source indicates to the RM Destination that a Sequence is to be bound to the underlying
1381 SSL/TLS session(s) via the `UsesSequenceSSL` header block. If the RM Source wishes to bind a
1382 Sequence to the underlying SSL/TLS sessions(s) it MUST include the `UsesSequenceSSL` element as a
1383 SOAP header block within the `CreateSequence` message.

1384 The following exemplar defines the `UsesSequenceSSL` syntax:

```
1385 <wsrm:UsesSequenceSSL soap:mustUnderstand="true" ... />
```

1386 /wsrm:UsesSequenceSSL

1387 The RM Source MAY include this element as a SOAP header block of a `CreateSequence` message to
1388 indicate to the RM Destination that the resulting Sequence is to be bound to the SSL/TLS session that was
1389 used to carry the `CreateSequence` message. If included, the RM Source MUST mark this header with a
1390 `soap:mustUnderstand` attribute with a value of 'true'. The receiving RM Destination MUST understand
1391 and correctly implement the functionality described in Section 5.2.1 or else generate a
1392 `soap:MustUnderstand` fault, thus aborting the requested Sequence creation.

1393 Note that the use inclusion of the above header by the RM Source implies that all Sequence-related
1394 information (Sequence Lifecycle or Acknowledgment messages or Sequence-related faults) flowing from
1395 the RM Destination to the RM Source will be bound to the SSL/TLS session that is used to carry the
1396 `CreateSequenceResponse` message.

1397 **7 References**

1398 **7.1 Normative**

1399 **[KEYWORDS]**

1400 S. Bradner, "[Key words for use in RFCs to Indicate Requirement Levels](#)," RFC 2119, Harvard University,
1401 March 1997

1402 **[SOAP 1.1]**

1403 W3C Note, "[SOAP: Simple Object Access Protocol 1.1](#)," 08 May 2000.

1404 **[SOAP 1.2]**

1405 W3C Recommendation, "[SOAP Version 1.2 Part 1: Messaging Framework](#)" June 2003.

1406 **[URI]**

1407 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#)," RFC 3986,
1408 MIT/LCS, U.C. Irvine, Xerox Corporation, January 2005.

1409 **[UUID]**

1410 P. Leach, M. Mealling, R. Salz, "[A Universally Unique Identifier \(UUID\) URN Namespace](#)," RFC 4122,
1411 Microsoft, Refactored Networks - LLC, DataPower Technology Inc, July 2005

1412 **[XML]**

1413 W3C Recommendation, "[Extensible Markup Language \(XML\) 1.0 \(Second Edition\)](#)", October 2000.

1414 **[XML-ns]**

1415 W3C Recommendation, "[Namespaces in XML](#)," 14 January 1999.

1416 **[XML-Schema Part1]**

1417 W3C Recommendation, "[XML Schema Part 1: Structures](#)," 2 May 2001.

1418 **[XML-Schema Part2]**

1419 W3C Recommendation, "[XML Schema Part 2: Datatypes](#)," 2 May 2001.

1420 **[XPath 1.0]**

1421 W3C Recommendation, "[XML Path Language \(XPath\) Version 1.0](#)," 16 November 1999.

1422 **[WSDL 1.1]**

1423 W3C Note, "[Web Services Description Language \(WSDL 1.1\)](#)," 15 March 2001.

1424 **[WS-Addressing]**

1425 W3C Recommendation, "[Web Services Addressing 1.0 - Core](#)", May 2006.

1426 W3C Recommendation, "[Web Services Addressing 1.0 – SOAP Binding](#)", May 2006.

1427 **7.2 Non-Normative**

1428 **[BSP 1.0]**

1429 WS-I Working Group Draft. "[Basic Security Profile Version 1.0](#)," March 2006

1430 **[RDDL 2.0]**

- 1431 Johnathan Borden, Tim Bray, eds. "[Resource Directory Description Language \(RDDL\) 2.0](#)," January 2004
- 1432 **[RFC 2617]**
- 1433 J. Franks, P. Hallam-Baker, J. Hostetler, S. Lawrence, P. Leach, A. Loutonen, L. Stewart, "[HTTP](#)
- 1434 [Authentication: Basic and Digest Access Authentication](#)," June 1999.
- 1435 **[RFC 4346]**
- 1436 T. Dierks, E. Rescorla, "[The Transport Layer Security \(TLS\) Protocol Version 1.1](#)," April 2006.
- 1437 **[WS-Policy]**
- 1438 W3C Member Submission, "[Web Services Policy Framework \(WS-Policy\)](#)," April 2006.
- 1439 **[WS-PolicyAttachment]**
- 1440 W3C Member Submission, "[Web Services Policy Attachment \(WS-PolicyAttachment\)](#)," April 2006.
- 1441 **[WS-Security]**
- 1442 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "[OASIS Web Services Security: SOAP Message Security 1.0 \(WS-Security 2004\)](#)", OASIS Standard 200401, March 2004.
- 1443
- 1444 Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "[OASIS Web Services Security: SOAP Message Security 1.1 \(WS-Security 2004\)](#)", OASIS Standard 200602, February 2006.
- 1445
- 1446 **[RTTM]**
- 1447 V. Jacobson, R. Braden, D. Borman, "[TCP Extensions for High Performance](#)", RFC 1323, May
- 1448 1992.
- 1449 **[SecurityPolicy]**
- 1450 G. Della-Libra, et. al. "[Web Services Security Policy Language \(WS-SecurityPolicy\)](#)", July 2005
- 1451 **[SecureConversation]**
- 1452 S. Anderson, et al, "[Web Services Secure Conversation Language \(WS-SecureConversation\)](#)," February
- 1453 2005.
- 1454 **[Trust]**
- 1455 S. Anderson, et al, "[Web Services Trust Language \(WS-Trust\)](#)," February 2005.

1456 Appendix A. Schema

1457 The normative schema that is defined for WS-ReliableMessaging using [XML-Schema Part1] and [XML-
1458 Schema Part2] is located at:

1459 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-200608.xsd>

1460 The following copy is provided for reference.

```
1461 <?xml version="1.0" encoding="UTF-8"?>
1462 <!--
1463 OASIS takes no position regarding the validity or scope of any intellectual
1464 property or other rights that might be claimed to pertain to the
1465 implementation or use of the technology described in this document or the
1466 extent to which any license under such rights might or might not be available;
1467 neither does it represent that it has made any effort to identify any such
1468 rights. Information on OASIS's procedures with respect to rights in OASIS
1469 specifications can be found at the OASIS website. Copies of claims of rights
1470 made available for publication and any assurances of licenses to be made
1471 available, or the result of an attempt made to obtain a general license or
1472 permission for the use of such proprietary rights by implementors or users of
1473 this specification, can be obtained from the OASIS Executive Director.
1474 OASIS invites any interested party to bring to its attention any copyrights,
1475 patents or patent applications, or other proprietary rights which may cover
1476 technology that may be required to implement this specification. Please
1477 address the information to the OASIS Executive Director.
1478 Copyright © OASIS Open 2002-2006. All Rights Reserved.
1479 This document and translations of it may be copied and furnished to others,
1480 and derivative works that comment on or otherwise explain it or assist in its
1481 implementation may be prepared, copied, published and distributed, in whole or
1482 in part, without restriction of any kind, provided that the above copyright
1483 notice and this paragraph are included on all such copies and derivative
1484 works. However, this document itself does not be modified in any way, such as
1485 by removing the copyright notice or references to OASIS, except as needed for
1486 the purpose of developing OASIS specifications, in which case the procedures
1487 for copyrights defined in the OASIS Intellectual Property Rights document must
1488 be followed, or as required to translate it into languages other than English.
1489 The limited permissions granted above are perpetual and will not be revoked by
1490 OASIS or its successors or assigns.
1491 This document and the information contained herein is provided on an "AS IS"
1492 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1493 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1494 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1495 FOR A PARTICULAR PURPOSE.
1496 -->
1497 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
1498 xmlns:wsa="http://www.w3.org/2005/08/addressing"
1499 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1500 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1501 elementFormDefault="qualified" attributeFormDefault="unqualified">
1502   <xs:import namespace="http://www.w3.org/2005/08/addressing"
1503   schemaLocation="http://www.w3.org/2006/03/addressing/ws-addr.xsd"/>
1504   <!-- Protocol Elements -->
1505   <xs:complexType name="SequenceType">
1506     <xs:sequence>
1507       <xs:element ref="wsrm:Identifier"/>
1508       <xs:element name="MessageNumber" type="wsrm:MessageNumberType"/>
1509       <xs:any namespace="##other" processContents="lax" minOccurs="0"
1510 maxOccurs="unbounded"/>
1511     </xs:sequence>
```

```

1512     <xs:anyAttribute namespace="##other" processContents="lax"/>
1513 </xs:complexType>
1514 <xs:element name="Sequence" type="wsrm:SequenceType"/>
1515 <xs:element name="SequenceAcknowledgement">
1516   <xs:complexType>
1517     <xs:sequence>
1518       <xs:element ref="wsrm:Identifier"/>
1519       <xs:choice>
1520         <xs:sequence>
1521           <xs:choice>
1522             <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
1523               <xs:complexType>
1524                 <xs:sequence/>
1525                 <xs:attribute name="Upper" type="xs:unsignedLong"
1526 use="required"/>
1527                 <xs:attribute name="Lower" type="xs:unsignedLong"
1528 use="required"/>
1529               <xs:anyAttribute namespace="##other" processContents="lax"/>
1530             </xs:complexType>
1531           </xs:element>
1532           <xs:element name="None">
1533             <xs:complexType>
1534               <xs:sequence/>
1535             </xs:complexType>
1536           </xs:element>
1537         </xs:choice>
1538         <xs:element name="Final" minOccurs="0">
1539           <xs:complexType>
1540             <xs:sequence/>
1541           </xs:complexType>
1542         </xs:element>
1543       </xs:sequence>
1544       <xs:element name="Nack" type="xs:unsignedLong"
1545 maxOccurs="unbounded"/>
1546     </xs:choice>
1547     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1548 maxOccurs="unbounded"/>
1549   </xs:sequence>
1550   <xs:anyAttribute namespace="##other" processContents="lax"/>
1551 </xs:complexType>
1552 </xs:element>
1553 <xs:complexType name="AckRequestedType">
1554   <xs:sequence>
1555     <xs:element ref="wsrm:Identifier"/>
1556     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1557 maxOccurs="unbounded"/>
1558   </xs:sequence>
1559   <xs:anyAttribute namespace="##other" processContents="lax"/>
1560 </xs:complexType>
1561 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1562 <xs:complexType name="MessagePendingType">
1563   <xs:sequence>
1564     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1565 maxOccurs="unbounded"/>
1566   </xs:sequence>
1567   <xs:attribute name="pending" type="xs:boolean"/>
1568   <xs:anyAttribute namespace="##other" processContents="lax"/>
1569 </xs:complexType>
1570 <xs:element name="MessagePending" type="wsrm:MessagePendingType"/>
1571 <xs:element name="Identifier">
1572   <xs:complexType>
1573     <xs:annotation>
1574       <xs:documentation>

```

```

1575         This type is for elements whose [children] is an anyURI and can have
1576 arbitrary attributes.
1577         </xs:documentation>
1578     </xs:annotation>
1579     <xs:simpleContent>
1580         <xs:extension base="xs:anyURI">
1581             <xs:anyAttribute namespace="##other" processContents="lax"/>
1582         </xs:extension>
1583     </xs:simpleContent>
1584 </xs:complexType>
1585 </xs:element>
1586 <xs:element name="Address">
1587     <xs:complexType>
1588         <xs:simpleContent>
1589             <xs:extension base="xs:anyURI">
1590                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1591             </xs:extension>
1592         </xs:simpleContent>
1593     </xs:complexType>
1594 </xs:element>
1595 <xs:complexType name="MakeConnectionType">
1596     <xs:sequence>
1597         <xs:element ref="wsrm:Identifier" minOccurs="0" maxOccurs="1"/>
1598         <xs:element ref="wsrm:Address" minOccurs="0" maxOccurs="1"/>
1599         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1600 maxOccurs="unbounded"/>
1601     </xs:sequence>
1602     <xs:anyAttribute namespace="##other" processContents="lax"/>
1603 </xs:complexType>
1604 <xs:element name="MakeConnection" type="wsrm:MakeConnectionType"/>
1605 <xs:simpleType name="MessageNumberType">
1606     <xs:restriction base="xs:unsignedLong">
1607         <xs:minInclusive value="1"/>
1608         <xs:maxInclusive value="9223372036854775807"/>
1609     </xs:restriction>
1610 </xs:simpleType>
1611 <!-- Fault Container and Codes -->
1612 <xs:simpleType name="FaultCodes">
1613     <xs:restriction base="xs:QName">
1614         <xs:enumeration value="wsrm:SequenceTerminated"/>
1615         <xs:enumeration value="wsrm:UnknownSequence"/>
1616         <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1617         <xs:enumeration value="wsrm:MessageNumberRollover"/>
1618         <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1619         <xs:enumeration value="wsrm:SequenceClosed"/>
1620         <xs:enumeration value="wsrm:WSRMRequired"/>
1621         <xs:enumeration value="wsrm:UnsupportedSelection"/>
1622     </xs:restriction>
1623 </xs:simpleType>
1624 <xs:complexType name="SequenceFaultType">
1625     <xs:sequence>
1626         <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1627         <xs:element name="Detail" type="wsrm:DetailType" minOccurs="0"/>
1628         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1629 maxOccurs="unbounded"/>
1630     </xs:sequence>
1631     <xs:anyAttribute namespace="##other" processContents="lax"/>
1632 </xs:complexType>
1633 <xs:complexType name="DetailType">
1634     <xs:sequence>
1635         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1636 maxOccurs="unbounded"/>
1637     </xs:sequence>

```

```

1638     <xs:anyAttribute namespace="##other" processContents="lax"/>
1639 </xs:complexType>
1640 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1641 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1642 <xs:element name="CreateSequenceResponse"
1643 type="wsrm:CreateSequenceResponseType"/>
1644 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1645 <xs:element name="CloseSequenceResponse"
1646 type="wsrm:CloseSequenceResponseType"/>
1647 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1648 <xs:element name="TerminateSequenceResponse"
1649 type="wsrm:TerminateSequenceResponseType"/>
1650 <xs:complexType name="CreateSequenceType">
1651 <xs:sequence>
1652 <xs:element ref="wsrm:AcksTo"/>
1653 <xs:element ref="wsrm:Expires" minOccurs="0"/>
1654 <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1655 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1656 maxOccurs="unbounded">
1657 <xs:annotation>
1658 <xs:documentation>
1659 It is the authors intent that this extensibility be used to
1660 transfer a Security Token Reference as defined in WS-Security.
1661 </xs:documentation>
1662 </xs:annotation>
1663 </xs:any>
1664 </xs:sequence>
1665 <xs:anyAttribute namespace="##other" processContents="lax"/>
1666 </xs:complexType>
1667 <xs:complexType name="CreateSequenceResponseType">
1668 <xs:sequence>
1669 <xs:element ref="wsrm:Identifier"/>
1670 <xs:element ref="wsrm:Expires" minOccurs="0"/>
1671 <xs:element name="IncompleteSequenceBehavior"
1672 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1673 <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1674 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1675 maxOccurs="unbounded"/>
1676 </xs:sequence>
1677 <xs:anyAttribute namespace="##other" processContents="lax"/>
1678 </xs:complexType>
1679 <xs:complexType name="CloseSequenceType">
1680 <xs:sequence>
1681 <xs:element ref="wsrm:Identifier"/>
1682 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1683 maxOccurs="unbounded"/>
1684 </xs:sequence>
1685 <xs:anyAttribute namespace="##other" processContents="lax"/>
1686 </xs:complexType>
1687 <xs:complexType name="CloseSequenceResponseType">
1688 <xs:sequence>
1689 <xs:element ref="wsrm:Identifier"/>
1690 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1691 maxOccurs="unbounded"/>
1692 </xs:sequence>
1693 <xs:anyAttribute namespace="##other" processContents="lax"/>
1694 </xs:complexType>
1695 <xs:complexType name="TerminateSequenceType">
1696 <xs:sequence>
1697 <xs:element ref="wsrm:Identifier"/>
1698 <xs:element name="LastMsgNumber" type="wsrm:MessageNumberType"/>
1699 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1700 maxOccurs="unbounded"/>

```

```

1701     </xs:sequence>
1702     <xs:anyAttribute namespace="##other" processContents="lax"/>
1703 </xs:complexType>
1704 <xs:complexType name="TerminateSequenceResponseType">
1705     <xs:sequence>
1706         <xs:element ref="wsrm:Identifier"/>
1707         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1708 maxOccurs="unbounded"/>
1709     </xs:sequence>
1710     <xs:anyAttribute namespace="##other" processContents="lax"/>
1711 </xs:complexType>
1712 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1713 <xs:complexType name="OfferType">
1714     <xs:sequence>
1715         <xs:element ref="wsrm:Identifier"/>
1716         <xs:element name="Endpoint" type="wsa:EndpointReferenceType"/>
1717         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1718         <xs:element name="IncompleteSequenceBehavior"
1719 type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0"/>
1720         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1721 maxOccurs="unbounded"/>
1722     </xs:sequence>
1723     <xs:anyAttribute namespace="##other" processContents="lax"/>
1724 </xs:complexType>
1725 <xs:complexType name="AcceptType">
1726     <xs:sequence>
1727         <xs:element ref="wsrm:AcksTo"/>
1728         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1729 maxOccurs="unbounded"/>
1730     </xs:sequence>
1731     <xs:anyAttribute namespace="##other" processContents="lax"/>
1732 </xs:complexType>
1733 <xs:element name="Expires">
1734     <xs:complexType>
1735         <xs:simpleContent>
1736             <xs:extension base="xs:duration">
1737                 <xs:anyAttribute namespace="##other" processContents="lax"/>
1738             </xs:extension>
1739         </xs:simpleContent>
1740     </xs:complexType>
1741 </xs:element>
1742 <xs:simpleType name="IncompleteSequenceBehaviorType">
1743     <xs:restriction base="xs:string">
1744         <xs:enumeration value="DiscardEntireSequence"/>
1745         <xs:enumeration value="DiscardFollowingFirstGap"/>
1746         <xs:enumeration value="NoDiscard"/>
1747     </xs:restriction>
1748 </xs:simpleType>
1749 <xs:element name="UsesSequenceSTR">
1750     <xs:sequence/>
1751     <xs:anyAttribute namespace="##other" processContents="lax"/>
1752 </xs:element>
1753 <xs:element name="UsesSequenceSSL">
1754     <xs:sequence/>
1755     <xs:anyAttribute namespace="##other" processContents="lax"/>
1756 </xs:element>
1757 <xs:element name="UnsupportedElement">
1758     <xs:simpleType>
1759         <xs:restriction base="xs:QName"/>
1760     </xs:simpleType>
1761 </xs:element>
1762 </xs:schema>

```

1763 Appendix B. WSDL

1764 The normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1765 <http://docs.oasis-open.org/ws-rx/wsrn/200608/wsd/wsrn-1.1-wsd-200608.wsd>

1766 The following non-normative copy is provided for reference.

```
1767 <?xml version="1.0" encoding="utf-8"?>
1768 <!--
1769 OASIS takes no position regarding the validity or scope of any intellectual
1770 property or other rights that might be claimed to pertain to the
1771 implementation or use of the technology described in this document or the
1772 extent to which any license under such rights might or might not be available;
1773 neither does it represent that it has made any effort to identify any such
1774 rights. Information on OASIS's procedures with respect to rights in OASIS
1775 specifications can be found at the OASIS website. Copies of claims of rights
1776 made available for publication and any assurances of licenses to be made
1777 available, or the result of an attempt made to obtain a general license or
1778 permission for the use of such proprietary rights by implementors or users of
1779 this specification, can be obtained from the OASIS Executive Director.
1780 OASIS invites any interested party to bring to its attention any copyrights,
1781 patents or patent applications, or other proprietary rights which may cover
1782 technology that may be required to implement this specification. Please
1783 address the information to the OASIS Executive Director.
1784 Copyright (c) OASIS Open 2002-2006. All Rights Reserved.
1785 This document and translations of it may be copied and furnished to others,
1786 and derivative works that comment on or otherwise explain it or assist in its
1787 implementation may be prepared, copied, published and distributed, in whole or
1788 in part, without restriction of any kind, provided that the above copyright
1789 notice and this paragraph are included on all such copies and derivative
1790 works. However, this document itself does not be modified in any way, such as
1791 by removing the copyright notice or references to OASIS, except as needed for
1792 the purpose of developing OASIS specifications, in which case the procedures
1793 for copyrights defined in the OASIS Intellectual Property Rights document must
1794 be followed, or as required to translate it into languages other than English.
1795 The limited permissions granted above are perpetual and will not be revoked by
1796 OASIS or its successors or assigns.
1797 This document and the information contained herein is provided on an "AS IS"
1798 basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT
1799 NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT
1800 INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS
1801 FOR A PARTICULAR PURPOSE.
1802 -->
1803 <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1804 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1805 xmlns:wsa="http://www.w3.org/2005/08/addressing" xmlns:rm="http://docs.oasis-
1806 open.org/ws-rx/wsrn/200608" xmlns:tns="http://docs.oasis-open.org/ws-
1807 rx/wsrn/200608/wsd" targetNamespace="http://docs.oasis-open.org/ws-
1808 rx/wsrn/200608/wsd">
1809   <wsdl:types>
1810     <xs:schema
1811       <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1812       schemaLocation="http://docs.oasis-open.org/ws-rx/wsrn/200608/wsrn-1.1-schema-
1813       200608.xsd"/>
1814     </xs:schema>
1815   </wsdl:types>
1816   <wsdl:message name="CreateSequence">
1817     <wsdl:part name="create" element="rm:CreateSequence"/>
```

```

1818 </wsdl:message>
1819 <wsdl:message name="CreateSequenceResponse">
1820 <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
1821 </wsdl:message>
1822 <wsdl:message name="CloseSequence">
1823 <wsdl:part name="close" element="rm:CloseSequence"/>
1824 </wsdl:message>
1825 <wsdl:message name="CloseSequenceResponse">
1826 <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1827 </wsdl:message>
1828 <wsdl:message name="TerminateSequence">
1829 <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1830 </wsdl:message>
1831 <wsdl:message name="TerminateSequenceResponse">
1832 <wsdl:part name="terminateResponse"
1833 element="rm:TerminateSequenceResponse"/>
1834 </wsdl:message>
1835 <wsdl:message name="MakeConnection">
1836 <wsdl:part name="makeConnection" element="rm:MakeConnection"/>
1837 </wsdl:message>

1838 <wsdl:portType name="SequenceAbstractPortType">
1839 <wsdl:operation name="CreateSequence">
1840 <wsdl:input message="tns:CreateSequence" wsaw:Action="http://docs.oasis-
1841 open.org/ws-rx/wsrn/200608/CreateSequence"/>
1842 <wsdl:output message="tns:CreateSequenceResponse"
1843 wsaw:Action="http://docs.oasis-open.org/ws-
1844 rx/wsrn/200608/CreateSequenceResponse"/>
1845 </wsdl:operation>
1846 <wsdl:operation name="CloseSequence">
1847 <wsdl:input message="tns:CloseSequence" wsaw:Action="http://docs.oasis-
1848 open.org/ws-rx/wsrn/200608/CloseSequence"/>
1849 <wsdl:output message="tns:CloseSequenceResponse"
1850 wsaw:Action="http://docs.oasis-open.org/ws-
1851 rx/wsrn/200608/CloseSequenceResponse"/>
1852 </wsdl:operation>
1853 <wsdl:operation name="TerminateSequence">
1854 <wsdl:input message="tns:TerminateSequence"
1855 wsaw:Action="http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence"/>
1856 <wsdl:output message="tns:TerminateSequenceResponse"
1857 wsaw:Action="http://docs.oasis-open.org/ws-
1858 rx/wsrn/200608/TerminateSequenceResponse"/>
1859 </wsdl:operation>
1860 <wsdl:operation name="MakeConnection">
1861 <wsdl:input message="tns:MakeConnection" wsaw:Action="http://docs.oasis-
1862 open.org/ws-rx/wsrn/200608/MakeConnection"/>
1863 </wsdl:operation>
1864 </wsdl:portType>
1865 </wsdl:definitions>

```

1866 Appendix C. Message Examples

1867 Appendix C.1 Create Sequence

1868 Create Sequence

```
1869 <?xml version="1.0" encoding="UTF-8"?>
1870 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1871 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1872 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1873   <S:Header>
1874     <wsa:MessageID>
1875       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1876     </wsa:MessageID>
1877     <wsa:To>http://example.com/serviceB/123</wsa:To>
1878     <wsa:Action>http://docs.oasis-open.org/ws-
1879 rx/wsmr/200608/CreateSequence</wsa:Action>
1880     <wsa:ReplyTo>
1881       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1882     </wsa:ReplyTo>
1883   </S:Header>
1884   <S:Body>
1885     <wsmr:CreateSequence>
1886       <wsmr:AcksTo>
1887         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1888       </wsmr:AcksTo>
1889     </wsmr:CreateSequence>
1890   </S:Body>
1891 </S:Envelope>
```

1892 Create Sequence Response

```
1893 <?xml version="1.0" encoding="UTF-8"?>
1894 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1895 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1896 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1897   <S:Header>
1898     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1899     <wsa:RelatesTo>
1900       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1901     </wsa:RelatesTo>
1902     <wsa:Action>
1903       http://docs.oasis-open.org/ws-rx/wsmr/200608/CreateSequenceResponse
1904     </wsa:Action>
1905   </S:Header>
1906   <S:Body>
1907     <wsmr:CreateSequenceResponse>
1908       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1909     </wsmr:CreateSequenceResponse>
1910   </S:Body>
1911 </S:Envelope>
```

1912 Appendix C.2 Initial Transmission

1913 The following example WS-ReliableMessaging headers illustrate the message exchange in the above
1914 figure. The three messages have the following headers; **the third message is identified as the last**
1915 **message in the Sequence:**

1916 **Message 1**

```
1917 <?xml version="1.0" encoding="UTF-8"?>
1918 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1919 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1920 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1921   <S:Header>
1922     <wsa:MessageID>
1923       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1924     </wsa:MessageID>
1925     <wsa:To>http://example.com/serviceB/123</wsa:To>
1926     <wsa:From>
1927       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1928     </wsa:From>
1929     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1930     <wsmr:Sequence>
1931       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1932       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1933     </wsmr:Sequence>
1934   </S:Header>
1935   <S:Body>
1936     <!-- Some Application Data -->
1937   </S:Body>
1938 </S:Envelope>
```

1939 **Message 2**

```
1940 <?xml version="1.0" encoding="UTF-8"?>
1941 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1942 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1943 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1944   <S:Header>
1945     <wsa:MessageID>
1946       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1947     </wsa:MessageID>
1948     <wsa:To>http://example.com/serviceB/123</wsa:To>
1949     <wsa:From>
1950       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1951     </wsa:From>
1952     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1953     <wsmr:Sequence>
1954       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1955       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1956     </wsmr:Sequence>
1957   </S:Header>
1958   <S:Body>
1959     <!-- Some Application Data -->
1960   </S:Body>
1961 </S:Envelope>
```

1962 **Message 3**

```
1963 <?xml version="1.0" encoding="UTF-8"?>
1964 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1965 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"
1966 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1967   <S:Header>
1968     <wsa:MessageID>
1969       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1970     </wsa:MessageID>
1971     <wsa:To>http://example.com/serviceB/123</wsa:To>
1972     <wsa:From>
1973       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
```

```

1974 </wsa:From>
1975 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1976 <wsrm:Sequence>
1977 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1978 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1979 </wsrm:Sequence>
1980 <wsrm:AckRequested>
1981 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1982 </wsrm:AckRequested>
1983 </S:Header>
1984 <S:Body>
1985 <!-- Some Application Data -->
1986 </S:Body>
1987 </S:Envelope>

```

1988 **Appendix C.3 First Acknowledgement**

1989 Message number 2 has not been accepted by the RM Destination due to some transmission error so it
1990 responds with an Acknowledgement for messages 1 and 3:

```

1991 <?xml version="1.0" encoding="UTF-8"?>
1992 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1993 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
1994 xmlns:wsa="http://www.w3.org/2005/08/addressing">
1995 <S:Header>
1996 <wsa:MessageID>
1997 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1998 </wsa:MessageID>
1999 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
2000 <wsa:From>
2001 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
2002 </wsa:From>
2003 <wsa:Action>
2004 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
2005 </wsa:Action>
2006 <wsrm:SequenceAcknowledgement>
2007 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2008 <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
2009 <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
2010 </wsrm:SequenceAcknowledgement>
2011 </S:Header>
2012 <S:Body/>
2013 </S:Envelope>

```

2014 **Appendix C.4 Retransmission**

2015 The RM Sourcediscovers that message number 2 was not accepted so it resends the message and
2016 requests an Acknowledgement:

```

2017 <?xml version="1.0" encoding="UTF-8"?>
2018 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2019 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
2020 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2021 <S:Header>
2022 <wsa:MessageID>
2023 http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
2024 </wsa:MessageID>
2025 <wsa:To>http://example.com/serviceB/123</wsa:To>
2026 <wsa:From>
2027 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2028 </wsa:From>

```

```

2029 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
2030 <wsrm:Sequence>
2031 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2032 <wsrm:MessageNumber>2</wsrm:MessageNumber>
2033 </wsrm:Sequence>
2034 <wsrm:AckRequested>
2035 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2036 </wsrm:AckRequested>
2037 </S:Header>
2038 <S:Body>
2039 <!-- Some Application Data -->
2040 </S:Body>
2041 </S:Envelope>

```

2042 Appendix C.5 Termination

2043 The RM Destination now responds with an Acknowledgement for the complete Sequence which can then
 2044 be terminated: [the third message is identified as the last message in the Sequence](#):

```

2045 <?xml version="1.0" encoding="UTF-8"?>
2046 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2047 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
2048 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2049 <S:Header>
2050 <wsa:MessageID>
2051 http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
2052 </wsa:MessageID>
2053 <wsa:To>http://Business456.com/serviceA/789</wsa:To>
2054 <wsa:From>
2055 <wsa:Address>http://example.com/serviceB/123</wsa:Address>
2056 </wsa:From>
2057 <wsa:Action>
2058 http://docs.oasis-open.org/ws-rx/wsrn/200608/SequenceAcknowledgement
2059 </wsa:Action>
2060 <wsrm:SequenceAcknowledgement>
2061 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2062 <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
2063 </wsrm:SequenceAcknowledgement>
2064 </S:Header>
2065 <S:Body/>
2066 </S:Envelope>

```

2067 Terminate Sequence

```

2068 <?xml version="1.0" encoding="UTF-8"?>
2069 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2070 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
2071 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2072 <S:Header>
2073 <wsa:MessageID>
2074 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
2075 </wsa:MessageID>
2076 <wsa:To>http://example.com/serviceB/123</wsa:To>
2077 <wsa:Action>
2078 http://docs.oasis-open.org/ws-rx/wsrn/200608/TerminateSequence
2079 </wsa:Action>
2080 <wsa:From>
2081 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2082 </wsa:From>
2083 </S:Header>
2084 <S:Body>
2085 <wsrm:TerminateSequence>

```

```

2086     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2087     <wsrm>LastMsgNumber>3</wsrm>LastMsgNumber>
2088     </wsrm:TerminateSequence>
2089     </S:Body>
2090 </S:Envelope>

```

2091 Terminate Sequence Response

```

2092 <?xml version="1.0" encoding="UTF-8"?>
2093 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2094 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrmp/200608"
2095 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2096   <S:Header>
2097     <wsa:MessageID>
2098       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
2099     </wsa:MessageID>
2100     <wsa:To>http://example.com/serviceA/789</wsa:To>
2101     <wsa:Action>
2102       http://docs.oasis-open.org/ws-rx/wsrmp/200608/TerminateSequenceResponse
2103     </wsa:Action>
2104     <wsa:RelatesTo>
2105       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
2106     </wsa:RelatesTo>
2107     <wsa:From>
2108       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
2109     </wsa:From>
2110   </S:Header>
2111   <S:Body>
2112     <wsrm:TerminateSequenceResponse>
2113       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
2114     </wsrm:TerminateSequenceResponse>
2115   </S:Body>
2116 </S:Envelope>

```

2117 Appendix C.6 MakeConnection

2118 To illustrate how a `MakeConnection` message exchange can be used to deliver messages to an
2119 Endpoint that is not addressable, consider the case of a pub/sub scenario in which the Endpoint to which
2120 notifications are to be delivered (the "event consumer") is not addressable by the notification sending
2121 Endpoint (the "event producer"). In this scenario the event consumer must initiate the connections in order
2122 for the notifications to be delivered. One possible set of message exchanges (using HTTP) that
2123 demonstrate how this can be achieved using `MakeConnection` is shown below.

2124 **Step 1** – During a "subscribe" operation, the event consumer's EPR specifies the RM anonymous URI
2125 and the RM Policy Assertion to indicate whether or not RM is required:

```

2126 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2127 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrmp/200608"
2128 xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200608"
2129 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2130   <S:Header>
2131     <wsa:To> http://example.org/subscriptionService </wsa:To>
2132     <wsa:MessageID> http://client456.org/id-a6d8-a7c2eb546813</wsa:MessageID>
2133     <wsa:ReplyTo>
2134       <wsa:To> http://client456.org/response </wsa:To>
2135     </wsa:ReplyTo>
2136   </S:Header>
2137   <S:Body>
2138     <sub:Subscribe xmlns:sub="http://example.org/subscriptionService">
2139       <!-- subscription service specific data -->
2140     <targetEPR>

```

```

2141     <wsa:Address>http://docs.oasis-open.org/ws-
2142 rx/wsrn/200608/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsa:Address>
2143     <wsa:Metadata>
2144         <wsp:Policy wsu:Id="MyPolicy">
2145             <wsrmp:RMAssertion/>
2146         </wsp:Policy>
2147     </wsa:Metadata>
2148     </targetEPR>
2149 </sub:Subscribe>
2150 </S:Body>
2151 </S:Envelope>

```

2152 In this example the `subscribe` and `targetEPR` elements are simply examples of what a subscription
2153 request message might contain. Note: the `wsa:Address` element contains the RM anonymous URI
2154 indicating that the notification producer needs to queue the messages until they are requested using the
2155 `MakeConnection` message exchange. The EPR also contains the RM Policy Assertion indicating the RM
2156 must be used when notifications related to this subscription are sent.

2157 **Step 2** – Once the subscription is established, the event consumer checks for a pending message:

```

2158 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2159 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
2160 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2161     <S:Header>
2162         <wsa:Action>http://docs.oasis-open.org/ws-
2163 rx/wsrn/200608/MakeConnection</wsa:Action>
2164         <wsa:To> http://example.org/subscriptionService </wsa:To>
2165     </S:Header>
2166     <S:Body>
2167         <wsrm:MakeConnection>
2168             <wsrm:Address>http://docs.oasis-open.org/ws-
2169 rx/wsrn/200608/anonymous?id=550e8400-e29b-11d4-a716-
2170 446655440000</wsrm:Address>
2171         </wsrm:MakeConnection>
2172     </S:Body>
2173 </S:Envelope>

```

2174 **Step 3** – If there are messages waiting to be delivered then a message will be returned back to the event
2175 consumer. However, because WS-RM is being used to deliver the messages, the first message returned
2176 is a `CreateSequence`:

```

2177 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2178 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200608"
2179 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2180     <S:Header>
2181         <wsa:Action>http://docs.oasis-open.org/ws-
2182 rx/wsrn/200608/CreateSequence</wsa:Action>
2183         <wsa:To>http://docs.oasis-open.org/ws-
2184 rx/wsrn/200608/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsa:To>
2185         <wsa:ReplyTo> http://example.org/subscriptionService </wsa:ReplyTo>
2186         <wsa:MessageID> http://example.org/id-123-456 </wsa:MessageID>
2187     </S:Header>
2188     <S:Body>
2189         <wsrm:CreateSequence>
2190             <wsrm:AcksTo>
2191                 <wsa:Address> http://example.org/subscriptionService </wsa:Address>
2192             </wsrm:AcksTo>
2193         </wsrm:CreateSequence>
2194     </S:Body>

```

2195

```
</S:Envelope>
```

2196 Notice from the perspective of how the RM Source on the event producer interacts with the RM
2197 Destination of those messages, nothing new is introduced by the use of the `MakeConnection`, the use
2198 of RM protocol is the same as the case where the event consumer is addressable.

2199 **Step 4** – The event consumer will respond with a `CreateSequenceResponse` message per normal WS-
2200 Addressing rules:

```
2201 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
2202 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"  
2203 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
2204   <S:Header>  
2205     <wsa:Action>http://docs.oasis-open.org/ws-  
2206 rx/wsmr/200608/CreateSequenceResponse</wsa:Action>  
2207     <wsa:To> http://example.org/subscriptionService </wsa:To>  
2208     <wsa:RelatesTo> http://example.org/id-123-456 </wsa:RelatesTo>  
2209   </S:Header>  
2210   <S:Body>  
2211     <wsmr:CreateSequenceResponse>  
2212       <wsmr:Identifier> http://example.org/rmid-456 </wsmr:Identifier>  
2213     </wsmr:CreateSequenceResponse>  
2214   </S:Body>  
2215 </S:Envelope>
```

2216 Note, this message is carried on an HTTP request directed to the `wsa:ReplyTo` EPR, and the HTTP
2217 response will be an HTTP 202.

2218 **Step 5** – The event consumer checks for another message pending:

```
2219 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
2220 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"  
2221 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
2222   <S:Header>  
2223     <wsa:Action>http://docs.oasis-open.org/ws-  
2224 rx/wsmr/200608/MakeConnection</wsa:Action>  
2225     <wsa:To> http://example.org/subscriptionService </wsa:To>  
2226   </S:Header>  
2227   <S:Body>  
2228     <wsmr:MakeConnection>  
2229       <wsmr:Address>http://docs.oasis-open.org/ws-  
2230 rx/wsmr/200608/anonymous?id=550e8400-e29b-11d4-a716-  
2231 446655440000</wsmr:Address>  
2232     </wsmr:MakeConnection>  
2233   </S:Body>  
2234 </S:Envelope>
```

2235 Notice this is the same message as the one sent in step 2.

2236 **Step 6** – If there is a message pending for this destination then it is returned on the HTTP response:

```
2237 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"  
2238 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200608"  
2239 xmlns:wsa="http://www.w3.org/2005/08/addressing">  
2240   <S:Header>  
2241     <wsa:Action> http://example.org/eventType1 </wsa:Action>  
2242     <wsa:To>http://docs.oasis-open.org/ws-  
2243 rx/wsmr/200608/anonymous?id=550e8400-e29b-11d4-a716-446655440000</wsa:To>
```

```

2244     <wsrm:Sequence>
2245         <wsrm:Identifier> http://example.org/rmid-456 </wsrm:Identifier>
2246     </wsrm:Sequence>
2247     <wsrm:MessagePending pending="true"/>
2248 </S:Header>
2249 <S:Body>
2250     <!-- event specific data -->
2251 </S:Body>
2252 </S:Envelope>

```

2253 As noted in step 3, the use of the RM protocol does not change when using `MakeConnection`. The
 2254 format of the messages, the order of the messages sent and the timing of when to send it remains the
 2255 same.

2256 **Step 7** – At some later interval, or immediately due to the `MessagePending` header's "pending"
 2257 attribute being set to "true", the event consumer will poll again:

```

2258 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
2259 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrp/200608"
2260 xmlns:wsa="http://www.w3.org/2005/08/addressing">
2261     <S:Header>
2262         <wsa:Action>http://docs.oasis-open.org/ws-
2263 rx/wsrp/200608/MakeConnection</wsa:Action>
2264         <wsa:To> http://example.org/subscriptionService </wsa:To>
2265     </S:Header>
2266     <S:Body>
2267         <wsrm:MakeConnection>
2268             <wsrm:Address>http://docs.oasis-open.org/ws-
2269 rx/wsrp/200608/anonymous?id=550e8400-e29b-11d4-a716-
2270 446655440000</wsrm:Address>
2271         </wsrm:MakeConnection>
2272     </S:Body>
2273 </S:Envelope>

```

2274 Notice this is the same message as the one sent in steps 2 and 5. As in steps 3 and 6, the response to
 2275 the `MakeConnection` can be any message destined to the specified Endpoint. This allows the event
 2276 producer to send not only application messages but RM protocol messages (e.g. `CloseSequence`,
 2277 `TerminateSequence` or even additional `CreateSequences`) as needed.

2278 **Step 8** – If at any point in time there are no messages pending, in response to a `MakeConnection` the
 2279 event producer returns an HTTP 202 back to the event consumer. The process then repeats (back to step
 2280 7) until the subscription ends.

2281 Appendix D. State Tables

2282 This appendix specifies the non-normative state transition tables for RM Source and RM Destination.

2283 The state tables describe the lifetime of a sequence in both the RM Source and the RM Destination

2284 Legend:

2285 The first column of these tables contains the motivating event and has the following format:

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

2286 Where:

2287 ● Event Name: indicates the name of the event. Event Names surrounded by "<>" are optional as
2288 described by the specification.

2289 ● [source]: indicates the source of the event; one of:

2290 ● [msg] a Received message

2291 ● [int]: an internal event such as the firing of a timer

2292 ● [app]: the application

2293 ● [unspec]: the source is unspecified

2294 Each event / state combination cell in the tables in this appendix has the following format:

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

2295 Where:

2296 ● action to take: indicates that the state machine performs the following action. Actions surrounded
2297 by "<>" are optional as described by the specification. "Xmit" is used as a short form for the word
2298 "Transmit"

2299 ● [next state]: indicates the state to which the state machine will advance upon the performance of
2300 the action. For ease of reading the next state "same" indicates that the state does not change.

2301 ● {ref} is a reference to the document section describing the behavior in this cell

2302 "N/A" in a cell indicates a state / event combination self-inconsistent with the state machine; should these
2303 conditions occur, it would indicate an implementation error. A blank cell indicates that the behavior is not
2304 described in this specification and does not indicate normal protocol operation. Implementations MAY
2305 generate a Sequence Terminated fault (see section 4.2) in these circumstances. Robust implementations
2306 MUST be able to operate in a stable manner despite the occurrence of unspecified event / state
2307 combinations.

2308 Table 1 RM Source Sequence State Transition Table

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
Create Sequence [unspec] {3.1}	Xmit Create Sequence [Creating] {3.1}	N/A	N/A	N/A	N/A	N/A
Create Sequence Response [msg] {3.1}		Process Create Sequence Response [Created] {3.1}				
Create Sequence Refused Fault [msg] {3.1}		No action [None] {4.6}				
Send message [app] {2.1}	N/A	N/A	Xmit message [Same] {2}	No action [Same] {2}	N/A	N/A
Retransmit of un-ack'd message [int]	N/A	N/A	Xmit message [Same] {2.4}	Xmit message [Same] {2.4}	N/A	N/A
SeqAck (non-final) [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Process Ack ranges [Same] {3.6}	Process Ack ranges [Same] {3.6}	Process Ack ranges [Same] {3.6}	Process Ack ranges [Same] {3.6}
Nack [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	<Xmit message(s)> [Same] {3.6}	<Xmit message(s)> [Same] {3.6}	No action [Same]	No action [Same]
Message Number Rollover Fault [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	No action [Rollover]	No action [Same]	No action [Same]	No action [Same]
<Close Sequence> [int] {3.2}	N/A		Xmit Close Sequence [Closing] {3.2}	N/A	N/A	N/A
Close Sequence Response [msg] {3.2}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}		No action [Closed] {3.2}	No action [Same] {3.2}	No action [Same] {3.2}
SeqAck (final) [msg] {3.6}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Process Ack ranges [Closed] {3.6}	Process Ack ranges [Closed] {3.6}	Process Ack ranges [Same]	Process Ack ranges [Same]
Sequence Closed Fault [msg]	Generate Unknown Sequence Fault	Generate Unknown Sequence Fault	No action [Closed] {4.7}	No action [Closed] {4.7}	No action [Same]	No action [Same]

Events	Sequence States					
	None	Creating	Created	Closing	Closed	Terminating
{4.7}	[Same] {4.3}	[Same] {4.3}				
Unknown Sequence Fault [msg] {4.3}			Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}
Sequence Terminated Fault [msg] {4.2}	N/A		Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}
Terminate Sequence [int]	N/A	No action [None] {unspec}	Xmit Terminate Sequence [Terminating]	Xmit Terminate Sequence [Terminating]	Xmit Terminate Sequence [Terminating]	N/A
Terminate Sequence Response [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}				Terminate Sequence [None] {3.3}
Expires exceeded [int]	N/A	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}
Invalid Acknowledgment [msg] {4.4}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Unknown Sequence Fault [Same] {4.3}	Generate Invalid Acknowledgment Fault [Same] {4.4}			

2309 Table 2 RM Destination Sequence State Transition Table

Events	Sequence States		
	None	Created	Closed
CreateSequence (successful) [msg/int] {3.1}	Xmit Create Sequence Response [Created] {3.1}	N/A	N/A
CreateSequence (unsuccessful) [msg/int] {3.1}	Generate Create Sequence Refused Fault [None] {3.1}	N/A	N/A
Message (with message number within range) [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Accept Message; <Xmit SeqAck> [Same]	Generate Sequence Closed Fault (with SeqAck+Final) [Same] {3.2}
Message (with message number outside of range) [msg]	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Message Number Rollover Fault [Same] {3.4}{4.5}	Generate Sequence Closed Fault (with SeqAck+Final) [Same] {3.2}
<AckRequested> [msg] {3.5}	Generate Unknown Seq Fault [Same] {4.3}	Xmit SeqAck [Same] {3.5}	Xmit SeqAck+Final [Same] {3.6}

Events	Sequence States		
	None	Created	Closed
CloseSequence [msg] {3.2}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit CloseSequence Response with SeqAck+Final [Closed] {3.2}	Generate Sequence Closed Fault [Same] {4.7}
<CloseSequence autonomously> [int]	N/A	No Action [Closed]	N/A
TerminateSequence [msg] {3.3}	Generate Unknown Sequence Fault [Same] {4.3}	Xmit Terminate Sequence Response [None] {3.3}	Xmit Terminate Sequence Response [None] {3.3}
UnknownSequence Fault [msg] {4.3}		Terminate Sequence [None] {4.3}	Terminate Sequence [None] {4.3}
SequenceTerminated Fault [msg] {4.2}		Terminate Sequence [None] {4.2}	Terminate Sequence [None] {4.2}
Invalid Acknowledgement Fault [msg] {4.4}	N/A		
Expires exceeded [int]	N/A	Terminate Sequence [None] {3.4}	Terminate Sequence [None] {3.4}
<Seq Acknowledgement autonomously> [int] {3.6}	N/A	Xmit SeqAck [Same] {3.6}	Xmit SeqAck+Final [Same] {3.6}
Non WSRM message when WSRM required [msg] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}	Generate WSRMRequired Fault [Same] {4.8}

2310 The following two tables apply only if the `MakeConnection` mechanism is utilized.

2311 Table 3 Sending Endpoint Message Transfer Engine

Event	None	Queued n=1	Queued, n>1
Message destined to anon Endpoint when channel unavailable [int] {3.7}	Queue message [Queued n=1]	Queue message [Queued n>1]	Queue message [Queued n>1]
MakeConnection [msg] {3.7}		Send message [none]	Xmit message with MessagePending [if n=2 then (Queued n=1) else (Queued n>1)]

2312 Table 4 Receiving Endpoint Message Transfer Engine

Event	None	Polling
Expectation of unreceived message [int, unspecified]	No Action [Polling]	No Action [Same]
Polling trigger [int, unspecified]		Xmit MakeConnection [Polling] (3.7)

2313 **Appendix E. Acknowledgments**

2314 This document is based on initial contribution to OASIS WS-RX Technical Committee by the following
2315 authors:

2316 Ruslan Bilorusets(BEA), Don Box(Microsoft), Luis Felipe Cabrera(Microsoft), Doug Davis(IBM),
2317 Donald Ferguson(IBM), Christopher Ferris-Editor(BM), Tom Freund(IBM), Mary Ann Hondo(IBM),
2318 John Ibbotson(IBM), Lei Jin(BEA), Chris Kaler(Microsoft), David Langworthy-Editor(Microsoft),
2319 Amelia Lewis(TIBCO Software), Rodney Limprecht(Microsoft), Steve Lucco(Microsoft), Don
2320 Mullen(TIBCO Software), Anthony Nadalin(IBM), Mark Nottingham(BEA), David Orchard(BEA),
2321 Jamie Roots(IBM), Shivajee Samdarshi(TIBCO Software), John Shewchuk(Microsoft), Tony
2322 Storey(IBM).

2323 The following individuals have provided invaluable input into the initial contribution:

2324 Keith Ballinger(Microsoft), Stefan Batres(Microsoft), Rebecca Bergersen(Iona), Allen
2325 Brown(Microsoft), Michael Conner(IBM), George Copeland(Microsoft), Francisco Curbera(IBM),
2326 Paul Fremantle(IBM), Steve Graham(IBM), Pat Helland(Microsoft), Rick Hill(Microsoft), Scott
2327 Hinkelman(IBM), Tim Holloway(IBM), Efim Hudis(Microsoft), David Ingham(Microsoft), Gopal
2328 Kakivaya(Microsoft), Johannes Klein(Microsoft), Frank Leymann(IBM), Martin Nally(IBM), Peter
2329 Niblett(IBM), Jeffrey Schlimmer(Microsoft), James Snell(IBM), Keith Stobie(Microsoft), Satish
2330 Thatte(Microsoft), Stephen Todd(IBM), Sanjiva Weerawarana(IBM), Roger Wolter(Microsoft).

2331 The following individuals were members of the committee during the development of this specification:

2332 Abbie Barbir(Nortel), Charlton Barreto(Adobe), Stefan Batres(Microsoft), Hamid Ben
2333 Malek(Fujitsu), Andreas Bjarlestam(Ericsson), Toufic Boubez(Layer 7), Doug Bunting(Sun), Lloyd
2334 Burch(Novell), Steve Carter(Novell), Martin Chapman(Oracle), Dave Chappell(Sonic), Paul
2335 Cotton(Microsoft), Glen Daniels(Sonic), Doug Davis(IBM), Blake Dournaee(Intel), Jacques
2336 Durand(Fujitsu), Colleen Evans(Microsoft), Christopher Ferris(IBM), Paul Fremantle(WSO2),
2337 Robert Freund(Hitachi), Peter Furniss(Erebor), Marc Goodner(Microsoft), Alastair
2338 Green(Choreology), Mike Grogan(Sun), Ondrej Hrebicek(Microsoft), Kazunori Iwasa(Fujitsu),
2339 Chamikara Jayalath(WSO2), Lei Jin(BEA), Ian Jones(BTplc), Anish Karmarkar(Oracle), Paul
2340 Knight(Nortel), Dan Leshchiner(Tibco), Mark Little(JBoss), Lily Liu(webMethods), Matt
2341 Lovett(IBM), Ashok Malhotra(Oracle), Jonathan Marsh(Microsoft), Daniel Millwood(IBM), Jeff
2342 Mischkinsky(Oracle), Nilo Mitra(Ericsson), Peter Niblett(IBM), Duane Nickull(Adobe), Eisaku
2343 Nishiyama(Hitachi), Dave Orchard(BEA), Chouthri Palanisamy(NEC), Sanjay Patil(SAP), Gilbert
2344 Pilz(BEA), Martin Raepfle(SAP), Eric Rajkovic(Oracle), Stefan Rossmannith(SAP), Tom
2345 Rutt(Fujitsu), Rich Salz(IBM), Shivajee Samdarshi(Tibco), Vladimir Videlov(SAP), Claus von
2346 Riegen(SAP), Pete Wenzel(Sun), Steve Winkler(SAP), Ümit Yalçinalp(SAP), Nobuyuki
2347 Yamamoto(Hitachi).

Appendix F. Revision History

Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.
ws-02	2005-07-21	Doug Davis	i011 (PT0S) added
wd-02	2005-08-16	Anish Karmarkar	Trivial editorial changes
ws-03	2005-09-15	Doug Davis	i019 and i028 (CloseSeq) added
wd-05	2005-09-26	Gilbert Pilz	i005 (Source resend of nacks messages when ack already received) added.
wd-05	2005-09-27	Doug Davis	i027 (InOrder delivery assurance spanning multiple sequences) added
wd-05	2005-09-27	Doug Davis	i020 (Semantics of "At most once" Delivery Assurance) added
wd-05	2005-09-27	Doug Davis	i034 (Fault while processing a piggy-backed RM header) added
wd-05	2005-09-27	Doug Davis	i033 (Processing model of NACKs) added
wd-05	2005-09-27	Doug Davis	i031 (AckRequested schema inconsistency) added
wd-05	2005-09-27	Doug Davis	i025 (SeqAck/None) added
wd-05	2005-09-27	Doug Davis	i029 (Remove dependency on WS-Security) added
wd-05	2005-09-27	Doug Davis	i039 (What does 'have a mU attribute' mean) added
wd-05	2005-09-27	Doug Davis	i040 (Change 'optiona'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrn/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrn:FaultCode element)
wd-06	2005-11-02	Gilbert Pilz	Start wd-06 by changing title page from cd-01.
wd-06	2005-11-03	Gilbert Pilz	i047 (Reorder spec sections)
wd-07	2005-11-17	Gilbert Pilz	Start wd-07
wd-07	2005-11-28	Doug Davis	i071 – except for period in Appendix headings
wd-07	2005-11-28	Doug Davis	i10
wd-07	2005-11-28	Doug Davis	i030
wd-07	2005-11-28	Doug Davis	i037
wd-07	2005-11-28	Doug Davis	i038
wd-07	2005-11-28	Doug Davis	i041
wd-07	2005-11-28	Doug Davis	i043
wd-07	2005-11-28	Doug Davis	i044

Rev	Date	By Whom	What
wd-07	2005-11-28	Doug Davis	i048
wd-07	2005-11-28	Doug Davis	i051
wd-07	2005-11-28	Doug Davis	i053
wd-07	2005-11-28	Doug Davis	i059
wd-07	2005-11-28	Doug Davis	i062
wd-07	2005-11-28	Doug Davis	i063
wd-07	2005-11-28	Doug Davis	i065
wd-07	2005-11-28	Doug Davis	i067
wd-07	2005-11-28	Doug Davis	i068
wd-07	2005-11-28	Doug Davis	i069
wd-07	2005-11-28	Doug Davis	Fix bulleted list (#2) in section 2.3
wd-07	2005-11-29	Gilbert Pilz	i074 (Use of [tcShortName] in artifact locations namespaces, etc)
wd-07	2005-11-29	Gilbert Pilz	i071 – Fixed styles and formatting for TOC. Fixed styles of the appendix headings.
wd-07	2005-11-30	Doug Davis	Removed dup definition of "Receive"
wd-07	2005-11-30	Gilbert Pilz	Fixed lost formatting from heading for Namespace section. Fixed style of text body elements to match OASIS example documents. Fixed tables to match OASIS example documents.
wd-07	2005-12-01	Gilbert Pilz	Updated fix for i074 to eliminate trailing '/'. Added corresponding text around action IRI composition.
wd-07	2005-12-01	Gilbert Pilz	Use non-fixed fields for date values on both title page and body footers.
wd-07	2005-12-01	Doug Davis	Alphabetize the glossary
wd-07	2005-12-02	Doug Davis	i064
wd-07	2005-12-02	Doug Davis	i066
wd-08	2005-12-15	Doug Davis	Add back in RM Source to glossary
wd-08	2005-12-15	Steve Winkler	Doug added Steve's editorial nits
wd-08	2005-12-21	Doug Davis	i050
wd-08	2005-12-21	Doug Davis	i081
wd-08	2005-12-21	Doug Davis	i080 – but i050 negates the need for any changes
wd-08	2005-12-21	Doug Davis	i079
wd-08	2005-12-21	Doug Davis	i076 – didn't add text about "replies" since the RMD to RMS sequence could be used for any message not just replies
wd-08	2005-12-21	Umit Yalcinalp	Action Su03: removed wsse from Table 1
wd-08	2005-12-21	Umit Yalcinalp	i057 per Sunnyvale F2F 2005, Cleaned up some formatting errors in contributors
wd-08	2005-12-27	Doug Davis	i060
wd-08	2005-12-27	Gilbert Pilz	Moved schema and WSDL files to their own artifacts. Converted source document to

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

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

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

Rev	Date	By Whom	What
cd-04	2006-08-11	Doug Davis	Issue 158 applied
cd-04	2006-08-11	Doug Davis	Issue 153 applied
cd-04	2006-08-11	Doug Davis	Issue 156 applied
cd-04	2006-08-15	Gilbert Pilz	More formatting changes for better HTML rendering.

2349 **Appendix G. Notices**

2350 OASIS takes no position regarding the validity or scope of any intellectual property or other rights that
2351 might be claimed to pertain to the implementation or use of the technology described in this document or
2352 the extent to which any license under such rights might or might not be available; neither does it represent
2353 that it has made any effort to identify any such rights. Information on OASIS's procedures with respect to
2354 rights in OASIS specifications can be found at the OASIS website. Copies of claims of rights made
2355 available for publication and any assurances of licenses to be made available, or the result of an attempt
2356 made to obtain a general license or permission for the use of such proprietary rights by implementors or
2357 users of this specification, can be obtained from the OASIS Executive Director.

2358 OASIS invites any interested party to bring to its attention any copyrights, patents or patent applications, or
2359 other proprietary rights which may cover technology that may be required to implement this specification.
2360 Please address the information to the OASIS Executive Director.

2361 Copyright (C) OASIS Open (2006). All Rights Reserved.

2362 This document and translations of it may be copied and furnished to others, and derivative works that
2363 comment on or otherwise explain it or assist in its implementation may be prepared, copied, published and
2364 distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and
2365 this paragraph are included on all such copies and derivative works. However, this document itself may
2366 not be modified in any way, such as by removing the copyright notice or references to OASIS, except as
2367 needed for the purpose of developing OASIS specifications, in which case the procedures for copyrights
2368 defined in the OASIS Intellectual Property Rights document must be followed, or as required to translate it
2369 into languages other than English.

2370 The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors
2371 or assigns.

2372 This document and the information contained herein is provided on an "AS IS" basis and OASIS
2373 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY
2374 WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR
2375 ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.