



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

3 **Working Draft 05, October 20<sup>th</sup> 2005**

## **Document identifier:**

4           WS-ReliableMessaging-1.1-draft-03.doc

5 **Location:**

6           TBD

7 **Editors:**

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

9           Anish Karmarkar, Oracle <[Anish.Karmarkar@oracle.com](mailto:Anish.Karmarkar@oracle.com)>

10          TBD

11 **Abstract:**

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

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

22 **Composable Architecture:**

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

30 **Status:**

31 TBD

---

## 32 Table of Contents

33	<b>1INTRODUCTION.....</b>	<b>5</b>
34	1.1GOALS AND REQUIREMENTS.....	5
35	1.1.1Requirements.....	5
36	1.2NOTATIONAL CONVENTIONS.....	5
37	1.3NAMESPACE.....	6
38	1.4COMPLIANCE.....	7
39	<b>2RELIABLE MESSAGING MODEL.....</b>	<b>8</b>
40	2.1GLOSSARY.....	9
41	2.2PROTOCOL PRECONDITIONS.....	10
42	2.3PROTOCOL INVARIANTS.....	10
43	2.4EXAMPLE MESSAGE EXCHANGE.....	11
44	<b>3RM PROTOCOL ELEMENTS.....</b>	<b>13</b>
45	3.1SEQUENCES.....	13
46	3.2SEQUENCE ACKNOWLEDGEMENT.....	15
47	3.3REQUEST ACKNOWLEDGEMENT.....	17
48	3.4SEQUENCE CREATION.....	19
49	3.5SEQUENCE TERMINATION.....	22
50	3.6CLOSING A SEQUENCE.....	23
51	<b>4FAULTS.....</b>	<b>26</b>
52	4.1SEQUENCEFAULT ELEMENT.....	28
53	4.2SEQUENCE TERMINATED.....	29
54	4.3UNKNOWN SEQUENCE.....	29
55	4.4INVALID ACKNOWLEDGEMENT.....	29
56	4.5MESSAGE NUMBER ROLLOVER.....	30
57	4.6LAST MESSAGE NUMBER EXCEEDED.....	30
58	4.7CREATE SEQUENCE REFUSED.....	31
59	4.8SEQUENCE CLOSED.....	31
60	<b>5SECURITY CONSIDERATIONS.....</b>	<b>32</b>
61	<b>6REFERENCES.....</b>	<b>34</b>
62	6.1NORMATIVE.....	34
63	6.2NON-NORMATIVE.....	34

64	APPENDIX A.SCHEMA .....	36
65	APPENDIX B.MESSAGE EXAMPLES.....	41
66	B.1.CREATE SEQUENCE.....	42
67	B.2. INITIAL TRANSMISSION.....	44
68	B.3.FIRST ACKNOWLEDGEMENT.....	46
69	B.4.RETRANSMISSION.....	47
70	B.5.TERMINATION.....	48
71	APPENDIX C.WSDL.....	50
72	APPENDIX D.ACKNOWLEDGMENTS.....	52
73	APPENDIX E.REVISION HISTORY.....	53
74	APPENDIX F.NOTICES.....	54

# 1 Introduction

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

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

## 1.1 Goals and Requirements

### 1.1.1 Requirements

### 1.2 Notational Conventions

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

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

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

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

### 1.3 Namespace

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

<http://docs.oasis-open.org/wsrn/200510/>

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

The following namespaces are used in this document:

Table 1

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

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

<http://docs.oasis-open.org/wsrn/200510/wsrn.xsd>

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

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

126 <http://docs.oasis-open.org/wsrn/200510/SequenceAcknowledgement>

## 127 **1.4 Compliance**

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

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

## 136 2 Reliable Messaging Model

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

139

140 The WS-ReliableMessaging specification defines an interoperable protocol that  
141 requires a Reliable Messaging (RM) Source and Reliable Messaging (RM) Destination  
142 to ensure that each message transmitted by the RM Source is successfully received  
143 by an RM Destination, or barring successful receipt, that an RM Source can, except in  
144 the most extreme circumstances, accurately determine the disposition of each  
145 message transmitted as perceived by the RM Destination, so as to resolve any in-  
146 doubt status.

147 In addition, The protocol allows the RM Source and RM Destination to provide their  
148 respective Application Source and Application Destination a guarantee that a  
149 message that is sent by an Application Source will be delivered to the Application  
150 Destination.

151 This guarantee is specified as a delivery assurance. It is the responsibility of the RM  
152 Source and RM Destination to fulfill the delivery assurances on behalf of their  
153 respective Application counterparts, or raise an error. The protocol defined here  
154 allows endpoints to meet this guarantee for the delivery assurances defined below.  
155 However, the means by which these delivery assurances are manifested by either the  
156 RM Source or RM Destination roles is an implementation concern, and is out of scope  
157 of this specification.

158 Note that the underlying protocol defined in this specification remains the same  
159 regardless of the delivery assurance.

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

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

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

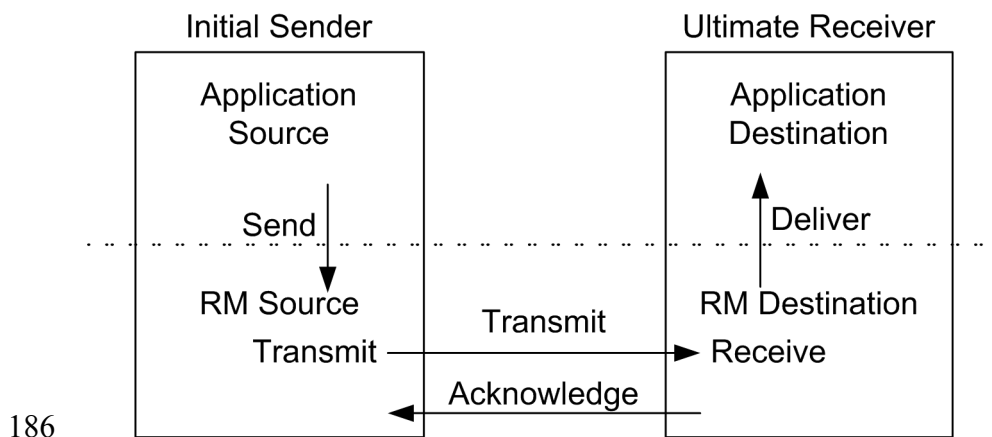
167 **AtLeastOnce** Every message sent will be delivered or an error will be raised on at  
168 least one endpoint. Some messages may be delivered more than once.

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



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

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



187 Figure 1: Reliable Messaging Model

## 188 2.1 Glossary

189 The following definitions are used throughout this specification:

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

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

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

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

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

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

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

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

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

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

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

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

## 208 **2.2 Protocol Preconditions**

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

- 211 • The RM Source **MUST** have an endpoint reference that uniquely identifies the RM Destination  
212 endpoint; correlations across messages addressed to the unique endpoint **MUST** be  
213 meaningful.
- 214 • The RM Source **MUST** have knowledge of the destination's policies, if any, and the RM  
215 Source **MUST** be capable of formulating messages that adhere to this policy.

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

## 218 **2.3 Protocol Invariants**

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

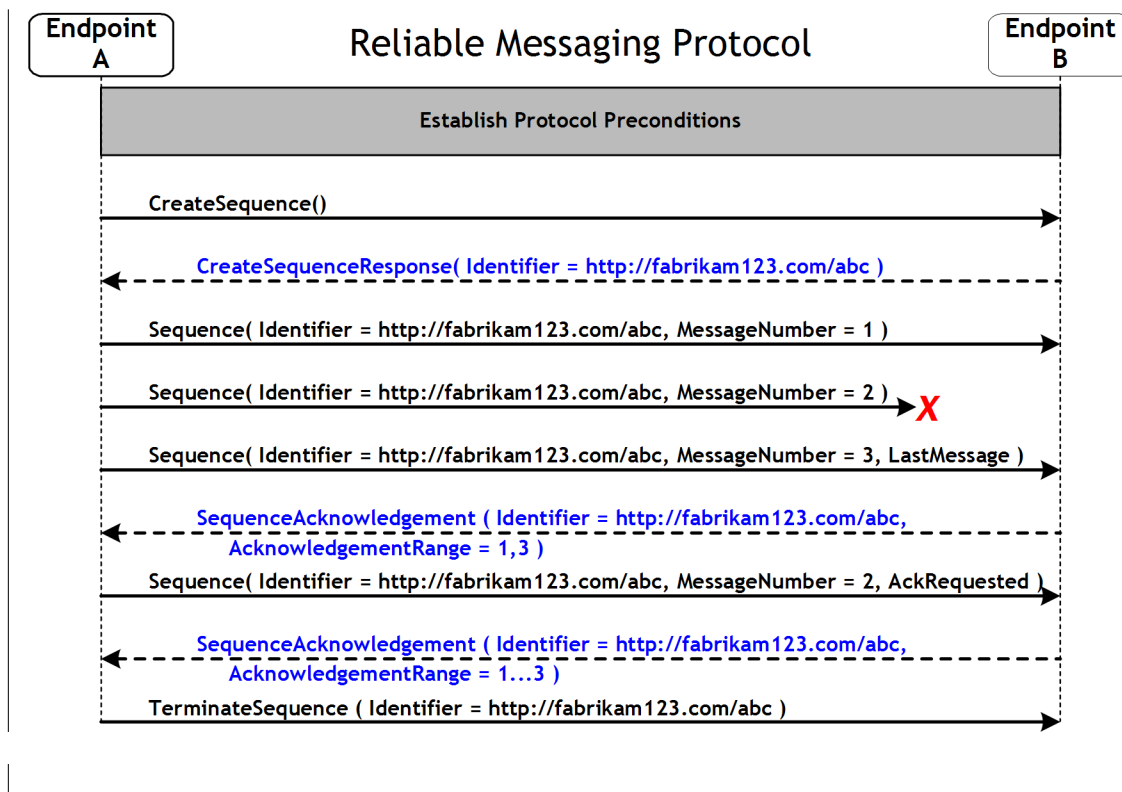
- 220 • The RM Source **MUST** assign each reliable message a sequence number (defined below)  
221 beginning at 1 and increasing by exactly 1 for each subsequent reliable message.

222 Every acknowledgement issued by the RM Destination **MUST** include within an  
223 acknowledgement range or ranges the sequence number of every message  
224 successfully received by the RM Destination and **MUST** exclude sequence numbers of  
225 any messages not yet received.



## 2.4 Example Message Exchange

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



229 Figure 2: The WS-ReliableMessaging Protocol

- 230 1. The protocol preconditions are established. These include policy exchange,  
 231 endpoint resolution, establishing trust.
- 232 2. The RM Source requests creation of a new Sequence.
- 233 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 234 4. The RM Source begins sending messages beginning with MessageNumber 1. In  
 235 the figure the RM Source sends 3 messages.
- 236 5. Since the 3rd message is the last in this exchange, the RM Source includes a  
 237 `<wsrm:AckRequestedLastMessage> tokenHeader`.
- 238 6. The 2nd message is lost in transit.
- 239 7. The RM Destination acknowledges receipt of message numbers 1 and 3 in  
 240 response to the RM Source's `<wsrm:LastMessageAckRequested> tokenHeader`.
- 241 8. The RM Source retransmits the 2nd message. This is a new message on the  
 242 underlying transport, but since it has the same sequence identifier and message

243 number so the RM Destination can recognize it as equivalent to the earlier  
244 message, in case both are received.

245 9. The RM Source includes an `<wsrm:AckRequested>` element so the RM Destination  
246 will expedite an acknowledgement.

247 10. The RM Destination receives the second transmission of the message with  
248 MessageNumber 2 and acknowledges receipt of message numbers 1, 2, and 3  
249 which carried the `<wsrm:LastMessageAckRequested>` ~~tokenHeader~~.

250 11. The RM Source receives this acknowledgement and sends a TerminateSequence  
251 message to the RM Destination indicating that the sequence is completed and  
252 reclaims any resources associated with the Sequence.

253 12. The RM Destination receives the TerminateSequence message indicating that the  
254 RM Source will not be sending any more messages, and reclaims any resources  
255 associated with the Sequence.

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

## 3 RM Protocol Elements

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

### 3.1 Sequences

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

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

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

A following exemplar defines its syntax:

```
<wsrm:Sequence ...>
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
  <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>
  <wsrm:LastMessage/>?
  ...
</wsrm:Sequence>
```

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

/wsrm:Sequence

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

/wsrm:Sequence/wsrm:Identifier

291 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely  
292 identifies the Sequence.

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

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

296 /wsrm:Sequence/wsrm:MessageNumber

297 This REQUIRED element MUST contain an xs:unsignedLong representing the ordinal position of  
298 the message within a Sequence. Sequence MessageNumbers start at 1 and monotonically  
299 increase throughout the Sequence. If the message number exceeds the internal limitations of an  
300 RM Source or RM Destination or reaches the maximum value of an xs:unsignedLong  
301 (18,446,744,073,709,551,615), the RM Source or Destination MUST issue a  
302 MessageNumberRollover fault.

303 ~~/wsrm:Sequence/wsrm:LastMessage~~

304 ~~This element MAY be included by the RM Source endpoint. The <wsrm:LastMessage> element~~  
305 ~~has no content.~~

306 /wsrm:Sequence/{any}

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

309 /wsrm:Sequence/@{any}

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

312 ~~A RM Source endpoint MUST include a <wsrm:LastMessage> element in the~~  
313 ~~<wsrm:Sequence> element for the last message in a Sequence. An RM Destination~~  
314 ~~endpoint MUST respond with a <wsrm:SequenceAcknowledgement> upon receipt of a~~  
315 ~~<wsrm:LastMessage> element. A Sequence MUST NOT use a <wsrm:MessageNumber>~~  
316 ~~value greater than that which accompanies a <wsrm:LastMessage> element. An RM~~  
317 ~~Destination MUST generate a LastMessageNumberExceeded (See Section 4.6) fault~~  
318 ~~upon receipt of such a message. In the event that an RM Source needs to close a~~  
319 ~~Sequence and there is no application message, the RM Source MAY send a message~~  
320 ~~with an empty body containing <wsrm:Sequence> header with the~~  
321 ~~<wsrm:LastMessage> element. In this usage, the action URI MUST be:~~

322 ~~<http://docs.oasis-open.org/wsrm/200510/LastMessage>~~

323 ~~in preference to the pattern defined in Section 1.2.~~

324 The following example illustrates a Sequence header block.

325 ~~<wsrm:Sequence>~~

LastMessage

11/2/2005

326  
327  
328  
329

```
<wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
<wsrm:MessageNumber>10</wsrm:MessageNumber>
<wsrm>LastMessage/>
</wsrm:Sequence>
```

## 330 3.2 Sequence Acknowledgement

331 The RM Destination informs the RM Source of successful message receipt using a  
332 <wsrm:SequenceAcknowledgement> header block. The  
333 <wsrm:SequenceAcknowledgement> header block MAY be transmitted independently  
334 or included on return messages. The RM Destination MAY send a  
335 <wsrm:SequenceAcknowledgement> header block at any point during which the  
336 sequence is valid. The timing of acknowledgements can be advertised using policy  
337 and acknowledgements can be explicitly requested using the <wsrm:AckRequested>  
338 directive (see Section 3.3). If a non-mustUnderstand fault occurs when processing  
339 an RM Header that was piggy-backed on another message, a fault MUST be  
340 generated, but the processing of the original message MUST NOT be affected.

341 The following exemplar defines its syntax:

342  
343  
344  
345  
346  
347  
348  
349  
350  
351

```
<wsrm:SequenceAcknowledgement ...>
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
  [ [ <wsrm:AcknowledgementRange ...
    Upper="xs:unsignedLong"
    Lower="xs:unsignedLong"/> +
    <wsrm:Final/> ? ]
    | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> +
    | <wsrm:None/> ]
  ...
</wsrm:SequenceAcknowledgement>
```

352 The following describes the content model of the <wsrm:SequenceAcknowledgement>  
353 header block.

354 /wsrm:SequenceAcknowledgement

355 This element contains the Sequence acknowledgement information.

356 /wsrm:SequenceAcknowledgement/wsrm:Identifier

357 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely  
358 identifies the Sequence.

359 /wsrm:SequenceAcknowledgement/wsrm:Identifier/@{any}

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



362 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange

363 This OPTIONAL element, if present, can occur 1 or more times. It contains a range of message  
 364 Sequence MessageNumbers successfully received by the receiving endpoint manager. The  
 365 ranges SHOULD NOT overlap. This element MUST NOT be present if either the <wsrm:Nack>  
 366 or <wsrm:None> elements are also present as a child of  
 367 <wsrm:SequenceAcknowledgement>.

368 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper

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

372 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower

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

376 /wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@{any}

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

379 /wsrm:SequenceAcknowledgement/wsrm:Final

380 This OPTIONAL element, if present, indicates that the RM Destination is not receiving new  
 381 messages for the specified Sequence. The RM Source can be assured that the ranges of  
 382 messages acknowledged by this SequenceAcknowledgement header block will not change in the  
 383 future. This element MUST be present when the Sequence is no longer receiving new message  
 384 for the specified sequence. Note: this element MUST NOT be used when sending a Nack, it can  
 385 only be used when sending AcknowledgementRanges.

386 /wsrm:SequenceAcknowledgement/wsrm:Nack

387 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the  
 388 <wsrm:MessageNumber> of an unreceived message in a Sequence. This element permits the  
 389 gap analysis of the <wsrm:AcknowledgementRange> elements to be performed at the RM  
 390 Destination rather than at the RM Source which may yield performance benefits in certain  
 391 environments. The <wsrm:Nack> element MUST NOT be present if either the  
 392 <wsrm:AcknowledgementRange> or <wsrm:None> elements are also present as a child of  
 393 <wsrm:SequenceAcknowledgement>. Upon the receipt of a Nack, an RM Source SHOULD  
 394 retransmit the message identified by the Nack. The RM Destination MUST NOT issue a  
 395 <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a message that it has  
 396 previously acknowledged within a <wsrm:AcknowledgementRange>. The RM Source SHOULD  
 397 ignore a <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a message  
 398 that has previously been acknowledged within a <wsrm:AcknowledgementRange>.

399 /wsrm:SequenceAcknowledgement/wsrm:None

400 This OPTIONAL element, if present, MUST be used when the RM Destination has not received  
401 any messages for the specified sequence. The <wsrm:None> element MUST NOT be present if  
402 either the <wsrm:AcknowledgementRange> or <wsrm:Nack> elements are also present as a  
403 child of the <wsrm:SequenceAcknowledgement>.

404 /wsrm:SequenceAcknowledgement/{any}

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

407 /wsrm:SequenceAcknowledgement/@{any}

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

410 The following examples illustrate <wsrm:SequenceAcknowledgement> elements:

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

```
412 <wsrm:SequenceAcknowledgement>  
413   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
414   <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
415 </wsrm:SequenceAcknowledgement>
```

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

```
418 <wsrm:SequenceAcknowledgement>  
419   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
420   <wsrm:AcknowledgementRange Upper="2" Lower="1"/>  
421   <wsrm:AcknowledgementRange Upper="6" Lower="4"/>  
422   <wsrm:AcknowledgementRange Upper="10" Lower="8"/>  
423 </wsrm:SequenceAcknowledgement>
```

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

```
425 <wsrm:SequenceAcknowledgement>  
426   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
427   <wsrm:Nack>3</wsrm:Nack>  
428 </wsrm:SequenceAcknowledgement>
```

### 429 3.3 Request Acknowledgement

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

433 At any time, the RM Source may request an acknowledgement message from the RM  
434 Destination endpoint using an `<wsrm:AckRequested>` header block.

435 The RM Source endpoint requests this acknowledgement by including an  
436 `<wsrm:AckRequested>` header block in the message. An RM Destination that receives  
437 a message that contains an `<wsrm:AckRequested>` header block MUST respond with  
438 a message containing a `<wsrm:SequenceAcknowledgement>` header block. If a non-  
439 `mustUnderstand` fault occurs when processing an RM Header that was piggy-backed  
440 on another message, a fault MUST be generated, but the processing of the original  
441 message MUST NOT be affected.

442 The following exemplar defines its syntax:

```
443 <wsrm:AckRequested ...>  
444   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
445   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber> ?  
446   ...  
447 </wsrm:AckRequested>
```

448 `/wsrm:AckRequested`

449 This element requests an acknowledgement for the identified sequence.

450 `/wsrm:AckRequested/wsrm:Identifier`

451 This REQUIRED element MUST contain an absolute URI, conformant with RFC2396, that  
452 uniquely identifies the Sequence to which the request applies.

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

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

456 `/wsrm:AckRequested/wsrm:MessageNumber`

457 This OPTIONAL element, if present, MUST contain an `xs:unsignedLong` representing the highest  
458 `<wsrm:MessageNumber>` sent by the RM Source within the Sequence. If present, it MAY be  
459 treated as a hint to the RM Destination as an optimization to the process of preparing to transmit a  
460 `<wsrm:SequenceAcknowledgement>`.

461 `/wsrm:AckRequested/{any}`

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

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

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

### 3.4 Sequence Creation

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

The RM Destination of the outbound sequence is the WS-Addressing EndpointReference [WS-Addressing] to which `<wsrm:CreateSequence>` is sent. The RM Destination of the inbound sequence is the WS-Addressing `<wsa:ReplyTo>` of the `<wsrm:CreateSequence>`.

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

```
<wsrm:CreateSequence ...>
  <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>
  <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
  <wsrm:Offer ...>
    <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
    <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
    ...
  </wsrm:Offer> ?
  ...
</wsrm:CreateSequence>
```

`/wsrm:CreateSequence`

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

`/wsrm:CreateSequence/wsrm:AcksTo`

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

`/wsrm:CreateSequence/wsrm:Expires`

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

504 /wsrm:CreateSequence/wsrm:Expires/@{any}  
 505 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added  
 506 to the element.

507 /wsrm:CreateSequence/wsrm:Offer  
 508 This element, if present, enables an RM Source to offer a corresponding Sequence for the reliable  
 509 exchange of messages transmitted from RM Destination to RM Source.

510 /wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier  
 511 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 that uniquely  
 512 identifies the offered Sequence.

513 /wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier/@{any}  
 514 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added  
 515 to the element.

516 /wsrm:CreateSequence/wsrm:Offer/wsrm:Expires  
 517 This element, if present, of type `xs:duration` specifies the duration for the Sequence. A value  
 518 of 'PT0S' indicates that the Sequence will never expire. Absence of the element indicates an  
 519 implied value of 'PT0S'.

520 /wsrm:CreateSequence/wsrm:Offer/wsrm:Expires/@{any}  
 521 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added  
 522 to the element.

523 /wsrm:CreateSequence/wsrm:Offer/{any}  
 524 This is an extensibility mechanism to allow different (extensible) types of information, based on a  
 525 schema, to be passed.

526 /wsrm:CreateSequence/wsrm:Offer/@{any}  
 527 This is an extensibility mechanism to allow different (extensible) types of information, based on a  
 528 schema, to be passed.

529 OPTIONAL/wsrm:CreateSequence/{any}  
 530 This is an extensibility mechanism to allow different (extensible) types of information, based on a  
 531 schema, to be passed.

532 /wsrm:CreateSequence/@{any}  
 533 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added  
 534 to the element.

535 A `<wsrm:CreateSequenceResponse>` is sent in the body of a response message by an  
536 RM Destination in response to receipt of a `<wsrm:CreateSequence>` request  
537 message. It carries the `<wsrm:Identifier>` of the created Sequence and indicates  
538 that the RM Source may begin sending messages in the context of the identified  
539 Sequence.

540 The following exemplar defines the `<wsrm:CreateSequenceResponse>` syntax:

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

550 `/wsrm:CreateSequenceResponse`

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

555 `/wsrm:CreateSequenceResponse/wsrm:Identifier`

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

558 `/wsrm:CreateSequenceResponse/wsrm:Identifier/@{any}`

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

561 `/wsrm:CreateSequenceResponse/wsrm:Expires`

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

567 `/wsrm:CreateSequenceResponse/wsrm:Expires/@{any}`

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

570 `/wsrm:CreateSequenceResponse/wsrm:Accept`

571 This element, if present, enables an RM Destination to accept the offer of a corresponding  
572 Sequence for the reliable exchange of messages transmitted from RM Destination to RM Source.  
573 This element MUST be present if the corresponding `<wsrm:CreateSequence>` message  
574 contained an `<wsrm:Offer>` element.

575 `/wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo`

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

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

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

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

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

586 `/wsrm:CreateSequenceResponse/{any}`

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

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

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

## 592 **3.5 Sequence Termination**

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

602 The following exemplar defines the TerminateSequence syntax:

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

605       ...  
606       </wsrm:TerminateSequence>  
607 /wsrm:TerminateSequence  
608 This element is sent by an RM Source to indicate it has completed its use of the Sequence, i.e. it  
609 MUST NOT send any additional message to the RM Destination referencing this sequence. It  
610 indicates that the RM Destination can safely reclaim any resources related to the identified  
611 Sequence. This element MUST NOT be sent as a header block.  
  
612 /wsrm:TerminateSequence/wsrm:Identifier  
613 This REQUIRED element MUST contain an absolute URI conformant with RFC2396 of the  
614 Sequence that is being terminated.  
  
615 /wsrm:TerminateSequence/wsrm:Identifier/@{any}  
616 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added  
617 to the element.  
  
618 /wsrm:TerminateSequence/{any}  
619 This is an extensibility mechanism to allow different (extensible) types of information, based on a  
620 schema, to be passed.  
  
621 /wsrm:TerminateSequence/@{any}  
622 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added  
623 to the element.

## 624 **3.6 Closing A Sequence**

625 There may be times during the use of an RM Sequence that the RM Source or RM  
626 Destination will wish to discontinue using a Sequence even if some of the messages  
627 have not been successfully delivered to the RM Destination.  
  
628 In the case where the RM Source wishes to discontinue use of a sequence, while it  
629 can send a TerminateSequence to the RM Destination, since this is a one-way  
630 message and due to the possibility of late arriving (or lost) messages and  
631 Acknowledgements, this would leave the RM Source unsure of the final ranges of  
632 messages that were successfully delivered to the RM Destination.  
  
633 To alleviate this, the RM Source can send a <wsrm:CloseSequence> element, in the  
634 body of a message, to the RM Destination to indicate that RM Destination MUST NOT  
635 receive any new messages for the specified sequence, other than those already  
636 received at the time the <wsrm:CloseSequence> element is interpreted by the RMD.  
637 Upon receipt of this message the RM Destination MUST send a



638 SequenceAcknowledgement to the RM Source. Note, this  
639 SequenceAcknowledgement MUST include the <wsrm:Final> element.

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

644 In the case where the RM Destination wishes to discontinue use of a sequence it may  
645 'close' the sequence itself. Please see wsrm:Final above and the SequenceClosed  
646 fault below. Note, the SequenceClosed Fault SHOULD be used in place of the  
647 SequenceTerminated Fault, whenever possible, to allow the RM Source to still receive  
648 Acknowledgements.

649 The following exemplar defines the CloseSequence syntax:

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

651 /wsrm:CloseSequence

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

655 /wsrm:CloseSequence@Identifier

656 This REQUIRED attribute contains an absolute URI conformant with RFC2396 that uniquely  
657 identifies the sequence.

658 /wsrm:CloseSequence/{any}

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

661 /wsrm:CloseSequence@{any}

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

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

667 The following exemplar defines the <wsrm:CloseSequenceResponse> syntax:

```
668 /wsrm:CloseSequenceResponse
```

669 /wsrm:CloseSequenceResponse

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

673 /wsrm:CloseSequenceResponse/{any}

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

676 /wsrm:CloseSequenceResponse@{any}

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

## 679 4 Faults

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

684 Sequence creation uses a CreateSequence, CreateSequenceResponse request-  
685 response pattern. Faults for this operation are treated as defined in WS-Addressing.  
686 CreateSequenceRefused is a possible fault reply for this operation.

687 UnknownSequence is a fault generated by endpoints when messages carrying RM  
688 header blocks targeted at unrecognized sequences are detected, these faults are also  
689 treated as defined in WS-Addressing. All other faults in this section relate to the  
690 processing of RM header blocks targeted at known sequences and are collectively  
691 referred to as sequence faults. Sequence faults SHOULD be sent to the same  
692 [destination] as <wsrm:SequenceAcknowledgement> messages. These faults are  
693 correlated using the Sequence identifier carried in the detail.

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

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

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

700 The definitions of faults use the following properties:

701 [Code] The fault code.

702 [Subcode] The fault subcode.

703 [Reason] The English language reason element.

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

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

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

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

LastMessage

11/2/2005

```

708 <S:Envelope>
709   <S:Header>
710     <wsa:Action>
711       http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
712     </wsa:Action>
713     <!-- Headers elided for clarity. -->
714   </S:Header>
715   <S:Body>
716     <S:Fault>
717       <S:Code>
718         <S:Value> [Code] </S:Value>
719         <S:Subcode>
720           <S:Value> [Subcode] </S:Value>
721         </S:Subcode>
722       </S:Code>
723       <S:Reason>
724         <S:Text xml:lang="en"> [Reason] </S:Text>
725       </S:Reason>
726       <S:Detail>
727         [Detail]
728         ...
729       </S:Detail>
730     </S:Fault>
731   </S:Body>
732 </S:Envelope>

```

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

```

735 <S11:Envelope>
736   <S11:Header>
737     <wsrm:SequenceFault>
738       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
739       ...
740     </wsrm:SequenceFault>
741     <!-- Headers elided for clarity. -->
742   </S11:Header>
743   <S11:Body>
744     <S11:Fault>
745       <faultcode> [Code] </faultcode>
746       <faultstring> [Reason] </faultstring>
747     </S11:Fault>
748   </S11:Body>

```

749 `</S11:Envelope>`

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

```
752 <S11:Envelope>
753   <S11:Body>
754     <S11:Fault>
755       <faultcode> [Subcode] </faultcode>
756       <faultstring xml:lang="en"> [Reason] </faultstring>
757     </S11:Fault>
758   </S11:Body>
759 </S11:Envelope>
```

## 760 4.1 SequenceFault Element

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

766 The following exemplar defines its syntax:

```
767 <wsrm:SequenceFault ...>
768   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
769   ...
770 </wsrm:SequenceFault>
```

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

772 `/wsrm:SequenceFault`

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

774 `/wsrm:SequenceFault/wsrm:FaultCode`

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

777 `/wsrm:SequenceFault/{any}`

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

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

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

## 783 **4.2 Sequence Terminated**

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

789 Properties:

790 [Code] Sender or Receiver

791 [Subcode] wsrn:SequenceTerminated

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

793 [Detail]

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

## 795 **4.3 Unknown Sequence**

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

798 Properties:

799 [Code] Sender

800 [Subcode] wsrn:UnknownSequence

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

802 [Detail]

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

## 804 **4.4 Invalid Acknowledgement**

805 This fault is sent by the RM Source in response to a  
806 `<wsrm:SequenceAcknowledgement>` that violates the cumulative acknowledgement  
807 invariant. An example of such a violation would be a `SequenceAcknowledgement`  
808 covering messages that have not been sent.

809 [Code] Sender

810 [Subcode] wsrn:InvalidAcknowledgement

811 [Reason] The SequenceAcknowledgement violates the cumulative acknowledgement  
812 invariant.

813 [Detail]

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

## 815 **4.5 Message Number Rollover**

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

818 Properties:

819 [Code] Sender

820 [Subcode] wsrn:MessageNumberRollover

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

822 [Detail]

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

## 824 **4.6 Last Message Number Exceeded**

825 ~~This fault is sent by an RM-Destination to indicate that it has received a message that~~  
826 ~~has a <wsrm:MessageNumber> within a Sequence that exceeds the value of the~~  
827 ~~<wsrm:MessageNumber> element that accompanied a <wsrm:LastMessage> element~~  
828 ~~for the Sequence.~~

829 ~~Properties:~~

830 ~~[Code] Sender~~

831 ~~[Subcode] wsrn:LastMessageNumberExceeded~~

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

834 ~~[Detail]~~

835 ~~`<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`~~

## 836 **4.7 Create Sequence Refused**

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

838 Properties:

839 [Code] Sender

840 [Subcode] wsrn:CreateSequenceRefused

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

842 [Detail] empty

## 843 **4.8 Sequence Closed**

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

847 Properties:

848 [Code] Sender

849 [Subcode] wsrn:SequenceClosed

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

851 [Detail] <wsrm:Identifier...> xs:anyURI </wsrm:Identifier>



## 5 Security Considerations

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

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

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

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

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

888 There is a core tension between security and reliable messaging that can be  
889 problematic if not considered in implementations. That is, one aspect of security is  
890 to prevent message replay and the core tenet of reliable messaging is to replay  
891 messages until they are acknowledged. Consequently, if the security sub-system  
892 processes a message but a failure occurs before the reliable messaging sub-system  
893 records the message (or the message is considered "processed"), then it is possible  
894 (and likely) that the security sub-system will treat subsequent copies as replays and  
895 discard them. At the same time, the reliable messaging sub-system will likely  
896 continue to expect and even solicit the missing message(s). Care should be taken to  
897 avoid and prevent this rare condition.

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

- 900 • **Message alteration** – Alteration is prevented by including signatures of the message  
901 information using WS-Security.
- 902 • **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-  
903 Security.
- 904 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by  
905 comparing secured policies – see WS-Policy and WS-SecurityPolicy).
- 906 • **Authentication** – Authentication is established using the mechanisms described in WS-  
907 Security and WS-Trust. Each message is authenticated using the mechanisms described in  
908 WS-Security.
- 909 • **Accountability** – Accountability is a function of the type of and string of the key and  
910 algorithms being used. In many cases, a strong symmetric key provides sufficient  
911 accountability. However, in some environments, strong PKI signatures are required.
- 912 • **Availability** – All reliable messaging services are subject to a variety of availability attacks.  
913 Replay detection is a common attack and it is recommended that this be addressed by the  
914 mechanisms described in WS-Security. (Note that because of legitimate message replays,  
915 detection should include a differentiator besides message id such as a timestamp). Other  
916 attacks, such as network-level denial of service attacks are harder to avoid and are outside  
917 the scope of this specification. That said, care should be taken to ensure that minimal state is  
918 saved prior to any authenticating sequences.

919 **6 References**

920 **6.1 Normative**

921 **[KEYWORDS]**

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

924 **[SOAP]**

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

926 **[URI]**

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

929 **[XML-ns]**

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

931 **[XML-Schema1]**

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

933 **[XML-Schema2]**

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

935 **[WSSecurity]**

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

939 **[Tanenbaum]**

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

941 **[WSDL]**

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

943 **[WS-Addressing]**

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

945 **6.2 Non-Normative**

946 **[WS-Policy]**

LastMessage

11/2/2005

Copyright © OASIS Open 2005. All Rights Reserved.

Page 35 of 55

947 D. Box, et al, "[Web Services Policy Framework \(WS-Policy\)](#)," September 2004.  
948 **[WS-PolicyAttachment]**  
949 D. Box, et al, "[Web Services Policy Attachment \(WS-PolicyAttachment\)](#)," September 2004.  
950 **[SecurityPolicy]**  
951 G. Della-Libra, "[Web Services Security Policy Language \(WS-SecurityPolicy\)](#)," December 2002.  
952 **[SecureConversation]**  
953 S. Anderson, et al, "[Web Services Secure Conversation Language \(WS-SecureConversation\)](#),"  
954 May 2004.  
955

## 956 Appendix A.Schema

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

958 <http://docs.oasis-open.org/wsrn/200510/wsrn.xsd>

959 The following copy is provided for reference.

```
960 <xs:schema targetNamespace="http://docs.oasis-open.org/wsrn/200510/"
961 xmlns:xs="http://www.w3.org/2001/XMLSchema"
962 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
963 xmlns:wsrm="http://docs.oasis-open.org/wsrn/200510/"
964 elementFormDefault="qualified" attributeFormDefault="unqualified">
965   <xs:import
966 namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
967 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
968   <!-- Protocol Elements -->
969   <xs:complexType name="SequenceType">
970     <xs:sequence>
971       <xs:element ref="wsrm:Identifier"/>
972       <xs:element name="MessageNumber" type="xs:unsignedLong"/>
973       <del><xs:element name="LastMessage" minOccurs="0"/></del>
974       <del><xs:complexType></del>
975       <del><xs:sequence/></del>
976       <del></xs:complexType></del>
977       <del></xs:element></del>
978       <xs:any namespace="##other" processContents="lax" minOccurs="0"
979 maxOccurs="unbounded"/>
980     </xs:sequence>
981     <xs:anyAttribute namespace="##other" processContents="lax"/>
982   </xs:complexType>
983   <xs:element name="Sequence" type="wsrm:SequenceType"/>
984   <xs:element name="SequenceAcknowledgement">
985     <xs:complexType>
986       <xs:sequence>
987         <xs:element ref="wsrm:Identifier"/>
988         <xs:choice>
989           <ws:sequence>
990             <xs:element name="AcknowledgementRange"
991 maxOccurs="unbounded">
992               <xs:complexType>
993                 <xs:sequence/>
```

LastMessage

11/2/2005

```

994         <xs:attribute name="Upper" type="xs:unsignedLong"
995 use="required"/>
996         <xs:attribute name="Lower" type="xs:unsignedLong"
997 use="required"/>
998         <xs:anyAttribute namespace="##other"
999 processContents="lax"/>
1000     </xs:complexType>
1001 </xs:element>
1002 <ws:element name="Final" minOccurs="0">
1003     <xs:complexType>
1004         <xs:sequence/>
1005     </xs:complexType>
1006 </ws:element>
1007 </ws:sequence>
1008 <xs:element name="Nack" type="xs:unsignedLong"
1009 minOccurs="unbounded"/>
1010     <xs:element name="None" minOccurs="0">
1011         <xs:complexType>
1012             <xs:sequence/>
1013         </xs:complexType>
1014     </xs:element>
1015 </xs:choice>
1016     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1017 minOccurs="unbounded"/>
1018 </xs:sequence>
1019 <xs:anyAttribute namespace="##other" processContents="lax"/>
1020 </xs:complexType>
1021 </xs:element>
1022 <xs:complexType name="AckRequestedType">
1023     <xs:sequence>
1024         <xs:element ref="wsrm:Identifier"/>
1025         <xs:element name="MessageNumber" type="xs:unsignedLong"
1026 minOccurs="0"/>
1027         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1028 minOccurs="unbounded"/>
1029     </xs:sequence>
1030 <xs:anyAttribute namespace="##other" processContents="lax"/>
1031 </xs:complexType>
1032 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1033 <xs:element name="Identifier">
1034     <xs:complexType>
1035         <xs:annotation>

```

```

1036         <xs:documentation>
1037 This type is for elements whose [children] is an anyURI and can have
1038 arbitrary attributes.
1039         </xs:documentation>
1040     </xs:annotation>
1041     <xs:simpleContent>
1042         <xs:extension base="xs:anyURI">
1043             <xs:anyAttribute namespace="##other" processContents="lax"/>
1044         </xs:extension>
1045     </xs:simpleContent>
1046 </xs:complexType>
1047 </xs:element>
1048 <!-- Fault Container and Codes -->
1049 <xs:simpleType name="FaultCodes">
1050     <xs:restriction base="xs:QName">
1051         <xs:enumeration value="wsrm:UnknownSequence"/>
1052         <xs:enumeration value="wsrm:SequenceTerminated"/>
1053         <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1054         <xs:enumeration value="wsrm:MessageNumberRollover"/>
1055         <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1056         <del><xs:enumeration value="wsrm:LastMessageNumberExceeded"/></del>
1057     </xs:restriction>
1058 </xs:simpleType>
1059 <xs:complexType name="SequenceFaultType">
1060     <xs:sequence>
1061         <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1062         <xs:any namespace="##any" processContents="lax" minOccurs="0"
1063 maxOccurs="unbounded"/>
1064     </xs:sequence>
1065     <xs:anyAttribute namespace="##any" processContents="lax"/>
1066 </xs:complexType>
1067 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1068 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1069 <xs:element name="CreateSequenceResponse"
1070 type="wsrm:CreateSequenceResponseType"/>
1071 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1072 <xs:element name="CloseSequenceResponse"
1073 type="wsrm:CloseSequenceResponseType"/>
1074 <xs:element name="TerminateSequence"
1075 type="wsrm:TerminateSequenceType"/>
1076 <xs:complexType name="CreateSequenceType">
1077     <xs:sequence>

```

```

1078     <xs:element ref="wsrm:AcksTo"/>
1079     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1080     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1081     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1082 maxOccurs="unbounded">
1083         <xs:annotation>
1084             <xs:documentation>
1085 It is the authors intent that this extensibility be used to transfer a
1086 Security Token Reference as defined in WS-Security.
1087 </xs:documentation>
1088             </xs:annotation>
1089         </xs:any>
1090     </xs:sequence>
1091     <xs:anyAttribute namespace="##other" processContents="lax"/>
1092 </xs:complexType>
1093 <xs:complexType name="CreateSequenceResponseType">
1094     <xs:sequence>
1095         <xs:element ref="wsrm:Identifier"/>
1096         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1097         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1098         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1099 maxOccurs="unbounded"/>
1100     </xs:sequence>
1101     <xs:anyAttribute namespace="##other" processContents="lax"/>
1102 </xs:complexType>
1103 <xs:complexType name="CloseSequenceType">
1104     <xs:sequence>
1105         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1106 maxOccurs="unbounded"/>
1107     </xs:sequence>
1108     <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1109     <xs:anyAttribute namespace="##other" processContents="lax"/>
1110 </xs:complexType>
1111 <xs:complexType name="CloseSequenceResponseType">
1112     <xs:sequence>
1113         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1114 maxOccurs="unbounded"/>
1115     </xs:sequence>
1116     <xs:anyAttribute namespace="##other" processContents="lax"/>
1117 </xs:complexType>
1118 <xs:complexType name="TerminateSequenceType">
1119     <xs:sequence>

```



```

1120     <xs:element ref="wsrm:Identifier"/>
1121     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1122 maxOccurs="unbounded"/>
1123   </xs:sequence>
1124   <xs:anyAttribute namespace="##other" processContents="lax"/>
1125 </xs:complexType>
1126 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1127 <xs:complexType name="OfferType">
1128   <xs:sequence>
1129     <xs:element ref="wsrm:Identifier"/>
1130     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1131     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1132 maxOccurs="unbounded"/>
1133   </xs:sequence>
1134   <xs:anyAttribute namespace="##other" processContents="lax"/>
1135 </xs:complexType>
1136 <xs:complexType name="AcceptType">
1137   <xs:sequence>
1138     <xs:element ref="wsrm:AcksTo"/>
1139     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1140 maxOccurs="unbounded"/>
1141   </xs:sequence>
1142   <xs:anyAttribute namespace="##other" processContents="lax"/>
1143 </xs:complexType>
1144 <xs:element name="Expires">
1145   <xs:complexType>
1146     <xs:simpleContent>
1147       <xs:extension base="xs:duration">
1148         <xs:anyAttribute namespace="##other" processContents="lax"/>
1149       </xs:extension>
1150     </xs:simpleContent>
1151   </xs:complexType>
1152 </xs:element>
1153 </xs:schema>

```

## 1154 **Appendix B.Message Examples**

## 1155 B.1.Create Sequence

### 1156 Create Sequence

```
1157 <?xml version="1.0" encoding="UTF-8"?>
1158 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1159 xmlns:wsrm="http://docs.oasis-open.org/wsrm/200510/"
1160 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1161   <S:Header>
1162     <wsa:MessageID>
1163       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
1164     </wsa:MessageID>
1165     <wsa:To>http://example.com/serviceB/123</wsa:To>
1166     <wsa:Action>http://docs.oasis-
1167 open.org/wsrm/200510/CreateSequence</wsa:Action>
1168     <wsa:ReplyTo>
1169       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1170     </wsa:ReplyTo>
1171   </S:Header>
1172   <S:Body>
1173     <wsrm:CreateSequence>
1174       <wsrm:AcksTo>
1175         <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1176       </wsrm:AcksTo>
1177     </wsrm:CreateSequence>
1178   </S:Body>
1179 </S:Envelope>
```

### 1180 Create Sequence Response

```
1181 <?xml version="1.0" encoding="UTF-8"?>
1182 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1183 xmlns:wsrm="http://docs.oasis-open.org/wsrm/200510/"
1184 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1185   <S:Header>
1186     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1187     <wsa:RelatesTo>
1188       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
1189     </wsa:RelatesTo>
1190     <wsa:Action>
1191       http://docs.oasis-open.org/wsrm/200510/CreateSequenceResponse
```

LastMessage

11/2/2005

```
1192     </wsa:Action>
1193 </S:Header>
1194 <S:Body>
1195     <wsrm:CreateSequenceResponse>
1196         <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1197     </wsrm:CreateSequenceResponse>
1198 </S:Body>
1199 </S:Envelope>
```

## 1200 B.2. Initial Transmission

1201 The following example WS-ReliableMessaging headers illustrate the message  
1202 exchange in the above figure. The three messages have the following headers; the  
1203 third message is identified as the last message in the sequence:

### 1204 Message 1

```
1205 <?xml version="1.0" encoding="UTF-8"?>
1206 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1207 xmlns:wsrm="http://docs.oasis-open.org/wsrm/200510/"
1208 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1209   <S:Header>
1210     <wsa:MessageID>
1211       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1212     </wsa:MessageID>
1213     <wsa:To>http://example.com/serviceB/123</wsa:To>
1214     <wsa:From>
1215       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1216     </wsa:From>
1217     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1218     <wsrm:Sequence>
1219       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1220       <wsrm:MessageNumber>1</wsrm:MessageNumber>
1221     </wsrm:Sequence>
1222   </S:Header>
1223   <S:Body>
1224     <!-- Some Application Data -->
1225   </S:Body>
1226 </S:Envelope>
```

### 1227 Message 2

```
1228 <?xml version="1.0" encoding="UTF-8"?>
1229 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1230 xmlns:wsrm="http://docs.oasis-open.org/wsrm/200510/"
1231 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1232   <S:Header>
1233     <wsa:MessageID>
1234       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1235     </wsa:MessageID>
1236     <wsa:To>http://example.com/serviceB/123</wsa:To>
```

LastMessage

11/2/2005

```

1237   <wsa:From>
1238       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1239   </wsa:From>
1240   <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1241   <wsrm:Sequence>
1242       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1243       <wsrm:MessageNumber>2</wsrm:MessageNumber>
1244   </wsrm:Sequence>
1245 </S:Header>
1246 <S:Body>
1247     <!-- Some Application Data -->
1248 </S:Body>
1249 </S:Envelope>

```

### 1250 Message 3

```

1251 <?xml version="1.0" encoding="UTF-8"?>
1252 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1253 xmlns:wsrm="http://docs.oasis-open.org/wsrm/200510/"
1254 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1255   <S:Header>
1256     <wsa:MessageID>
1257       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1258     </wsa:MessageID>
1259     <wsa:To>http://example.com/serviceB/123</wsa:To>
1260     <wsa:From>
1261       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1262     </wsa:From>
1263     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1264     <wsrm:Sequence>
1265       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1266       <wsrm:MessageNumber>3</wsrm:MessageNumber>
1267       <wsrm:LastMessage/>
1268     </wsrm:Sequence>
1269     <wsrm:AckRequested>
1270     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1271     </wsrm:AckRequested>
1272   </S:Header>
1273   <S:Body>
1274     <!-- Some Application Data -->
1275   </S:Body>
1276 </S:Envelope>

```

## 1277 B.3.First Acknowledgement

1278 Message number 2 has not been received by the RM Destination due to some  
1279 transmission error so it responds with an acknowledgement for messages 1 and 3:

```
1280 <?xml version="1.0" encoding="UTF-8"?>
1281 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1282 xmlns:wsrm="http://docs.oasis-open.org/wsrn/200510/"
1283 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1284   <S:Header>
1285     <wsa:MessageID>
1286       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1287     </wsa:MessageID>
1288     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1289     <wsa:From>
1290       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1291     </wsa:From>
1292     <wsa:Action>
1293       http://docs.oasis-open.org/wsrn/200510/SequenceAcknowledgement
1294     </wsa:Action>
1295     <wsrm:SequenceAcknowledgement>
1296       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1297       <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1298       <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1299     </wsrm:SequenceAcknowledgement>
1300   </S:Header>
1301   <S:Body/>
1302 </S:Envelope>
```

## B.4.Retransmission

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

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsmr="http://docs.oasis-open.org/wsmr/200510/"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
    </wsa:MessageID>
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <wsa:From>
      <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
    </wsa:From>
    <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
    <wsmr:Sequence>
      <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
      <wsmr:MessageNumber>2</wsmr:MessageNumber>
    </wsmr:Sequence>
    <wsmr:AckRequested>
      <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
    </wsmr:AckRequested>
  </S:Header>
  <S:Body>
    <!-- Some Application Data -->
  </S:Body>
</S:Envelope>
```



## 1331 B.5.Termination

1332 The RM Destination now responds with an acknowledgement for the complete  
1333 sequence which can then be terminated:

```
1334 <?xml version="1.0" encoding="UTF-8"?>
1335 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1336 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1337 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1338   <S:Header>
1339     <wsa:MessageID>
1340       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1341     </wsa:MessageID>
1342     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1343     <wsa:From>
1344       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1345     </wsa:From>
1346     <wsa:Action>
1347       http://docs.oasis-open.org/wsm/200510/SequenceAcknowledgement
1348     </wsa:Action>
1349     <wsm:SequenceAcknowledgement>
1350       <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
1351       <wsm:AcknowledgementRange Upper="3" Lower="1"/>
1352     </wsm:SequenceAcknowledgement>
1353   </S:Header>
1354   <S:Body/>
1355 </S:Envelope>
```

### 1356 Terminate Sequence

```
1357 <?xml version="1.0" encoding="UTF-8"?>
1358 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1359 xmlns:wsm="http://docs.oasis-open.org/wsm/200510/"
1360 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1361   <S:Header>
1362     <wsa:MessageID>
1363       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1364     </wsa:MessageID>
1365     <wsa:To>http://example.com/serviceB/123</wsa:To>
1366     <wsa:Action>
1367       http://docs.oasis-open.org/wsm/200510/TerminateSequence
1368     </wsa:Action>
```

LastMessage

11/2/2005

```
1369     <wsa:From>
1370     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1371     </wsa:From>
1372 </S:Header>
1373 <S:Body>
1374     <wsrm:TerminateSequence>
1375     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1376     </wsrm:TerminateSequence>
1377 </S:Body>
1378 </S:Envelope>
```

## 1379 Appendix C.WSDL

1380 The non-normative WSDL 1.1 definition for WS-ReliableMessaging is located at:

1381 <http://docs.oasis-open.org/wsrn/200510/wsd1/wsrn.wsd1>

1382 The following non-normative copy is provided for reference.

```
1383 <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1384 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1385 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1386 xmlns:rm="http://docs.oasis-open.org/wsrn/200510/"
1387 xmlns:tns="http://docs.oasis-open.org/wsrn/200510/wsd1"
1388 targetNamespace="http://docs.oasis-open.org/wsrn/200510/wsd1">
1389 <wsdl:types>
1390   <xs:schema>
1391     <xs:import namespace="http://docs.oasis-open.org/wsrn/200510/"
1392     schemaLocation="http://docs.oasis-open.org/wsrn/200510/wsrn.xsd"/>
1393     <xs:import
1394     namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1395     schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1396   </xs:schema>
1397 </wsdl:types>
1398 <wsdl:message name="CreateSequence">
1399   <wsdl:part name="create" element="rm:CreateSequence"/>
1400 </wsdl:message>
1401 <wsdl:message name="CreateSequenceResponse">
1402   <wsdl:part name="createResponse"
1403   element="rm:CreateSequenceResponse"/>
1404 </wsdl:message>
1405 <wsdl:message name="CloseSequence">
1406   <wsdl:part name="close" element="rm:CloseSequence"/>
1407 </wsdl:message>
1408 <wsdl:message name="CloseSequenceResponse">
1409   <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
1410 </wsdl:message>
1411 <wsdl:message name="TerminateSequence">
1412   <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1413 </wsdl:message>
1414 <wsdl:portType name="SequenceAbstractPortType">
1415   <wsdl:operation name="CreateSequence">
```

```
1416         <wsdl:input message="tns:CreateSequence"
1417 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CreateSequence"/>
1418         <wsdl:output message="tns:CreateSequenceResponse"
1419 wsa:Action="http://docs.oasis-
1420 open.org/wsrn/200510/CreateSequenceResponse"/>
1421     </wsdl:operation>
1422     <wsdl:operation name="CloseSequence">
1423         <wsdl:input name="tns:CloseSequence"
1424 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CloseSequence"/>
1425         <wsdl:output name="tns:CloseSequenceResponse"
1426 wsa:Action="http://docs.oasis-
1427 open.org/wsrn/200510/CloseSequenceResponse"/>
1428     </wsdl:operation>
1429     <wsdl:operation name="TerminateSequence">
1430         <wsdl:input message="tns:TerminateSequence"
1431 wsa:Action="http://docs.oasis-
1432 open.org/wsrn/200510/CreateSequenceResponse"/>
1433     </wsdl:operation>
1434 </wsdl:portType>
1435 </wsdl:definitions>
```

## 1436 **Appendix D.Acknowledgments**

1437 This document is based on initial contribution to OASIS WS-RX Technical Committee by the  
1438 following authors: Ruslan Bilorusets, BEA, Don Box, Microsoft, Luis Felipe Cabrera, Microsoft,  
1439 Doug Davis, IBM, Donald Ferguson, IBM, Christopher Ferris, IBM (Editor), Tom Freund, IBM,  
1440 Mary Ann Hondo, IBM, John Ibbotson, IBM, Lei Jin, BEA, Chris Kaler, Microsoft, David  
1441 Langworthy, Microsoft (Editor), Amelia Lewis, TIBCO Software, Rodney Limprecht, Microsoft,  
1442 Steve Lucco, Microsoft, Don Mullen, TIBCO Software, Anthony Nadalin, IBM, Mark Nottingham,  
1443 BEA, David Orchard, BEA, Jamie Roots, IBM, Shivajee Samdarshi, TIBCO Software, John  
1444 Shewchuk, Microsoft, Tony Storey, IBM

1445 The following individuals have provided invaluable input into the initial contribution:

1446 Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Allen Brown, Microsoft, Michael Conner, IBM,  
1447 George Copeland, Microsoft, Francisco Curbera, IBM, Paul Fremantle, IBM, Steve Graham, IBM,  
1448 Pat Helland, Microsoft, Rick Hill, Microsoft, Scott Hinkelman, IBM, Tim Holloway, IBM, Efim Hudis,  
1449 Microsoft, Gopal Kakivaya, Microsoft, Johannes Klein, Microsoft, Frank Leymann, IBM, Martin  
1450 Nally, IBM, Peter Niblett, IBM, Jeffrey Schlimmer, Microsoft, James Snell, IBM, Keith Stobie,  
1451 Microsoft, Satish Thatte, Microsoft, Stephen Todd, IBM, Sanjiva Weerawarana, IBM, Roger  
1452 Wolter, Microsoft

1453 The following individuals were members of the committee during the development of this  
1454 specification:

1455 TBD

## 1456 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.
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 'optional'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to <a href="http://docs.oasis-open.org/wsrn/200510/">http://docs.oasis-open.org/wsrn/200510/</a> )
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrn:FaultCode element)

## 1457 **Appendix F. Notices**

1458 OASIS takes no position regarding the validity or scope of any intellectual property or other rights  
1459 that might be claimed to pertain to the implementation or use of the technology described in this  
1460 document or the extent to which any license under such rights might or might not be available;  
1461 neither does it represent that it has made any effort to identify any such rights. Information on  
1462 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS  
1463 website. Copies of claims of rights made available for publication and any assurances of licenses  
1464 to be made available, or the result of an attempt made to obtain a general license or permission  
1465 for the use of such proprietary rights by implementors or users of this specification, can be  
1466 obtained from the OASIS Executive Director.

1467 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
1468 applications, or other proprietary rights which may cover technology that may be required to  
1469 implement this specification. Please address the information to the OASIS Executive Director.

1470 Copyright (C) OASIS Open (2005). All Rights Reserved.

1471 This document and translations of it may be copied and furnished to others, and derivative works  
1472 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,  
1473 published and distributed, in whole or in part, without restriction of any kind, provided that the  
1474 above copyright notice and this paragraph are included on all such copies and derivative works.  
1475 However, this document itself may not be modified in any way, such as by removing the copyright  
1476 notice or references to OASIS, except as needed for the purpose of developing OASIS  
1477 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
1478 Property Rights document must be followed, or as required to translate it into languages other  
1479 than English.

1480 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
1481 successors or assigns.

1482 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
1483 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO  
1484 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE  
1485 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A  
1486 PARTICULAR PURPOSE.