



1 **Web Services Reliable Messaging**
2 **(WS-Reliable Messaging)**

3 **Working Draft 01, ~~August~~ July 16th 2005**

4 **Document identifier:**

5 WS-ReliableMessaging-1.0draft-01.doc

6 **Location:**

7 <http://docs.oasis-open.org/ws-rx/2005/07/WS-ReliableMessaging-1.0-draft-01.doc>

8 **Editors:**

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

10 TBD

11 **Copyright:**

12 ~~(c) 2002-2005 BEA Systems, IBM, Microsoft Corporation, Inc, and TIBCO Software Inc.~~
13 ~~All rights reserved.~~

14 ~~Permission to copy and display the Web Services Reliable Messaging Policy Assertion~~
15 ~~Specification (the "Specification", which includes schema documents), in any medium~~
16 ~~without fee or royalty is hereby granted, provided that you include the following on ALL~~
17 ~~copies of the Specification that you make:~~

18 ~~1. A link or URL to the Specification at one of the Authors' websites~~

19 ~~2. The copyright notice as shown in the Specification.~~

20 ~~BEA Systems, IBM, Microsoft and TIBCO Software (collectively, the "Authors") each~~
21 ~~agree to grant you a license, under royalty free and otherwise reasonable, non-~~
22 ~~discriminatory terms and conditions, to their respective essential patent claims that they~~
23 ~~deem necessary to implement the Specification.~~

24 ~~THE SPECIFICATION IS PROVIDED "AS IS," AND THE AUTHORS MAKE NO~~
25 ~~REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT~~
26 ~~NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A~~
27 ~~PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS~~
28 ~~OF THE SPECIFICATION ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE~~
29 ~~IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY~~
30 ~~PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.~~

31 ~~THE AUTHORS WILL NOT BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL,~~
32 ~~INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO~~
33 ~~ANY USE OR DISTRIBUTION OF THE SPECIFICATION.~~

34 ~~The name and trademarks of the Authors may NOT be used in any manner, including~~
35 ~~advertising or publicity pertaining to the Specification or its contents without specific,~~
36 ~~written prior permission. Title to copyright in the Specification will at all times remain with~~
37 ~~the Authors.~~

38 ~~No other rights are granted by implication, estoppel or otherwise.~~

39 **Abstract:**

40 This specification (WS-ReliableMessaging) describes a protocol that allows messages
41 to be delivered reliably between distributed applications in the presence of software
42 component, system, or network failures. The protocol is described in this
43 specification in a transport-independent manner allowing it to be implemented using
44 different network technologies. To support interoperable Web services, a SOAP
45 binding is defined within this specification.

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

50 **Composable Architecture:**

51 By using the SOAP [[SOAP](#)] and WSDL [[WSDL](#)] extensibility model, SOAP-based and
52 WSDL-based specifications are designed to be composed with each other to define a
53 rich Web services environment. As such, WS-ReliableMessaging by itself does not
54 define all the features required for a complete messaging solution. WS-
55 ReliableMessaging is a building block that is used in conjunction with other
56 specifications and application-specific protocols to accommodate a wide variety of
57 protocols related to the operation of distributed Web services.

58 **Status:**

59 TBD

60 Table of Contents

61	1INTRODUCTION.....	6
62	1.1GOALS AND REQUIREMENTS.....	6
63	1.1.1Requirements.....	6
64	1.2NOTATIONAL CONVENTIONS.....	6
65	1.3NAMESPACE.....	7
66	1.4COMPLIANCE.....	8
67	2RELIABLE MESSAGING MODEL.....	9
68	2.1GLOSSARY.....	10
69	2.2PROTOCOL PRECONDITIONS.....	11
70	2.3PROTOCOL INVARIANTS.....	11
71	2.4EXAMPLE MESSAGE EXCHANGE.....	11
72	3RM PROTOCOL ELEMENTS.....	14
73	3.1SEQUENCES.....	14
74	3.2SEQUENCE ACKNOWLEDGEMENT.....	16
75	3.3REQUEST ACKNOWLEDGEMENT.....	18
76	3.4SEQUENCE CREATION.....	19
77	3.5SEQUENCE TERMINATION.....	23
78	4FAULTS.....	24
79	4.1SEQUENCEFAULT ELEMENT.....	26
80	4.2SEQUENCE TERMINATED.....	27
81	4.3UNKNOWN SEQUENCE.....	27
82	4.4INVALID ACKNOWLEDGEMENT.....	27
83	4.5MESSAGE NUMBER ROLLOVER.....	28
84	4.6LAST MESSAGE NUMBER EXCEEDED.....	28
85	4.7CREATE SEQUENCE REFUSED.....	29
86	5SECURITY CONSIDERATIONS.....	30
87	6REFERENCES.....	32
88	6.1NORMATIVE.....	32
89	6.2NON-NORMATIVE.....	33
90	APPENDIX A.SCHEMA	34

91	APPENDIX B.MESSAGE EXAMPLES.....	39
92	B.1.CREATE SEQUENCE.....	40
93	B.2. INITIAL TRANSMISSION.....	42
94	B.3.FIRST ACKNOWLEDGEMENT.....	44
95	B.4.RETRANSMISSION.....	45
96	B.5.TERMINATION.....	46
97	APPENDIX C.WSDL.....	48
98	APPENDIX D.ACKNOWLEDGMENTS.....	50
99	APPENDIX E.REVISION HISTORY.....	51
100	APPENDIX F.NOTICES.....	52

101 **1 Introduction**

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

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

113 **1.1 Goals and Requirements**

114 **1.1.1 Requirements**

115 **1.2 Notational Conventions**

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

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

- 121 • The syntax appears as an XML instance, but values in italics indicate data types instead
122 of values.
- 123 • Characters are appended to elements and attributes to indicate cardinality:
 - 124 ○ "?" (0 or 1)
 - 125 ○ "*" (0 or more)
 - 126 ○ "+" (1 or more)
- 127 • The character "|" is used to indicate a choice between alternatives.
- 128 • The characters "[" and "]" are used to indicate that contained items are to be treated as a
129 group with respect to cardinality or choice.

- 130 • An ellipsis (i.e. "...") indicates a point of extensibility that allows other child, or attribute,
131 content. Additional children and/or attributes MAY be added at the indicated extension
132 points but MUST NOT contradict the semantics of the parent and/or owner, respectively.
133 If an extension is not recognized it SHOULD be ignored.
- 134 • XML namespace prefixes (See Section [Namespace](#)) are used to indicate the namespace
135 of the element being defined.

136 1.3 Namespace

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

139 <http://schemas.xmlsoap.org/ws/2005/02/rm/policy>

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

142 The following namespaces are used in this document:

143 *Table 1*

Prefix	Namespace
S	http://www.w3.org/2003/05/soap-envelope
S11	http://schemas.xmlsoap.org/soap/envelope/
wstrm	http://schemas.xmlsoap.org/ws/2005/02/rm
wsa	http://schemas.xmlsoap.org/ws/2004/08/addressing
wsse	http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd
xs	http://www.w3.org/2001/XMLSchema

144 The normative schema for WS-Reliable Messaging can be found at:

145 <http://schemas.xmlsoap.org/ws/2005/02/rm/wstrm.xsd>

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

148 If an action URI is used, and one is not already defined per the rules of the WS-
149 Addressing specification [WS-Addressing], then the action URI MUST consist of the
150 reliable messaging namespace URI concatenated with the "/" character and the
151 element name. For example:

152 <http://schemas.xmlsoap.org/ws/2005/02/rm/SequenceAcknowledgement>

153 **1.4 Compliance**

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

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

162 2 Reliable Messaging Model

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

165 WS-ReliableMessaging provides an interoperable protocol that a Reliable Messaging
166 (RM) Source and Reliable Messaging (RM) Destination use to provide Application
167 Source and Destination a guarantee that a message that is sent will be delivered.
168 The guarantee is specified as a delivery assurance. The protocol supports the
169 endpoints in providing these delivery assurances. It is the responsibility of the RM
170 Source and RM Destination to fulfill the delivery assurances, or raise an error. The
171 protocol defined here allows endpoints to meet this guarantee for the delivery
172 assurances defined below.

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

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

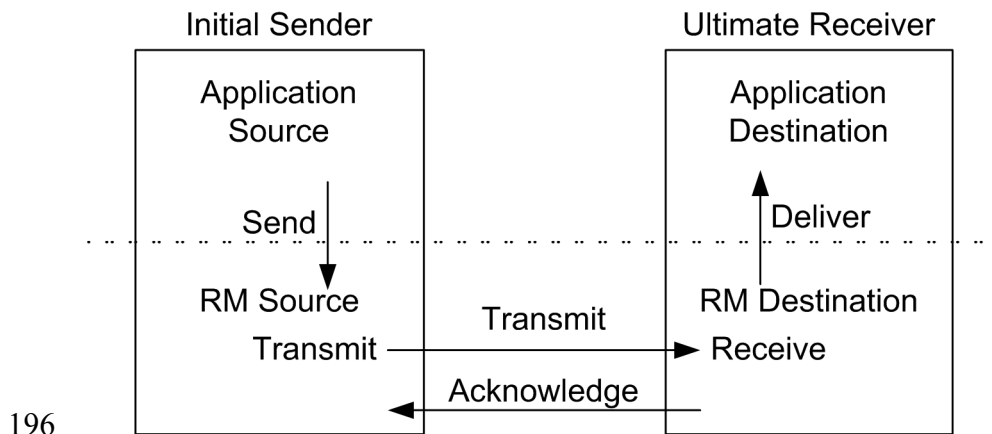
177 **AtMostOnce** Messages will be delivered at most once without duplication or an error
178 will be raised on at least one endpoint. It is possible that some messages in a
179 sequence may not be delivered.

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

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

185 **InOrder** Messages will be delivered in the order that they were sent. This delivery
186 assurance may be combined with any of the above delivery assurances. It requires
187 that the sequence observed by the ultimate receiver be non-decreasing. It says
188 nothing about duplications or omissions.

189 [Figure 1](#) ~~The diagram~~ below illustrates the entities and events in a simple reliable
190 message exchange. First, the Application Source Sends a message for reliable
191 delivery. The Reliable Messaging (RM) Source accepts the message and Transmits it
192 one or more times. After receiving the message, the RM Destination Acknowledges
193 it. Finally, the RM Destination delivers the message to the Application Destination.
194 The exact roles the entities play and the complete meaning of the events will be
195 defined throughout this specification.



197 Figure 1: Reliable Messaging Model

198 2.1 Glossary

199 The following definitions are used throughout this specification:

200 **Endpoint:** A referencable entity, processor, or resource where Web service messages
201 are originated or targeted.

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

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

204 **Delivery Assurance:** The guarantee that the messaging infrastructure provides on
205 the delivery of a message.

206 **RM Source:** The endpoint that transmits the message.

207 **RM Destination:** The endpoint that receives the message.

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

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

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

213 **Receive:** The act of reading a message from a network connection.

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

216 **2.2 Protocol Preconditions**

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

- 219 • The RM Source MUST have an endpoint reference that uniquely identifies the RM Destination
220 endpoint; correlations across messages addressed to the unique endpoint MUST be
221 meaningful.
- 222 • The RM Source MUST have knowledge of the destination's policies, if any, and the RM
223 Source MUST be capable of formulating messages that adhere to this policy.

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

226 **2.3 Protocol Invariants**

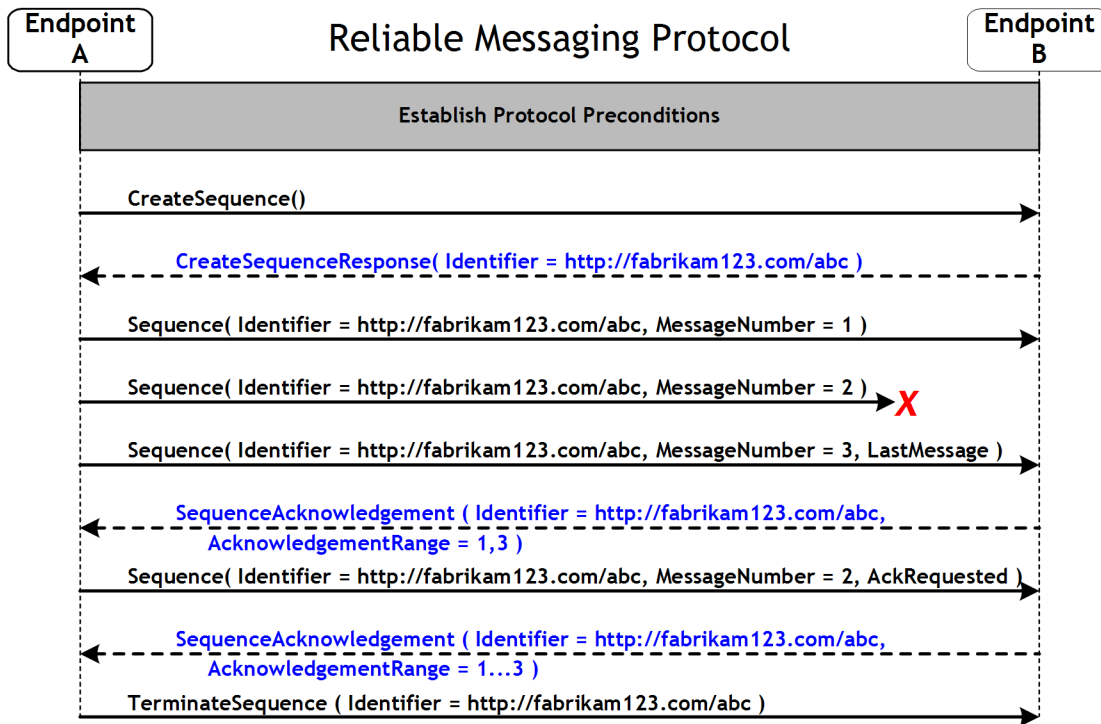
227 During the lifetime of the protocol, two invariants are REQUIRED for correctness:

- 228 • The RM Source MUST assign each reliable message a sequence number (defined below)
229 beginning at 1 and increasing by exactly 1 for each subsequent reliable message.

230 Every acknowledgement issued by the RM Destination MUST include within an
231 acknowledgement range or ranges the sequence number of every message
232 successfully received by the RM Destination and MUST exclude sequence numbers of
233 any messages not yet received.

234 **2.4 Example Message Exchange**

235 ~~The following~~ Figure 2 illustrates a possible message exchange between two reliable
236 messaging endpoints [A and B](#).



237 Figure 2: The WS-ReliableMessaging Protocol

- 238 1. The protocol preconditions are established. These include policy exchange,
 239 endpoint resolution, establishing trust.
- 240 2. The RM Source requests creation of a new Sequence.
- 241 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 242 4. The RM Source begins sending messages beginning with MessageNumber 1. In
 243 the figure the RM Source sends 3 messages.
- 244 5. Since the 3rd message is the last in this exchange, the RM Source includes a
 245 ~~last~~<wsrm>LastMessage> token.
- 246 6. The 2nd message is lost in transit.
- 247 7. The RM Destination acknowledges receipt of message numbers 1 and 3 in
 248 response to the RM Source's ~~last~~<wsrm>LastMessage> token.
- 249 8. The RM Source retransmits the 2nd message. This is a new message on the
 250 underlying transport, but since it has the same sequence identifier and message
 251 number so the RM Destination can recognize it as equivalent to the earlier
 252 message, in case both are received.

- 253 9. The RM Source includes an ~~<AckR~~[<wsrm:AckRequested>](#) element so the RM
254 Destination will expedite an acknowledgement.
- 255 10. The RM Destination receives the second transmission of the message with
256 MessageNumber 2 and acknowledges receipt of message numbers 1, 2, and 3
257 which carried the ~~<Last~~[<wsrm:LastMessage>](#) token.
- 258 11. The RM Source receives this acknowledgement and sends a TerminateSequence
259 message to the RM Destination indicating that the sequence is completed and
260 reclaims any resources associated with the Sequence.
- 261 12. The RM Destination receives the TerminateSequence message indicating that the
262 RM Source will not be sending any more messages, and reclaims any resources
263 associated with the Sequence.
- 264 Now that the basic model has been outlined, the details of the elements used in this
265 protocol are now provided [in Section 3](#).

266 3 RM Protocol Elements

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

271 3.1 Sequences

272 The RM protocol uses a `<Sequence<wsrm:Sequence>` header block to track and
273 manage the reliable delivery of messages. Messages for which the delivery
274 assurance applies MUST contain a `<Sequence<wsrm:Sequence>` header block. Each
275 Sequence MUST have a unique `<Identifier<wsrm:Identifier>` element and each
276 message within a Sequence MUST have a `<Message<wsrm:MessageNumber>` element
277 that increments by 1 from an initial value of 1. These values are contained within a
278 `<Sequence<wsrm:Sequence>` header block accompanying each message being
279 delivered in the context of a Sequence. In addition to mandatory
280 `<Identifier<wsrm:Identifier>` and `<Message<wsrm:MessageNumber>` elements,
281 the header MAY include a `<Last<wsrm>LastMessage>` element.

282 There MUST be no more than one `<Sequence<wsrm:Sequence>` header block in any
283 message.

284 The purpose of the `<Last<wsrm>LastMessage>` element is to signal to the RM
285 Destination that the message represents the last message in the Sequence.

286 A following exemplar defines its syntax:

```
287 <wsrm:Sequence ...>  
288   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
289   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>  
290   <wsrm>LastMessage/>?  
291   ...  
292 </wsrm:Sequence>
```

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

294 /wsrm:Sequence

295 This is the element containing Sequence information for WS-ReliableMessaging. The
296 `<wsrm:Sequence>` element MUST be understood by the RM Destination. The `<wsrm:Sequence>`
297 element MUST have a `mustUnderstand` attribute from the namespace corresponding to the
298 version of SOAP to which the `<wsrm:Sequence>` SOAP header block is bound.

299 /wsrm:Sequence/wsrm:Identifier

300 This required element MUST contain an absolute URI conformant with RFC2396 that uniquely
301 identifies the Sequence.

302 /wsrm:Sequence/wsrm:Identifier/@{any}

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

305 /wsrm:Sequence/wsrm:MessageNumber

306 This required element MUST contain an [xs:unsignedLong](#) representing the ordinal position of the
307 message within a Sequence. Sequence MessageNumbers start at 1 and monotonically increase
308 throughout the Sequence. If the message number exceeds the internal limitations of an RM
309 Source or RM Destination or reaches the maximum value of an [xs:unsignedLong](#)
310 (18,446,744,073,709,551,615), the RM Source or Destination MUST issue a
311 MessageNumberRollover fault.

312 /wsrm:Sequence/wsrm:LastMessage

313 This element MAY be included by the RM Source endpoint. The [<Last<wsrm:LastMessage>](#)
314 element has no content.

315 /wsrm:Sequence/{any}

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

318 /wsrm:Sequence/@{any}

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

321 A RM Source endpoint MUST include a [<Last<wsrm:LastMessage>](#) element in the
322 [<Sequence<wsrm:Sequence>](#) element for the last message in a Sequence. An RM
323 Destination endpoint MUST respond with a
324 [<Sequence<wsrm:SequenceAcknowledgement>](#) upon receipt of a
325 [<Last<wsrm:LastMessage>](#) element. A Sequence MUST NOT use a
326 [<Message<wsrm:MessageNumber>](#) value greater than that which accompanies a
327 [<Last<wsrm:LastMessage>](#) element. An RM Destination MUST generate a
328 LastMessageNumberExceeded ([See Section 4.6](#)) fault upon receipt of such a
329 message. In the event that an RM Source needs to close a Sequence and there is no
330 application message, the RM Source MAY send a message with an empty body
331 containing [<Sequence<wsrm:Sequence>](#) header with the [<Last<wsrm:LastMessage>](#)
332 element. In this usage, the action URI MUST be:

333 <http://schemas.xmlsoap.org/ws/2005/02/rm/LastMessage>

334 in preference to the pattern defined in Section 1.2.

335 The following example illustrates a Sequence header block.

```
336 <wsrm:Sequence>
337 <wsrm:Identifier>http://fabrikam123example.com/abc</wsrm:Identifier>
338 <wsrm:MessageNumber>10</wsrm:MessageNumber>
339 <wsrm:LastMessage/>
340 </wsrm:Sequence>
```

341 3.2 Sequence Acknowledgement

342 The RM Destination informs the RM Source of successful message receipt using a
343 `<Sequence wsrm:SequenceAcknowledgement>` header block. The
344 `<Sequence wsrm:SequenceAcknowledgement>` header block MAY be transmitted
345 independently or included on return messages. The RM Destination MAY send a
346 `<Sequence wsrm:SequenceAcknowledgement>` header block at any point. The timing
347 of acknowledgements can be advertised using policy and acknowledgements can be
348 explicitly requested using the `<AckR wsrm:AckRequested>` directive (see Section
349 3.3).

350 The following exemplar defines its syntax:

```
351 <wsrm:SequenceAcknowledgement ...>
352 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
353 [ <wsrm:AcknowledgementRange ...
354 Upper="xs:unsignedLong"
355 Lower="xs:unsignedLong"/> +
356 | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> + ]
357 ...
358 </wsrm:SequenceAcknowledgement>
```

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

361 `/wsrm:SequenceAcknowledgement`

362 This element contains the Sequence acknowledgement information.

363 `/wsrm:SequenceAcknowledgement/wsrm:Identifier`

364 This required element MUST contain an absolute URI conformant with RFC2396 that uniquely
365 identifies the Sequence.

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

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

369 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange
370 This optional element, if present, can occur 1 or more times. It contains a range of message
371 Sequence MessageNumbers successfully received by the receiving endpoint manager. The
372 ranges SHOULD NOT overlap. This element MUST NOT be present if [<Nack<wsrm:Nack>](#) is
373 also present as a child of [<Sequence<wsrm:SequenceAcknowledgement>](#).

374 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper
375 This required attribute contains an [xs:unsignedLong](#) representing the
376 [<Message<wsrm:MessageNumber>](#) of the highest contiguous message in a Sequence range
377 [received by the RM Destination](#).

378 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower
379 This required attribute contains an [xs:unsignedLong](#) representing the
380 [<Message<wsrm:MessageNumber>](#) of the lowest contiguous message in a Sequence range
381 [received by the RM Destination](#).

382 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@{any}
383 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
384 to the element.

385 /wsrm:SequenceAcknowledgement/wsrm:Nack
386 This optional element, if present, MUST contain an [xs:unsignedLong](#) representing the
387 [<Message<wsrm:MessageNumber>](#) of an unreceived message in a Sequence. This element
388 MUST NOT be present if the [<Ack<wsrm:AcknowledgementRange>](#) is also present as a child
389 of [<Sequence<wsrm:SequenceAcknowledgement>](#). The [<Nack<wsrm:Nack>](#) element
390 permits the gap analysis of the [<Ack<wsrm:AcknowledgementRange>](#) elements to be
391 performed at the RM Destination rather than at the RM Source which may yield performance
392 benefits in certain environments.

393 /wsrm:SequenceAcknowledgement/{any}
394 This is an extensibility mechanism to allow different (extensible) types of information, based on a
395 schema, to be passed.

396 /wsrm:SequenceAcknowledgement/@{any}
397 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
398 to the element.

399 The following examples illustrate [<Sequence<wsrm:SequenceAcknowledgement>](#)
400 elements:

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

402 [<wsrm:SequenceAcknowledgement>](#)

```
403 <wsrm:Identifier>http://fabrikam123example.com/abc</wsrm:Identifier>
404     <wsrm:AcknowledgementRange Upper="10" Lower="1"/>
405 </wsrm:SequenceAcknowledgement>
```

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

```
408 <wsrm:SequenceAcknowledgement>
409 <wsrm:Identifier>http://fabrikam123example.com/abc</wsrm:Identifier>
410     <wsrm:AcknowledgementRange Upper="2" Lower="1"/>
411     <wsrm:AcknowledgementRange Upper="6" Lower="4"/>
412     <wsrm:AcknowledgementRange Upper="10" Lower="8"/>
413 </wsrm:SequenceAcknowledgement>
```

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

```
415 <wsrm:SequenceAcknowledgement>
416 <wsrm:Identifier>http://fabrikam123example.com/abc</wsrm:Identifier>
417     <wsrm:Nack>3</wsrm:Nack>
418 </wsrm:SequenceAcknowledgement>
```

419 3.3 Request Acknowledgement

420 The purpose of the `<AckR<wsrm:AckR>requested>` header block is to signal to the RM
421 Destination that the RM Source is requesting that a
422 `<Sequence<wsrm:Sequence>Acknowledgement>` be returned.

423 At any time, the RM Source may request an acknowledgement message from the RM
424 Destination `endpoint` using an `<AckR<wsrm:AckR>requested>` header block.

425 The RM Source endpoint requests this acknowledgement by including an
426 `<AckR<wsrm:AckR>requested>` header block in the message. An RM Destination that
427 receives a message that contains an `<AckR<wsrm:AckR>requested>` header block MUST
428 respond with a message containing a `<Sequence<wsrm:Sequence>Acknowledgement>`
429 header block.

430 The following exemplar defines its syntax:

```
431 <wsrm:AckRequested ...>
432     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
433     <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber> ?
434     ...
```

435 `</wsrm:AckRequested>`

436 `/wsrm:AckRequested`

437 This element requests an acknowledgement for the identified sequence.

438 `/wsrm:AckRequested/wsrm:Identifier`

439 This required element MUST contain an absolute URI, conformant with RFC2396, that uniquely
440 identifies the Sequence to which the request applies.

441 `/wsrm:AckRequested/wsrm:Identifier/@{any}`

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

444 `/wsrm:AckRequested/wsrm:MessageNumber`

445 This optional element, if present, MUST contain an xs:unsignedLong representing the highest
446 `<Message<wsrm:MessageNumber>` sent by the RM Source within [thea](#) Sequence. If present, it
447 MAY be treated as a hint to the RM Destination as an optimization to the process of preparing to
448 transmit a `<Sequence<wsrm:SequenceAcknowledgement>`.

449 `/wsrm:AckRequested/{any}`

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

452 `/wsrm:AckRequested/@{any}`

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

455 **3.4 Sequence Creation**

456 The RM Source MUST request creation of an outbound Sequence by sending a
457 `<CreateS<wsrm:CreateSequence>` element in the body of a message to the RM
458 Destination which in turn responds either with a
459 `<CreateS<wsrm:CreateSequenceResponse>` or a `CreateSequenceRefused` fault in
460 the body of the response message. `<CreateS<wsrm:CreateSequence>` MAY carry an
461 offer to create an inbound sequence which is either accepted or rejected in the
462 `<CreateS<wsrm:CreateSequenceResponse>`.

463 The RM Destination of the outbound sequence is the WS-Addressing
464 EndpointReference [\[WS-Addressing\]](#) to which `<CreateS<wsrm:CreateSequence>` is
465 sent. The RM Destination of the inbound sequence is the WS-Addressing
466 `<wsa:ReplyTo>` of the `<CreateS<wsrm:CreateSequence>`.

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

```

468 <wsmr:CreateSequence ...>
469   <wsmr:AcksTo ...> wsa:EndpointReferenceType </wsmr:AcksTo>
470   <wsmr:Expires ...> xs:duration </wsmr:Expires> ?
471   <wsmr:Offer ...>
472     <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>
473     <wsmr:Expires ...> xs:duration </wsmr:Expires> ?
474     ...
475   </wsmr:Offer> ?
476   ...
477   <wsse:SecurityTokenReference>
478     ...
479   </wsse:SecurityTokenReference> ?
480   ...
481 </wsmr:CreateSequence>

```

482 /wsmr:CreateSequence

483 This element requests creation of a new Sequence between the RM Source that sends it, and the
484 RM Destination to which it is sent. This element MUST NOT be sent as a header block. The RM
485 Destination MUST respond either with a [CreateSequenceResponse](#)
486 response message or a `CreateSequenceRefused` fault.

487 /wsmr:CreateSequence/wsmr:AcksTo

488 This required element, of type `wsa:EndpointReferenceType` as specified by WS-Addressing [WS-
489 [AddressingDDR](#)] specifies the endpoint reference to which
490 [SequenceAcknowledgement](#) messages and faults related to the created
491 Sequence are to be sent.

492 /wsmr:CreateSequence/wsmr:Expires

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

497 /wsmr:CreateSequence/wsmr:Expires/@{any}

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

500 /wsmr:CreateSequence/wsmr:Offer

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

503 /wsmr:CreateSequence/wsmr:Offer/wsmr:Identifier

504 This required element MUST contain an absolute URI conformant with RFC2396 that uniquely
505 identifies the offered Sequence.

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

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

509 /wsmr:CreateSequence/wsmr:Offer/wsmr:Expires

510 This element, if present, of type `xs:duration` specifies the duration for the Sequence. A value
511 of 'PT0S' indicates that the Sequence will never expire. Absence of the element indicates an
512 implied value of 'PT0S'.

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

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

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

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

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

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

522 /wsmr:CreateSequence/wsse:SecurityTokenReference

523 This optional element uses the extensibility mechanism defined next to communicate an explicit
524 reference to the security token to be used to authorize messages for the created outbound
525 Sequence and if offered the inbound Sequence, using a `<wsse:SecurityTokenReference>`
526 as documented in WS-Security [WSSecurity]. All subsequent messages in the outbound
527 Sequence and if offered the inbound Sequence MUST demonstrate proof-of-possession of the
528 referenced key.

529 /wsmr:CreateSequence/{any}

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

532 /wsmr:CreateSequence/@{any}

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

535 A `<CreateS<wsmr:CreateSequenceResponse>` is sent in the body of a response
536 message by an RM Destination in response to receipt of a
537 `<CreateS<wsmr:CreateSequence>` request message. It carries the

538 `<Identifier<wsmr:Identifier>` of the created Sequence and indicates that the RM
539 Source may begin sending messages in the context of the identified Sequence.

540 The following exemplar defines the `<CreateS<wsmr:CreateSequenceResponse>`
541 syntax:

```
542 <wsmr:CreateSequenceResponse ...>  
543   <wsmr:Identifier ...> xs:anyURI </wsmr:Identifier>  
544   <wsmr:Expires> xs:duration </wsmr:Expires> ?  
545   <wsmr:Accept ...>  
546     <wsmr:AcksTo ...> wsa:EndpointReferenceType </wsmr:AcksTo>  
547     ...  
548   </wsmr:Accept> ?  
549   ...  
550 </wsmr:CreateSequenceResponse>
```

551 `/wsmr:CreateSequenceResponse`

552 This element is sent in the body of the response message in response to a
553 `<CreateS<wsmr:CreateSequence>` request message. It indicates that the RM Destination
554 has created a new Sequence at the request of the RM Source. This element MUST NOT be sent
555 as a header block.

556 `/wsmr:CreateSequenceResponse/wsmr:Identifier`

557 This required element MUST contain an absolute URI conformant with RFC2396 of the Sequence
558 that has been created by the RM Destination.

559 `/wsmr:CreateSequenceResponse/wsmr:Identifier/@{any}`

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

562 `/wsmr:CreateSequenceResponse/wsmr:Expires`

563 This element, if present, of type `xs:duration` accepts or refines the RM Source's requested
564 duration for the Sequence. A value of 'PT0S' indicates that the Sequence will never expire.
565 Absence of the element indicates an implied value of 'PT0S'. This value MUST be equal or lesser
566 than the value requested by the RM Source in the corresponding
567 `<CreateS<wsmr:CreateSequence>` message.

568 `/wsmr:CreateSequenceResponse/wsmr:Expires/@{any}`

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

571 `/wsmr:CreateSequenceResponse/wsmr:Accept`

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

574 This element MUST be present if the corresponding `<CreateSequence wsrn:CreateSequence>`
575 message contained an `<Offer wsrn:Offer>` element.

576 `/wsrm:CreateSequenceResponse/wsrn:Accept/wsrn:AcksTo`

577 This required element, of type `wsa:EndpointReferenceType` as specified by WS-Addressing [WS-
578 Addressing], specifies the endpoint reference to which

579 `<Sequence wsrn:SequenceAcknowledgement>` messages related to the accepted

580 Sequence are to be sent.

581 `/wsrm:CreateSequenceResponse/wsrn:Accept/{any}`

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

584 `/wsrm:CreateSequenceResponse/wsrn:Accept/@{any}`

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

587 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

593 3.5 Sequence Termination

594 After an RM Source receives the `<Sequence wsrn:SequenceAcknowledgement>`
595 acknowledging the complete range of messages in a Sequence, it sends a

596 `<TerminateSequence wsrn:TerminateSequence>` element, in the body of a message to the

597 RM Destination to indicate that the Sequence is complete, and that it will not be

598 sending any further messages related to the Sequence. The RM Destination can

599 safely reclaim any resources associated with the Sequence upon receipt of the

600 `<TerminateSequence wsrn:TerminateSequence>` message.

601 The following exemplar defines the TerminateSequence syntax:

```
602 <wsrm:TerminateSequence ...>  
603   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
604   ...  
605 </wsrm:TerminateSequence>
```

606 `/wsrm:TerminateSequence`

607 This element is sent by an RM Source after it has received the final
608 ~~<Sequence~~<wsm:SequenceAcknowledgement> covering the full range of a Sequence. It
609 indicates that the RM Destination can safely reclaim any resources related to the identified
610 Sequence. This element MUST NOT be sent as a header block.

611 /wsm:TerminateSequence/wsm:Identifier
612 This required element MUST contain an absolute URI conformant with RFC2396 of the Sequence
613 that is being terminated.

614 /wsm:TerminateSequence/wsm:Identifier/@{any}
615 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
616 to the element.

617 /wsm:TerminateSequence/{any}
618 This is an extensibility mechanism to allow different (extensible) types of information, based on a
619 schema, to be passed.

620 /wsm:TerminateSequence/@{any}
621 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added
622 to the element.

623 4 Faults

624 The fault definitions defined in this section reference certain abstract properties, such
625 as [fault endpoint], that are defined in section 3 of the WS-Addressing [WS-
626 Addressing] specification. Endpoints compliant with this specification MUST include
627 required **Mmessage Addressinginformation Propertiesheaders** on all fault messages.

628 Sequence creation uses a CreateSequence, CreateSequenceResponse request-
629 **response pattern-reply**. Faults for this operation are treated as defined in WS-
630 Addressing. CreateSequenceRefused is a possible fault reply for this operation.
631 UnknownSequence is a fault generated by endpoints when messages carrying RM
632 header blocks targeted at unrecognized sequences are detected, these faults are also
633 treated as defined in WS-Addressing. All other faults in this section relate to the
634 processing of RM header blocks targeted at known sequences and are collectively
635 referred to as sequence faults. Sequence faults SHOULD be sent to the same
636 [destination] as **<Sequence<wsrm:SequenceAcknowledgement>** messages. These
637 faults are correlated using the **Ssequence** identifier carried in the detail.

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

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

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

644 The definitions of faults use the following properties:

645 [Code] The fault code.

646 [Subcode] The fault subcode.

647 [Reason] The English language reason element.

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

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

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

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

WS-ReliableMessaging-v1%5B1%5D.0-wd-01.sxw

8/16/2005

Copyright © OASIS Open 2005. All Rights Reserved.

Page 25 of 53

```

652 <S:Envelope>
653   <S:Header>
654     <wsa:Action>
655       http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
656     </wsa:Action>
657     <!-- Headers elided for clarity. -->
658   </S:Header>
659   <S:Body>
660     <S:Fault>
661       <S:Code>
662         <S:Value> [Code] </S:Value>
663         <S:Subcode>
664           <S:Value> [Subcode] </S:Value>
665         </S:Subcode>
666       </S:Code>
667       <S:Reason>
668         <S:Text xml:lang="en"> [Reason] </S:Text>
669       </S:Reason>
670       <S:Detail>
671         [Detail]
672         ...
673       </S:Detail>
674     </S:Fault>
675   </S:Body>
676 </S:Envelope>

```

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

```

679 <S11:Envelope>
680   <S11:Header>
681     <wsrm:SequenceFault>
682       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
683       ...
684     </wsrm:SequenceFault>
685     <!-- Headers elided for clarity. -->
686   </S11:Header>
687   <S11:Body>
688     <S11:Fault>
689       <faultcode> [Code] </faultcode>
690       <faultstring> [Reason] </faultstring>
691     </S11:Fault>
692   </S11:Body>

```

693 </S11:Envelope>

694 The properties bind to a SOAP 1.1 fault as follows when the fault is generated as a
695 result of processing a <CreateS<wsm:CreateSequence> request message:

```
696 <S11:Envelope>
697   <S11:Body>
698     <S11:Fault>
699       <faultcode> [Subcode] </faultcode>
700       <faultstring xml:lang="en"> [Reason] </faultstring>
701     </S11:Fault>
702   </S11:Body>
703 </S11:Envelope>
```

704 4.1 SequenceFault Element

705 The purpose of the <Sequence<wsm:SequenceFault> element is to carry the specific
706 details of a fault generated during the reliable messaging specific processing of a
707 message belonging to a Sequence. The <Sequence<wsm:SequenceFault> container
708 MUST only be used in conjunction with the SOAP1.1 fault mechanism. It MUST NOT
709 be used in conjunction with the SOAP1.2 binding.

710 The following exemplar defines its syntax:

```
711 <wsm:SequenceFault ...>
712   <wsm:FaultCode> wsm:FaultCodes </wsm:FaultCode>
713   ...
714 </wsm:SequenceFault>
```

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

716 /wsm:SequenceFault

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

718 /wsm:SequenceFault/wsm:FaultCode

719 This element, if present, MUST contain a qualified name from the set of fault codes defined
720 below.

721 /wsm:SequenceFault/{any}

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

724 /wsm:SequenceFault/@{any}

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

727 **4.2 Sequence Terminated**

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

733 Properties:

734 [Code] Sender or Receiver

735 [Subcode] wsrn:SequenceTerminated

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

737 [Detail]

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

739 **4.3 Unknown Sequence**

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

742 Properties:

743 [Code] Sender

744 [Subcode] wsrn:UnknownSequence

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

746 [Detail]

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

748 **4.4 Invalid Acknowledgement**

749 This fault is sent by the RM Source in response to a
750 `<SequenceAcknowledge>` that violates the cumulative
751 acknowledgement invariant. An example of such a violation would be a
752 `SequenceAcknowledge` covering messages that have not been sent.

753 [Code] Sender

754 [Subcode] wsrn:InvalidAcknowledgement

755 [Reason] The SequenceAcknowledgement violates the cumulative acknowledgement
756 invariant.

757 [Detail]

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

759 **4.5 Message Number Rollover**

760 This fault is sent to indicate that message numbers for a sequence have been
761 exhausted. It is an unrecoverable error and terminates the Sequence.

762 Properties:

763 [Code] Sender

764 [Subcode] wsrn:MessageNumberRollover

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

766 [Detail]

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

768 **4.6 Last Message Number Exceeded**

769 This fault is sent by an RM Destination to indicate that it has received a message that
770 has a `<Message<wsrm:MessageNumber>` within a Sequence that exceeds the value of
771 the `<Message<wsrm:MessageNumber>` element that accompanied a
772 `<Last<wsrm>LastMessage>` element for the Sequence. This is an unrecoverable error
773 and terminates the Sequence.

774 Properties:

775 [Code] Sender

776 [Subcode] wsrn:LastMessageNumberExceeded

777 [Reason] The value for wsrn:MessageNumber exceeds the value of the
778 MessageNumber accompanying a LastMessage element in this Sequence.

779 [Detail]

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

781 **4.7 Create Sequence Refused**

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

783 Properties:

784 [Code] Sender

785 [Subcode] wsrn:CreateSequenceRefused

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

787 [Detail] empty

788 **5 Security Considerations**

789 It is strongly recommended that the communication between services be secured
790 using the mechanisms described in WS-Security. In order to properly secure
791 messages, the body and all relevant headers need to be included in the signature.
792 Specifically, the `<wsrm:Sequence>` header needs to be signed with the body in order
793 to "bind" the two together. The `<wsrm:SequenceAcknowledgement>` header may be
794 signed independently because a reply independent of the message is not a security
795 concern.

796 Because Sequences are expected to exchange a number of messages, it is
797 recommended that a security context be established using the mechanisms described
798 in WS-Trust and WS-SecureConversation. If a Sequence is bound to a specific
799 endpoint, then the security context needs to be established or shared with the
800 endpoint servicing the Sequence. While the context can be established at any time,
801 it is critical that the messages establishing the Sequence be secured even if they
802 precede security context establishment. However, it is recommended that the
803 security context be established first. Security contexts are independent of reliable
804 messaging Sequences. Consequently, security contexts can come and go
805 independent of the lifetime of the Sequence. In fact, it is recommended that the
806 lifetime of a security context be less than the lifetime of the Sequence unless the
807 Sequence is very short-lived.

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

- 813 • Closing and re-establishing a security context
- 814 • Exchanging new secrets between the parties
- 815 • Using a derived key sequence and switch "generations"
- 816 • Attaching a nonce to each message and using it in a derived key function with the shared
817 secret

818 The security context may be re-established using the mechanisms described in WS-
819 Trust and WS-SecureConversation. Similarly, secrets can be exchanged using the
820 mechanisms described in WS-Trust. Note, however, that the current shared secret
821 should not be used to encrypt the new shared secret. Derived keys, the preferred
822 solution from this list, can be specified using the mechanisms described in WS-
823 SecureConversation.

824 There is a core tension between security and reliable messaging that can be
825 problematic if not considered in implementations. That is, one aspect of security is
826 to prevent message replay and the core tenet of reliable messaging is to replay
827 messages until they are acknowledged. Consequently, if the security sub-system
828 processes a message but a failure occurs before the reliable messaging sub-system
829 records the message (or the message is considered "processed"), then it is possible
830 (and likely) that the security sub-system will treat subsequent copies as replays and
831 discard them. At the same time, the reliable messaging sub-system will likely
832 continue to expect and even solicit the missing message(s). Care should be taken to
833 avoid and prevent this rare condition.

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

- 836 • **Message alteration** – Alteration is prevented by including signatures of the message
837 information using WS-Security.
- 838 • **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-
839 Security.
- 840 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by
841 comparing secured policies – see WS-Policy and WS-SecurityPolicy).
- 842 • **Authentication** – Authentication is established using the mechanisms described in WS-
843 Security and WS-Trust. Each message is authenticated using the mechanisms described in
844 WS-Security.
- 845 • **Accountability** – Accountability is a function of the type of and string of the key and
846 algorithms being used. In many cases, a strong symmetric key provides sufficient
847 accountability. However, in some environments, strong PKI signatures are required.
- 848 • **Availability** – All reliable messaging services are subject to a variety of availability attacks.
849 Replay detection is a common attack and it is recommended that this be addressed by the
850 mechanisms described in WS-Security. (Note that because of legitimate message replays,
851 detection should include a differentiator besides message id such as a timestamp). Other
852 attacks, such as network-level denial of service attacks are harder to avoid and are outside
853 the scope of this specification. That said, care should be taken to ensure that minimal state is
854 saved prior to any authenticating sequences.

855 **6 References**

856 **6.1 Normative**

857 **[KEYWORDS]**

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

860 **[SOAP]**

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

862 **[URI]**

863 T. Berners-Lee, R. Fielding, L. Masinter, "[Uniform Resource Identifiers \(URI\): Generic Syntax](#),"
864 RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.

865 **[XML-ns]**

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

867 **[XML-Schema1]**

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

869 **[XML-Schema2]**

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

871 **[WSSecurity]**

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

875 **[SecureConversation]**

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

878 **[Tanenbaum]**

879 "Computer Networks," Andrew S. Tanenbaum, Prentice Hall PTR, 2003.

880 **[WSDL]**

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

882 **[WS-Addressing]**

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

884 **6.2 Non-Normative**

885 **[WS-Policy]**

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

887 **[WS-PolicyAttachment]**

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

889 **[SecurityPolicy]**

890 G. Della-Libra, "[Web Services Security Policy Language \(WS-SecurityPolicy\)](#)," December 2002.

891

892

893 Appendix A.Schema

894 The normative schema for WS-ReliableMessaging is located at:

895 <http://schemas.xmlsoap.org/ws/2005/02/rm/wsrn.xsd>

896 The following copy is provided for reference.

```
897 <xs:schema targetNamespace="http://schemas.xmlsoap.org/ws/2005/02/rm"
898 xmlns:xs="http://www.w3.org/2001/XMLSchema"
899 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
900 xmlns:wsrn="http://schemas.xmlsoap.org/ws/2005/02/rm"
901 elementFormDefault="qualified" attributeFormDefault="unqualified">
902   <xs:import
903 namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
904 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
905   <!-- Protocol Elements -->
906   <xs:complexType name="SequenceType">
907     <xs:sequence>
908       <xs:element ref="wsrn:Identifier"/>
909       <xs:element name="MessageNumber" type="xs:unsignedLong"/>
910       <xs:element name="LastMessage" minOccurs="0">
911         <xs:complexType>
912           <xs:sequence/>
913         </xs:complexType>
914       </xs:element>
915       <xs:any namespace="##other" processContents="lax" minOccurs="0"
916 maxOccurs="unbounded"/>
917     </xs:sequence>
918     <xs:anyAttribute namespace="##other" processContents="lax"/>
919   </xs:complexType>
920   <xs:element name="Sequence" type="wsrn:SequenceType"/>
921   <xs:element name="SequenceAcknowledgement">
922     <xs:complexType>
923       <xs:sequence>
924         <xs:element ref="wsrn:Identifier"/>
925         <xs:choice>
926           <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
927             <xs:complexType>
928               <xs:sequence/>
```

```

929         <xs:attribute name="Upper" type="xs:unsignedLong"
930 use="required"/>
931         <xs:attribute name="Lower" type="xs:unsignedLong"
932 use="required"/>
933         <xs:anyAttribute namespace="##other"
934 processContents="lax"/>
935     </xs:complexType>
936 </xs:element>
937     <xs:element name="Nack" type="xs:unsignedLong"
938 minOccurs="unbounded"/>
939 </xs:choice>
940     <xs:any namespace="##other" processContents="lax" minOccurs="0"
941 minOccurs="unbounded"/>
942 </xs:sequence>
943     <xs:anyAttribute namespace="##other" processContents="lax"/>
944 </xs:complexType>
945 </xs:element>
946 <xs:complexType name="AckRequestedType">
947     <xs:sequence>
948         <xs:element ref="wsrm:Identifier"/>
949         <xs:element name="MaxMessageNumberUsed" type="xs:unsignedLong"
950 minOccurs="0"/>
951         <xs:any namespace="##other" processContents="lax" minOccurs="0"
952 minOccurs="unbounded"/>
953     </xs:sequence>
954     <xs:anyAttribute namespace="##other" processContents="lax"/>
955 </xs:complexType>
956 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
957 <xs:element name="Identifier">
958     <xs:complexType>
959         <xs:annotation>
960             <xs:documentation>
961 This type is for elements whose [children] is an anyURI and can have
962 arbitrary attributes.
963             </xs:documentation>
964         </xs:annotation>
965         <xs:simpleContent>
966             <xs:extension base="xs:anyURI">
967                 <xs:anyAttribute namespace="##other" processContents="lax"/>
968             </xs:extension>
969         </xs:simpleContent>

```

```

970     </xs:complexType>
971 </xs:element>
972 <!-- Fault Container and Codes -->
973 <xs:simpleType name="FaultCodes">
974     <xs:restriction base="xs:QName">
975         <xs:enumeration value="wsrm:UnknownSequence"/>
976         <xs:enumeration value="wsrm:SequenceTerminated"/>
977         <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
978         <xs:enumeration value="wsrm:MessageNumberRollover"/>
979         <xs:enumeration value="wsrm:CreateSequenceRefused"/>
980     -   <xs:enumeration value="wsrm:LastMessageNumberExceeded"/>
981     </xs:restriction>
982 </xs:simpleType>
983 <xs:complexType name="SequenceFaultType">
984     <xs:sequence>
985         <xs:element name="FaultCode" type="xs:QName"/>
986         <xs:any namespace="##any" processContents="lax" minOccurs="0"
987 maxOccurs="unbounded"/>
988     </xs:sequence>
989     <xs:anyAttribute namespace="##any" processContents="lax"/>
990 </xs:complexType>
991 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
992 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
993 <xs:element name="CreateSequenceResponse"
994 type="wsrm:CreateSequenceResponseType"/>
995 <xs:element name="TerminateSequence"
996 type="wsrm:TerminateSequenceType"/>
997 <xs:complexType name="CreateSequenceType">
998     <xs:sequence>
999         <xs:element ref="wsrm:AcksTo"/>
1000        <xs:element ref="wsrm:Expires" minOccurs="0"/>
1001        <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1002        <xs:any namespace="##other" processContents="lax" minOccurs="0"
1003 maxOccurs="unbounded">
1004            <xs:annotation>
1005                <xs:documentation>
1006 It is the authors intent that this extensibility be used to transfer a
1007 Security Token Reference as defined in WS-Security.
1008 </xs:documentation>
1009            </xs:annotation>
1010        </xs:any>

```

```

1011     </xs:sequence>
1012     <xs:anyAttribute namespace="##other" processContents="lax"/>
1013 </xs:complexType>
1014 <xs:complexType name="CreateSequenceResponseType">
1015     <xs:sequence>
1016         <xs:element ref="wsrm:Identifier"/>
1017         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1018         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1019         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1020 maxOccurs="unbounded"/>
1021     </xs:sequence>
1022     <xs:anyAttribute namespace="##other" processContents="lax"/>
1023 </xs:complexType>
1024 <xs:complexType name="TerminateSequenceType">
1025     <xs:sequence>
1026         <xs:element ref="wsrm:Identifier"/>
1027         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1028 maxOccurs="unbounded"/>
1029     </xs:sequence>
1030     <xs:anyAttribute namespace="##other" processContents="lax"/>
1031 </xs:complexType>
1032 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1033 <xs:complexType name="OfferType">
1034     <xs:sequence>
1035         <xs:element ref="wsrm:Identifier"/>
1036         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1037         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1038 maxOccurs="unbounded"/>
1039     </xs:sequence>
1040     <xs:anyAttribute namespace="##other" processContents="lax"/>
1041 </xs:complexType>
1042 <xs:complexType name="AcceptType">
1043     <xs:sequence>
1044         <xs:element ref="wsrm:AcksTo"/>
1045         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1046 maxOccurs="unbounded"/>
1047     </xs:sequence>
1048     <xs:anyAttribute namespace="##other" processContents="lax"/>
1049 </xs:complexType>
1050 <xs:element name="Expires">
1051     <xs:complexType>

```

```
1052     <xs:simpleContent>
1053         <xs:extension base="xs:duration">
1054             <xs:anyAttribute namespace="##other" processContents="lax"/>
1055         </xs:extension>
1056     </xs:simpleContent>
1057 </xs:complexType>
1058 </xs:element>
1059 </xs:schema>
```

1060 **Appendix B.Message Examples**

1061 B.1.Create Sequence

1062 Create Sequence

```
1063 <?xml version="1.0" encoding="UTF-8"?>
1064 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1065 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1066 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1067   <S:Header>
1068     <wsa:MessageID>
1069       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1070     </wsa:MessageID>
1071     <wsa:To>http://fabrikami23example.com/serviceB/123</wsa:To>
1072     <wsa:Action>http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequence
1073   </wsa:Action>
1074     <wsa:ReplyTo>
1075       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1076     </wsa:ReplyTo>
1077   </S:Header>
1078   <S:Body>
1079     <wsm:CreateSequence>
1080       <wsm:AcksTo>
1081         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1082       </wsm:AcksTo>
1083     </wsm:CreateSequence>
1084   </S:Body>
1085 </S:Envelope>
```

1086 Create Sequence Response

```
1087 <?xml version="1.0" encoding="UTF-8"?>
1088 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1089 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1090 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1091   <S:Header>
1092     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1093     <wsa:RelatesTo>
1094       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1095     </wsa:RelatesTo>
1096     <wsa:Action>
```

```
1097     http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequenceResponse
1098     </wsa:Action>
1099 </S:Header>
1100 <S:Body>
1101     <wsrm:CreateSequenceResponse>
1102         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1103     </wsrm:CreateSequenceResponse>
1104 </S:Body>
1105 </S:Envelope>
```

1106 B.2. Initial Transmission

1107 The following example WS-ReliableMessaging headers illustrate the message
1108 exchange in the above figure. The three messages have the following headers; the
1109 third message is identified as the last message in the sequence:

1110 Message 1

```
1111 <?xml version="1.0" encoding="UTF-8"?>
1112 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1113 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1114 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1115   <S:Header>
1116     <wsa:MessageID>
1117       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1118     </wsa:MessageID>
1119     <wsa:To>http://fabrikam123example.com/serviceB/123</wsa:To>
1120     <wsa:From>
1121       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1122     </wsa:From>
1123     <wsa:Action>http://fabrikam123example.com/serviceB/123/request</wsa
1124 :Action>
1125     <wsm:Sequence>
1126       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1127       <wsm:MessageNumber>1</wsm:MessageNumber>
1128     </wsm:Sequence>
1129   </S:Header>
1130   <S:Body>
1131     <!-- Some Application Data -->
1132   </S:Body>
1133 </S:Envelope>
```

1134 Message 2

```
1135 <?xml version="1.0" encoding="UTF-8"?>
1136 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1137 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1138 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1139   <S:Header>
1140     <wsa:MessageID>
1141       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1142     </wsa:MessageID>
```

```
1143 <wsa:To>http://fabrikam123example.com/serviceB/123</wsa:To>
1144 <wsa:From>
1145 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1146 </wsa:From>
1147 <wsa:Action>http://fabrikam123example.com/serviceB/123/request</wsa
1148 :Action>
1149 <wsrm:Sequence>
1150 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1151 <wsrm:MessageNumber>2</wsrm:MessageNumber>
1152 </wsrm:Sequence>
1153 </S:Header>
1154 <S:Body>
1155 <!-- Some Application Data -->
1156 </S:Body>
1157 </S:Envelope>
```

1158 Message 3

```
1159 <?xml version="1.0" encoding="UTF-8"?>
1160 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1161 xmlns:wsrm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1162 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1163 <S:Header>
1164 <wsa:MessageID>
1165 http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1166 </wsa:MessageID>
1167 <wsa:To>http://fabrikam123example.com/serviceB/123</wsa:To>
1168 <wsa:From>
1169 <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1170 </wsa:From>
1171 <wsa:Action>http://fabrikam123example.com/serviceB/123/request</wsa:A
1172 ction>
1173 <wsrm:Sequence>
1174 <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1175 <wsrm:MessageNumber>3</wsrm:MessageNumber>
1176 <wsrm:LastMessage/>
1177 </wsrm:Sequence>
1178 </S:Header>
1179 <S:Body>
1180 <!-- Some Application Data -->
1181 </S:Body>
1182 </S:Envelope>
```

1183 B.3.First Acknowledgement

1184 Message number 2 has not been received by the RM Destination due to some
1185 transmission error so it responds with an acknowledgement for messages 1 and 3:

```
1186 <?xml version="1.0" encoding="UTF-8"?>
1187 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1188 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1189 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1190   <S:Header>
1191     <wsa:MessageID>
1192       http://fabrikam123example.com/guid/0baaf88d-483b-4ecf-a6d8-
1193 a7c2eb546810
1194     </wsa:MessageID>
1195     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1196     <wsa:From>
1197       <wsa:Address>http://fabrikam123example.com/serviceB/123</wsa:Address>
1198     </wsa:From>
1199     <wsa:Action>
1200       http://schemas.xmlsoap.org/ws/2005/02/rm/SequenceAcknowledgement
1201     </wsa:Action>
1202     <wsm:SequenceAcknowledgement>
1203       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1204       <wsm:AcknowledgementRange Upper="1" Lower="1"/>
1205       <wsm:AcknowledgementRange Upper="3" Lower="3"/>
1206     </wsm:SequenceAcknowledgement>
1207   </S:Header>
1208   <S:Body/>
1209 </S:Envelope>
```

1210 B.4.Retransmission

1211 The sending endpoint discovers that message number 2 was not received so it
1212 resends the message and requests an acknowledgement:

```
1213 <?xml version="1.0" encoding="UTF-8"?>
1214 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1215 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1216 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1217   <S:Header>
1218     <wsa:MessageID>
1219       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1220     </wsa:MessageID>
1221     <wsa:To>http://fabrikam123example.com/serviceB/123</wsa:To>
1222     <wsa:From>
1223       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1224     </wsa:From>
1225     <wsa:Action>http://fabrikam123example.com/serviceB/123/request</wsa:A
1226 ction>
1227     <wsm:Sequence>
1228       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1229       <wsm:MessageNumber>2</wsm:MessageNumber>
1230     </wsm:Sequence>
1231     <wsm:AckRequested>
1232       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1233     </wsm:AckRequested>
1234   </S:Header>
1235   <S:Body>
1236     <!-- Some Application Data -->
1237   </S:Body>
1238 </S:Envelope>
```

1239 B.5.Termination

1240 The RM Destination now responds with an acknowledgement for the complete
1241 sequence which can then be terminated:

```
1242 <?xml version="1.0" encoding="UTF-8"?>
1243 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1244 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1245 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1246   <S:Header>
1247     <wsa:MessageID>
1248       http://fabrikam123example.com/guid/0baaf88d-483b-4ecf-a6d8-
1249 a7c2eb546811
1250     </wsa:MessageID>
1251     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1252     <wsa:From>
1253       <wsa:Address>http://fabrikam123example.com/serviceB/123</wsa:Address>
1254     </wsa:From>
1255     <wsa:Action>
1256       http://schemas.xmlsoap.org/ws/2005/02/rm/SequenceAcknowledgement
1257     </wsa:Action>
1258     <wsm:SequenceAcknowledgement>
1259       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1260       <wsm:AcknowledgementRange Upper="3" Lower="1"/>
1261     </wsm:SequenceAcknowledgement>
1262   </S:Header>
1263   <S:Body/>
1264 </S:Envelope>
```

1265 Terminate Sequence

```
1266 <?xml version="1.0" encoding="UTF-8"?>
1267 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1268 xmlns:wsm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1269 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1270   <S:Header>
1271     <wsa:MessageID>
1272       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1273     </wsa:MessageID>
1274     <wsa:To>http://fabrikam123example.com/serviceB/123</wsa:To>
1275     <wsa:Action>
```

```
1276     http://schemas.xmlsoap.org/ws/2005/02/rm/TerminateSequence
1277     </wsa:Action>
1278     <wsa:From>
1279     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1280     </wsa:From>
1281 </S:Header>
1282 <S:Body>
1283     <wsrm:TerminateSequence>
1284     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1285     </wsrm:TerminateSequence>
1286 </S:Body>
1287 </S:Envelope>
```


1288 Appendix C.WSDL

1289 The non-normative WSDL [1.1](#) definition for WS-ReliableMessaging is located at:

1290 `http://schemas.xmlsoap.org/ws/2005/02/rm/wsd1/wsrn.wsd1`

1291 The following non-normative copy is provided for reference.

```
1292 <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1293 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1294 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1295 xmlns:rm="http://schemas.xmlsoap.org/ws/2005/02/rm"
1296 xmlns:tns="http://schemas.xmlsoap.org/ws/2005/02/rm/wsd1"
1297 targetNamespace="http://schemas.xmlsoap.org/ws/2005/02/rm/wsd1">
1298 <wsdl:types>
1299   <xs:schema>
1300     <xs:import namespace="http://schemas.xmlsoap.org/ws/2005/02/rm"
1301     schemaLocation="http://schemas.xmlsoap.org/ws/2005/02/rm/wsrn.xsd"/>
1302     <xs:import
1303     namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1304     schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1305   </xs:schema>
1306 </wsdl:types>
1307 <wsdl:message name="CreateSequence">
1308   <wsdl:part name="create" element="rm:CreateSequence"/>
1309 </wsdl:message>
1310 <wsdl:message name="CreateSequenceResponse">
1311   <wsdl:part name="createResponse"
1312   element="rm:CreateSequenceResponse"/>
1313 </wsdl:message>
1314 <wsdl:message name="TerminateSequence">
1315   <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1316 </wsdl:message>
1317 <wsdl:portType name="SequenceAbstractPortType">
1318   <wsdl:operation name="CreateSequence">
1319     <wsdl:input message="tns:CreateSequence"
1320     wsa:Action="http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequence"/>
1321     <wsdl:output message="tns:CreateSequenceResponse"
1322     wsa:Action="http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequenceResponse"/>
1323   </wsdl:operation>
```

```
1325     <wsdl:operation name="TerminateSequence">
1326         <wsdl:input message="tns:TerminateSequence"
1327 wsa:Action="http://schemas.xmlsoap.org/ws/2005/02/rm/CreateSequenceResp
1328 onse"/>
1329     </wsdl:operation>
1330 </wsdl:portType>
1331 </wsdl:definitions>
```

1332 **Appendix D.Acknowledgments**

1333 This document is based on initial contribution to OASIS WS-RX Technical Committee by the
1334 following authors: Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft,
1335 Doug Davis, IBM, Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM,
1336 Mary Ann Hondo, IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David
1337 Langworthy, Microsoft (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft,
1338 Steve Lucco, Microsoft, Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham,
1339 BEA, David Orchard, BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John
1340 Shewchuk, Microsoft, Tony Storey, IBM

1341 The following individuals have provided invaluable input into the initial contribution:

1342 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,
1343 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,
1344 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,
1345 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin
1346 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,
1347 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger
1348 Wolter, Microsoft

1349 The following individuals were members of the committee during the development of this
1350 specification:

1351 TBD

1352 **Appendix E.Revision History**

Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.

1353 **Appendix F. Notices**

1354 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1355 that might be claimed to pertain to the implementation or use of the technology described in this
1356 document or the extent to which any license under such rights might or might not be available;
1357 neither does it represent that it has made any effort to identify any such rights. Information on
1358 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1359 website. Copies of claims of rights made available for publication and any assurances of licenses
1360 to be made available, or the result of an attempt made to obtain a general license or permission
1361 for the use of such proprietary rights by implementors or users of this specification, can be
1362 obtained from the OASIS Executive Director.

1363 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1364 applications, or other proprietary rights which may cover technology that may be required to
1365 implement this specification. Please address the information to the OASIS Executive Director.

1366 Copyright (C) OASIS Open (2005). All Rights Reserved.

1367 This document and translations of it may be copied and furnished to others, and derivative works
1368 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
1369 published and distributed, in whole or in part, without restriction of any kind, provided that the
1370 above copyright notice and this paragraph are included on all such copies and derivative works.
1371 However, this document itself may not be modified in any way, such as by removing the copyright
1372 notice or references to OASIS, except as needed for the purpose of developing OASIS
1373 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
1374 Property Rights document must be followed, or as required to translate it into languages other
1375 than English.

1376 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
1377 successors or assigns.

1378 This document and the information contained herein is provided on an "AS IS" basis and OASIS
1379 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
1380 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
1381 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
1382 PARTICULAR PURPOSE.