



1 Web Services ReliableMessaging 2 (WS-Reliable Messaging)

3 Working Draft 08, January 4, 2006

4 Document identifier:

5 wsrn-1.1-spec-wd-08

6 Location:

7 Editors:

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

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

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

11 Steve Winkler, SAP <steve.winkler@sap.com>

12 Ümit Yalçınalp, SAP <umit.yalcinalp@sap.com>

13 Contributors:

14 TBD

15 Abstract:

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

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

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

30 Status:

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

Table of Contents

33		
34	1 Introduction.....	4
35	1.1 Goals and Requirements.....	4
36	1.1.1 Requirements.....	4
37	1.2 Notational Conventions.....	4
38	1.3 Namespace.....	4
39	1.4 Compliance.....	5
40	2 Reliable Messaging Model.....	6
41	2.1 Glossary.....	6
42	2.2 Protocol Preconditions.....	7
43	2.3 Protocol Invariants.....	7
44	2.4 Example Message Exchange.....	7
45	3 RM Protocol Elements.....	10
46	3.1 Sequence Creation.....	10
47	3.2 Closing A Sequence.....	13
48	3.3 Sequence Termination.....	14
49	3.4 Sequences.....	15
50	3.5 Request Acknowledgement.....	16
51	3.6 Sequence Acknowledgement.....	17
52	4 Faults.....	20
53	4.1 SequenceFault Element.....	21
54	4.2 Sequence Terminated.....	22
55	4.3 Unknown Sequence.....	22
56	4.4 Invalid Acknowledgement.....	22
57	4.5 Message Number Rollover.....	23
58	4.6 Create Sequence Refused.....	23
59	4.7 Sequence Closed.....	23
60	5 Security Considerations.....	24
61	6 References.....	26
62	6.1 Normative.....	26
63	6.2 Non-Normative.....	26
64	A. Schema.....	28
65	B. Message Examples.....	32
66	B.1 Create Sequence.....	32
67	B.2 Initial Transmission.....	32
68	B.3 First Acknowledgement.....	34
69	B.4 Retransmission.....	34
70	B.5 Termination.....	35

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

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 delivery of messages. It defines a messaging protocol to identify, track, and manage the reliable delivery of messages between 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 specified in this document. Additional children elements and/or attributes MAY be added at the indicated extension points but they 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/ws-rx/wsrn/200510
```

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

110 The following namespaces are used in this document:

111 *Table 1*

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

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

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

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

115 If an action IRI is used, and one is not already defined per the rules of the WS-Addressing specification
116 [WS-Addressing], then the action IRI MUST consist of the WS-RM namespace URI concatenated with a
117 '/', followed by the message element name. For example:

118 <http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement>

119 1.4 Compliance

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

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

2 Reliable Messaging Model

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

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

The protocol supports reliability features which include ordered delivery, duplicate elimination, and guaranteed receipt for the RMD. It is expected that the AD and RMD will implement as many of these or as few of these characteristics as necessary to implement the AD. In any case the wire protocol does not change.

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

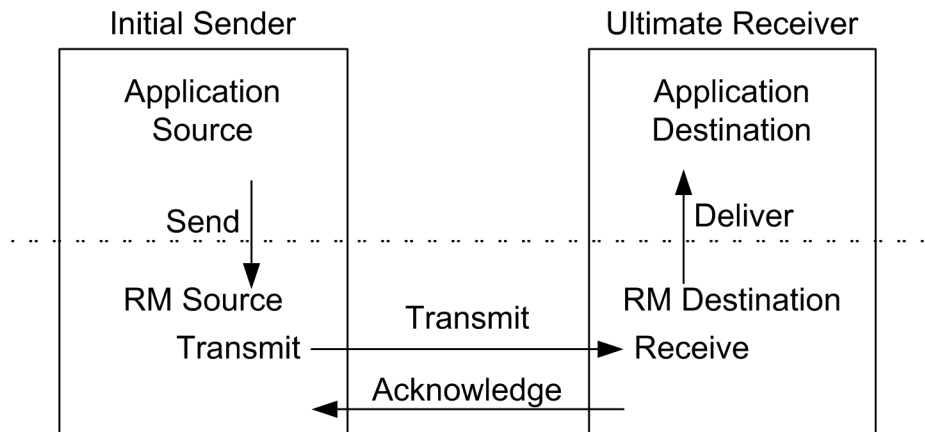


Figure 1: Reliable Messaging Model

2.1 Glossary

The following definitions are used throughout this specification:

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

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

Application Source: The endpoint that Sends a message.

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

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

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

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

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

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

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

166 2.2 Protocol Preconditions

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

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

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

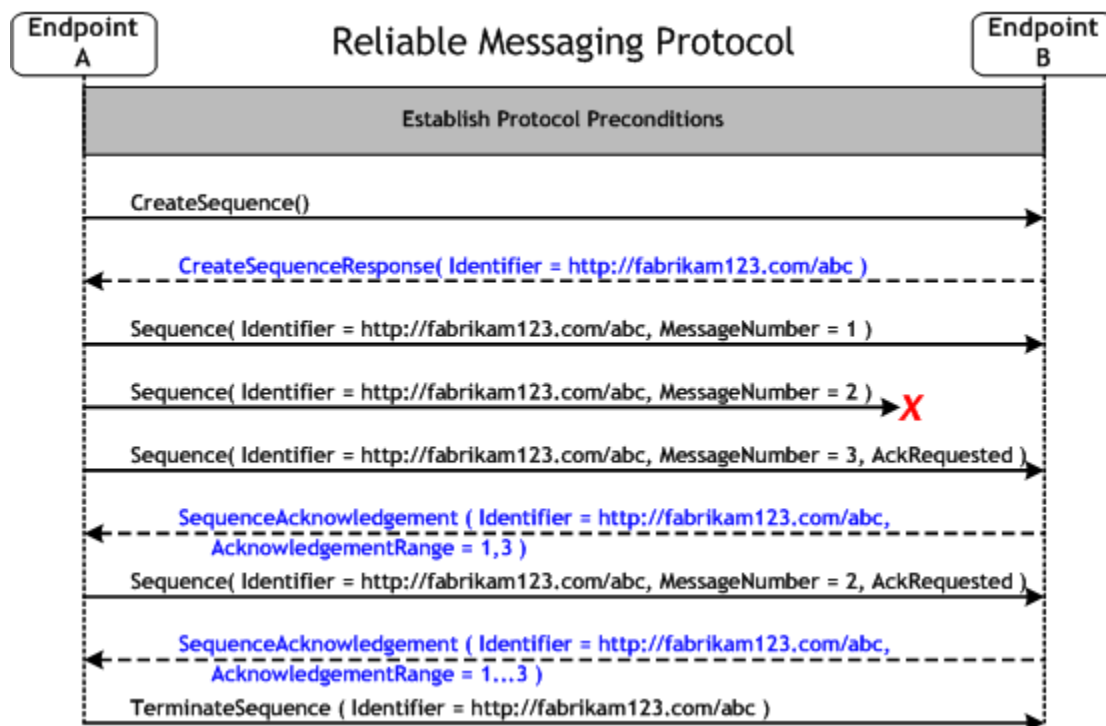
175 2.3 Protocol Invariants

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

- 177 • The RM Source **MUST** assign each message within a Sequence a message number (defined
178 below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers
179 **MUST** be assigned in the same order in which messages are sent by the Application Source.
- 180 • Every acknowledgement issued by the RM Destination **MUST** include within an acknowledgement
181 range or ranges the sequence number of every message successfully received by the RM
182 Destination and **MUST** exclude sequence numbers of any messages not yet received.

183 2.4 Example Message Exchange

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



185 Figure 2: The WS-ReliableMessaging Protocol

- 186 1. The protocol preconditions are established. These include policy exchange, endpoint resolution,
187 establishing trust.
- 188 2. The RM Source requests creation of a new Sequence.
- 189 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 190 4. The RM Source begins sending messages beginning with MessageNumber 1. In the figure above,
191 the RM Source sends 3 messages.
- 192 5. Since the 3rd message is the last in this exchange, the RM Source includes a
193 `<wsrm:AckRequested>` Header.
- 194 6. The 2nd message is lost in transit.
- 195 7. The RM Destination acknowledges receipt of message numbers 1 and 3 as a result of receiving the
196 RM Source's `<wsrm:AckRequested>` Header.
- 197 8. The RM Source retransmits the 2nd message. This is a new message on the underlying transport,
198 but it has the same sequence identifier and message number so the RM Destination can recognize
199 it as equivalent to the earlier message, in case both are received.
- 200 9. The RM Source includes an `<wsrm:AckRequested>` element so the RM Destination will expedite
201 an acknowledgement.
- 202 10. The RM Destination receives the second transmission of the message with MessageNumber 2 and
203 acknowledges receipt of message numbers 1, 2, and 3.
- 204 11. The RM Source receives this acknowledgement and sends a `TerminateSequence` message to the
205 RM Destination indicating that the sequence is completed and reclaims any resources associated
206 with the Sequence.
- 207 12. The RM Destination receives the `TerminateSequence` message indicating that the RM Source will
208 not be sending any more messages, and reclaims any resources associated with the Sequence.
- 209 The RM Source will expect to receive acknowledgements from the RM Destination during the course of a
210 message exchange at occasions described in Section 3 below. Should the acknowledgement not be

211 received in a timely fashion, the RM Source MUST re-transmit the request since either the request or the
212 associated acknowledgement may have been lost. Since the nature and dynamic characteristics of the
213 underlying transport and potential intermediaries are unknown in the general case, the timing of re-
214 transmissions cannot be specified. Additionally, over-aggressive re-transmissions have been
215 demonstrated to cause transport or intermediary flooding which are counterproductive to the intention of
216 providing a reliable exchange of messages. Consequently, implementers are encouraged to utilize
217 adaptive mechanisms that dynamically adjust re-transmission time and the back-off intervals that are
218 appropriate to the nature of the transports and intermediaries envisioned. For the case of TCP/IP
219 transports, a mechanism similar to that described as RTTM in RFC 1323 [RTTM] should be considered.

220 Now that the basic model has been outlined, the details of the elements used in this protocol are now
221 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 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>`. Note, offering a Sequence within the `<wsrm:CreateSequence>` element is simply a protocol optimization. There is no semantic difference between offering a Sequence, and choosing not to offer one and subsequently creating a new Sequence to carry messages from the RM Destination to the RM Source.

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.

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

`/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'.

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

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

267 /wsrm:CreateSequence/wsrm:Offer

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

270 /wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier

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

273 /wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier/@{any}

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

276 /wsrm:CreateSequence/wsrm:Offer/wsrm:Expires

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

280 /wsrm:CreateSequence/wsrm:Offer/wsrm:Expires/@{any}

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

283 /wsrm:CreateSequence/wsrm:Offer/{any}

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

286 /wsrm:CreateSequence/wsrm:Offer/@{any}

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

289 /wsrm:CreateSequence/{any}

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

292 /wsrm:CreateSequence/@{any}

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

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

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

```
300 <wsrm:CreateSequenceResponse ...>  
301   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
302   <wsrm:Expires> xs:duration </wsrm:Expires> ?  
303   <wsrm:Accept ...>  
304     <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>
```

```

305     ...
306     </wsrm:Accept> ?
307     ...
308 </wsrm:CreateSequenceResponse>

```

309 /wsrm:CreateSequenceResponse

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

313 /wsrm:CreateSequenceResponse/wsrm:Identifier

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

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

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

319 /wsrm:CreateSequenceResponse/wsrm:Expires

320 This element, if present, of type `xs:duration` accepts or refines the RM Source's requested duration for
 321 the Sequence. A value of 'PT0S' indicates that the Sequence will never expire. Absence of the element
 322 indicates an implied value of 'PT0S'. This value MUST be equal to or less than the value requested by the
 323 RM Source in the corresponding <wsrm:CreateSequence> message.

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

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

327 /wsrm:CreateSequenceResponse/wsrm:Accept

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

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

333 /wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo

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

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

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

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

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

343 /wsrm:CreateSequenceResponse/{any}

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

346 /wsrm:CreateSequenceResponse/@{any}

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

349 3.2 Closing A Sequence

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

355 If the RM Source wishes to close the Sequence then it sends a <wsrm:CloseSequence> element, in the
356 body of a message, to the RM Destination. This message indicates that RM Destination MUST NOT
357 receive any new messages for the specified sequence, other than those already received at the time the
358 <wsrm:CloseSequence> element is interpreted by the RMD. Upon receipt of this message, or
359 subsequent to the RM Destination closing the Sequence of its own volition, the RM Destination MUST
360 include a final SequenceAcknowledgement (that MUST include the <wsrm:Final> element) header block
361 on each message destined to the RM Source, including the CloseSequenceResponse message and on
362 any Sequence Fault transmitted to the RMS.

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

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

371 The following exemplar defines the CloseSequence syntax:

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

373 /wsrm:CloseSequence

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

377 /wsrm:CloseSequence@Identifier

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

380 /wsrm:CloseSequence/{any}

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

383 /wsrm:CloseSequence@{any}

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

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

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

```
390 /wsrm:CloseSequenceResponse
```

```
391 /wsrm:CloseSequenceResponse
```

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

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

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

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

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

401 3.3 Sequence Termination

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

409 The following exemplar defines the TerminateSequence syntax:

```
410 <wsrm:TerminateSequence ...>  
411   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
412   ...  
413 </wsrm:TerminateSequence>
```

```
414 /wsrm:TerminateSequence
```

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

```
419 /wsrm:TerminateSequence/wsrm:Identifier
```

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

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

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

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

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

428 /wsrm:TerminateSequence/@{any}

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

431 3.4 Sequences

432 The RM protocol uses a <wsrm:Sequence> header block to track and manage the reliable delivery of
433 messages. Messages for which a reliable delivery is required MUST contain a <wsrm:Sequence>
434 header block. Each Sequence MUST have a unique <wsrm:Identifier> element and each message
435 within a Sequence MUST have a <wsrm:MessageNumber> element that increments by 1 from an initial
436 value of 1. These values are contained within a <wsrm:Sequence> header block accompanying each
437 message being delivered in the context of a Sequence.

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

439 A following exemplar defines its syntax:

```
440 <wsrm:Sequence ...>  
441   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
442   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber>  
443   ...  
444 </wsrm:Sequence>
```

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

446 /wsrm:Sequence

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

451 /wsrm:Sequence/wsrm:Identifier

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

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

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

457 /wsrm:Sequence/wsrm:MessageNumber

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

463 /wsrm:Sequence/{any}

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

466 /wsrm:Sequence/@{any}

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

469 The following example illustrates a Sequence header block.

```
470 <wsrm:Sequence>  
471   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
472   <wsrm:MessageNumber>10</wsrm:MessageNumber>  
473 </wsrm:Sequence>
```

474 3.5 Request Acknowledgement

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

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

479 The RM Source may request an acknowledgement message from the RM Destination at any time by
480 including an <wsrm:AckRequested> header block in the message. An RM Destination that receives a
481 message that contains an <wsrm:AckRequested> header block MUST respond with a message
482 containing a <wsrm:SequenceAcknowledgement> header block. If a non-mustUnderstand fault occurs
483 when processing an RM Header that was piggy-backed on another message, a fault MUST be generated,
484 but the processing of the original message MUST NOT be affected.

485 The following exemplar defines its syntax:

```
486 <wsrm:AckRequested ...>  
487   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
  
488   ...  
489 </wsrm:AckRequested>
```

490 /wsrm:AckRequested

491 This element requests an acknowledgement for the identified sequence.

492 /wsrm:AckRequested/wsrm:Identifier

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

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

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

498 /wsrm:AckRequested/{any}

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

501 /wsrm:AckRequested/@{any}

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

3.6 Sequence Acknowledgement

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

The following exemplar defines its syntax:

```
<wsrm:SequenceAcknowledgement ...>
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
  [ [ <wsrm:AcknowledgementRange ...
      Upper="xs:unsignedLong"
      Lower="xs:unsignedLong"/> +

      | <wsrm:None/> ]
    <wsrm:Final/> ?
    | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> + ]
  ...
</wsrm:SequenceAcknowledgement>
```

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

`/wsrm:SequenceAcknowledgement`

This element contains the Sequence acknowledgement information.

`/wsrm:SequenceAcknowledgement/wsrm:Identifier`

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

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

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

`/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange`

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

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

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

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

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

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

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

550 /wsrm:SequenceAcknowledgement/wsrm:Final

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

557 /wsrm:SequenceAcknowledgement/wsrm:Nack

558 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the
559 <wsrm:MessageNumber> of an unreceived message in a Sequence. This element permits the gap
560 analysis of the <wsrm:AcknowledgementRange> elements to be performed at the RM Destination
561 rather than at the RM Source which may yield performance benefits in certain environments. The
562 <wsrm:Nack> element MUST NOT be present if a sibling <wsrm:AcknowledgementRange> or
563 <wsrm:None> elements are also present as a child of <wsrm:SequenceAcknowledgement>. Upon the
564 receipt of a Nack, an RM Source SHOULD retransmit the message identified by the Nack. The RM
565 Destination MUST NOT issue a <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for
566 a message that it has previously acknowledged within a <wsrm:AcknowledgementRange>. The RM
567 Source SHOULD ignore a <wsrm:SequenceAcknowledgement> containing a <wsrm:Nack> for a
568 message that has previously been acknowledged within a <wsrm:AcknowledgementRange>.

569 /wsrm:SequenceAcknowledgement/wsrm:None

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

574 /wsrm:SequenceAcknowledgement/{any}

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

577 /wsrm:SequenceAcknowledgement/@{any}

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

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

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

```
582 <wsrm:SequenceAcknowledgement>  
583   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
584   <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
585 </wsrm:SequenceAcknowledgement>
```

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

```
588 <wsrm:SequenceAcknowledgement>  
589   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
```

```
590      <wsrm:AcknowledgementRange Upper="2" Lower="1"/>
591      <wsrm:AcknowledgementRange Upper="6" Lower="4"/>
592      <wsrm:AcknowledgementRange Upper="10" Lower="8"/>
593 </wsrm:SequenceAcknowledgement>
```

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

```
595 <wsrm:SequenceAcknowledgement>
596   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
597   <wsrm:Nack>3</wsrm:Nack>
598 </wsrm:SequenceAcknowledgement>
```

4 Faults

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

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

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

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

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

The definitions of faults use the following properties:

[Code] The fault code.

[Subcode] The fault subcode.

[Reason] The English language reason element.

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

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

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

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

```
<S:Envelope>
  <S:Header>
    <wsa:Action>
      http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
    </wsa:Action>
    <!-- Headers elided for clarity. -->
  </S:Header>
  <S:Body>
    <S:Fault>
      <S:Code>
        <S:Value> [Code] </S:Value>
        <S:Subcode>
          <S:Value> [Subcode] </S:Value>
        </S:Subcode>
      </S:Code>
      <S:Reason>
```

```

641     <S:Text xml:lang="en"> [Reason] </S:Text>
642   </S:Reason>
643   <S:Detail>
644     [Detail]
645     ...
646   </S:Detail>
647 </S:Fault>
648 </S:Body>
649 </S:Envelope>

```

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

```

652 <S11:Envelope>
653   <S11:Header>
654     <wsrm:SequenceFault>
655       <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
656       ...
657     </wsrm:SequenceFault>
658     <!-- Headers elided for clarity. -->
659   </S11:Header>
660   <S11:Body>
661     <S11:Fault>
662       <faultcode> [Code] </faultcode>
663       <faultstring> [Reason] </faultstring>
664     </S11:Fault>
665   </S11:Body>
666 </S11:Envelope>

```

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

```

669 <S11:Envelope>
670   <S11:Body>
671     <S11:Fault>
672       <faultcode> [Subcode] </faultcode>
673       <faultstring xml:lang="en"> [Reason] </faultstring>
674     </S11:Fault>
675   </S11:Body>
676 </S11:Envelope>

```

677 4.1 SequenceFault Element

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

682 The following exemplar defines its syntax:

```

683 <wsrm:SequenceFault ...>
684   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
685   ...
686 </wsrm:SequenceFault>

```

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

688 /wsrm:SequenceFault

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

690 /wsrm:SequenceFault/wsrm:FaultCode

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

692 /wsrm:SequenceFault/{any}

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

695 /wsrm:SequenceFault/@{any}

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

698 4.2 Sequence Terminated

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

703 Properties:

704 [Code] Sender or Receiver

705 [Subcode] wsrm:SequenceTerminated

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

707 [Detail]

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

709 4.3 Unknown Sequence

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

712 Properties:

713 [Code] Sender

714 [Subcode] wsrm:UnknownSequence

715 [Reason] The value of wsrm:Identifier is not a known Sequence identifier.

716 [Detail]

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

718 4.4 Invalid Acknowledgement

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

722 [Code] Sender

723 [Subcode] wsrm:InvalidAcknowledgement

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

725 [Detail]

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

727 **4.5 Message Number Rollover**

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

729 Properties:

730 [Code] Sender

731 [Subcode] wsrm:MessageNumberRollover

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

733 [Detail]

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

735 **4.6 Create Sequence Refused**

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

737 Properties:

738 [Code] Sender

739 [Subcode] wsrm:CreateSequenceRefused

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

741 [Detail]

742 `xs:any`

743 **4.7 Sequence Closed**

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

747 Properties:

748 [Code] Sender

749 [Subcode] wsrm:SequenceClosed

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

751 [Detail]

752 `<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 destination, then the security context needs to be established or shared with the destination 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.

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

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

- **Message alteration** – Alteration is prevented by including signatures of the message information using WS-Security.
- **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-Security.

- 795 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by comparing
796 secured policies – see WS-Policy and WS-SecurityPolicy).
- 797 • **Authentication** – Authentication is established using the mechanisms described in WS-Security
798 and WS-Trust. Each message is authenticated using the mechanisms described in WS-Security.
- 799 • **Accountability** – Accountability is a function of the type of and string of the key and algorithms
800 being used. In many cases, a strong symmetric key provides sufficient accountability. However, in
801 some environments, strong PKI signatures are required.
- 802 • **Availability** – All reliable messaging services are subject to a variety of availability attacks. Replay
803 detection is a common attack and it is recommended that this be addressed by the mechanisms
804 described in WS-Security. (Note that because of legitimate message replays, detection should
805 include a differentiator besides message id such as a timestamp). Other attacks, such as network-
806 level denial of service attacks are harder to avoid and are outside the scope of this specification.
807 That said, care should be taken to ensure that minimal state is saved prior to any authenticating
808 sequences.

6 References

6.1 Normative

[KEYWORDS]

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

[SOAP 1.1]

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

[SOAP 1.2]

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

[URI]

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

[XML]

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

[XML-ns]

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

[XML-Schema Part1]

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

[XML-Schema Part2]

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

[WSDL 1.1]

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

[WS-Addressing]

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

6.2 Non-Normative

[WS-Policy]

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

[WS-PolicyAttachment]

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

[WSSecurity]

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

[RTTM]

842 V. Jacobson, R. Braden, D. Borman, "[TCP Extensions for High Performance](#)", RFC 1323, May
843 1992.

844 **[SecurityPolicy]**

845 G. Della-Libra, et. al. "[Web Services Security Policy Language \(WS-SecurityPolicy\)](#)", July 2005

846 **[SecureConversation]**

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

848 A. Schema

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

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

852 The following copy is provided for reference.

```
853 <?xml version="1.0" encoding="UTF-8"?>
854 <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
855 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
856 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
857 targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"
858 elementFormDefault="qualified" attributeFormDefault="unqualified">
859   <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
860   schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
861   <!-- Protocol Elements -->
862   <xs:complexType name="SequenceType">
863     <xs:sequence>
864       <xs:element ref="wsrm:Identifier"/>
865       <xs:element name="MessageNumber" type="xs:unsignedLong"/>
866       <xs:any namespace="##other" processContents="lax" minOccurs="0"
867 maxOccurs="unbounded"/>
868     </xs:sequence>
869     <xs:anyAttribute namespace="##other" processContents="lax"/>
870   </xs:complexType>
871   <xs:element name="Sequence" type="wsrm:SequenceType"/>
872   <xs:element name="SequenceAcknowledgement">
873     <xs:complexType>
874       <xs:sequence>
875         <xs:element ref="wsrm:Identifier"/>
876         <xs:choice>
877           <xs:sequence>
878             <xs:choice>
879               <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
880                 <xs:complexType>
881                   <xs:sequence/>
882                   <xs:attribute name="Upper" type="xs:unsignedLong"
883 use="required"/>
884                   <xs:attribute name="Lower" type="xs:unsignedLong"
885 use="required"/>
886                 <xs:anyAttribute namespace="##other" processContents="lax"/>
887               </xs:complexType>
888             </xs:element>
889             <xs:element name="None" minOccurs="0">
890               <xs:complexType>
891                 <xs:sequence/>
892               </xs:complexType>
893             </xs:element>
894           </xs:choice>
895           <xs:element name="Final" minOccurs="0">
896             <xs:complexType>
897               <xs:sequence/>
898             </xs:complexType>
899           </xs:element>
900         </xs:sequence>
901         <xs:element name="Nack" type="xs:unsignedLong"
902 maxOccurs="unbounded"/>
903       </xs:choice>
```

```

902     <xs:any namespace="##other" processContents="lax" minOccurs="0"
903 maxOccurs="unbounded"/>
904   </xs:sequence>
905   <xs:anyAttribute namespace="##other" processContents="lax"/>
906 </xs:complexType>
907 </xs:element>
908 <xs:complexType name="AckRequestedType">
909   <xs:sequence>
910     <xs:element ref="wsrm:Identifier"/>
911     <xs:any namespace="##other" processContents="lax" minOccurs="0"
912 maxOccurs="unbounded"/>
913   </xs:sequence>
914   <xs:anyAttribute namespace="##other" processContents="lax"/>
915 </xs:complexType>
916 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
917 <xs:element name="Identifier">
918   <xs:complexType>
919     <xs:annotation>
920       <xs:documentation>
921         This type is for elements whose [children] is an anyURI and can have
922 arbitrary attributes.
923       </xs:documentation>
924     </xs:annotation>
925     <xs:simpleContent>
926       <xs:extension base="xs:anyURI">
927         <xs:anyAttribute namespace="##other" processContents="lax"/>
928       </xs:extension>
929     </xs:simpleContent>
930   </xs:complexType>
931 </xs:element>
932 <!-- Fault Container and Codes -->
933 <xs:simpleType name="FaultCodes">
934   <xs:restriction base="xs:QName">
935     <xs:enumeration value="wsrm:UnknownSequence"/>
936     <xs:enumeration value="wsrm:SequenceTerminated"/>
937     <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
938     <xs:enumeration value="wsrm:MessageNumberRollover"/>
939     <xs:enumeration value="wsrm:CreateSequenceRefused"/>
940   </xs:restriction>
941 </xs:simpleType>
942 <xs:complexType name="SequenceFaultType">
943   <xs:sequence>
944     <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
945     <xs:any namespace="##any" processContents="lax" minOccurs="0"
946 maxOccurs="unbounded"/>
947   </xs:sequence>
948   <xs:anyAttribute namespace="##any" processContents="lax"/>
949 </xs:complexType>
950 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
951 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
952 <xs:element name="CreateSequenceResponse"
953 type="wsrm:CreateSequenceResponseType"/>
954 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
955 <xs:element name="CloseSequenceResponse"
956 type="wsrm:CloseSequenceResponseType"/>
957 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
958 <xs:complexType name="CreateSequenceType">
959   <xs:sequence>
960     <xs:element ref="wsrm:AcksTo"/>
961     <xs:element ref="wsrm:Expires" minOccurs="0"/>
962     <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
963     <xs:any namespace="##other" processContents="lax" minOccurs="0"

```

```

964 maxOccurs="unbounded">
965     <xs:annotation>
966         <xs:documentation>
967             It is the authors intent that this extensibility be used to
968 transfer a Security Token Reference as defined in WS-Security.
969         </xs:documentation>
970     </xs:annotation>
971 </xs:any>
972 </xs:sequence>
973 <xs:anyAttribute namespace="##other" processContents="lax"/>
974 </xs:complexType>
975 <xs:complexType name="CreateSequenceResponseType">
976     <xs:sequence>
977         <xs:element ref="wsrm:Identifier"/>
978         <xs:element ref="wsrm:Expires" minOccurs="0"/>
979         <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
980         <xs:any namespace="##other" processContents="lax" minOccurs="0"
981 maxOccurs="unbounded"/>
982     </xs:sequence>
983     <xs:anyAttribute namespace="##other" processContents="lax"/>
984 </xs:complexType>
985 <xs:complexType name="CloseSequenceType">
986     <xs:sequence>
987         <xs:any namespace="##other" processContents="lax" minOccurs="0"
988 maxOccurs="unbounded"/>
989     </xs:sequence>
990     <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
991     <xs:anyAttribute namespace="##other" processContents="lax"/>
992 </xs:complexType>
993 <xs:complexType name="CloseSequenceResponseType">
994     <xs:sequence>
995         <xs:any namespace="##other" processContents="lax" minOccurs="0"
996 maxOccurs="unbounded"/>
997     </xs:sequence>
998     <xs:anyAttribute namespace="##other" processContents="lax"/>
999 </xs:complexType>
1000 <xs:complexType name="TerminateSequenceType">
1001     <xs:sequence>
1002         <xs:element ref="wsrm:Identifier"/>
1003         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1004 maxOccurs="unbounded"/>
1005     </xs:sequence>
1006     <xs:anyAttribute namespace="##other" processContents="lax"/>
1007 </xs:complexType>
1008 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1009 <xs:complexType name="OfferType">
1010     <xs:sequence>
1011         <xs:element ref="wsrm:Identifier"/>
1012         <xs:element ref="wsrm:Expires" minOccurs="0"/>
1013         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1014 maxOccurs="unbounded"/>
1015     </xs:sequence>
1016     <xs:anyAttribute namespace="##other" processContents="lax"/>
1017 </xs:complexType>
1018 <xs:complexType name="AcceptType">
1019     <xs:sequence>
1020         <xs:element ref="wsrm:AcksTo"/>
1021         <xs:any namespace="##other" processContents="lax" minOccurs="0"
1022 maxOccurs="unbounded"/>
1023     </xs:sequence>
1024     <xs:anyAttribute namespace="##other" processContents="lax"/>
1025 </xs:complexType>

```

```
1026 <xs:element name="Expires">
1027   <xs:complexType>
1028     <xs:simpleContent>
1029       <xs:extension base="xs:duration">
1030         <xs:anyAttribute namespace="##other" processContents="lax"/>
1031       </xs:extension>
1032     </xs:simpleContent>
1033   </xs:complexType>
1034 </xs:element>
1035 </xs:schema>
```

B. Message Examples

B.1 Create Sequence

Create Sequence

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsm="http://docs.oasis-open.org/ws-rx/wsm/200510"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
    </wsa:MessageID>
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <wsa:Action>http://docs.oasis-open.org/ws-
rx/wsm/200510/CreateSequence</wsa:Action>
    <wsa:ReplyTo>
      <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
    </wsa:ReplyTo>
  </S:Header>
  <S:Body>
    <wsm:CreateSequence>
      <wsm:AcksTo>
        <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
      </wsm:AcksTo>
    </wsm:CreateSequence>
  </S:Body>
</S:Envelope>
```

Create Sequence Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
  xmlns:wsm="http://docs.oasis-open.org/ws-rx/wsm/200510"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <S:Header>
    <wsa:To>http://Business456.com/serviceA/789</wsa:To>
    <wsa:RelatesTo>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8a7c2eb546817
    </wsa:RelatesTo>
    <wsa:Action>
      http://docs.oasis-open.org/ws-rx/wsm/200510/CreateSequenceResponse
    </wsa:Action>
  </S:Header>
  <S:Body>
    <wsm:CreateSequenceResponse>
      <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
    </wsm:CreateSequenceResponse>
  </S:Body>
</S:Envelope>
```

B.2 Initial Transmission

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

1088 Message 1

```
1089 <?xml version="1.0" encoding="UTF-8"?>
1090 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1091 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1092 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1093   <S:Header>
1094     <wsa:MessageID>
1095       http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
1096     </wsa:MessageID>
1097     <wsa:To>http://example.com/serviceB/123</wsa:To>
1098     <wsa:From>
1099       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1100     </wsa:From>
1101     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1102     <wsmr:Sequence>
1103       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1104       <wsmr:MessageNumber>1</wsmr:MessageNumber>
1105     </wsmr:Sequence>
1106   </S:Header>
1107   <S:Body>
1108     <!-- Some Application Data -->
1109   </S:Body>
1110 </S:Envelope>
```

1111 Message 2

```
1112 <?xml version="1.0" encoding="UTF-8"?>
1113 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1114 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1115 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1116   <S:Header>
1117     <wsa:MessageID>
1118       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1119     </wsa:MessageID>
1120     <wsa:To>http://example.com/serviceB/123</wsa:To>
1121     <wsa:From>
1122       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1123     </wsa:From>
1124     <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1125     <wsmr:Sequence>
1126       <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
1127       <wsmr:MessageNumber>2</wsmr:MessageNumber>
1128     </wsmr:Sequence>
1129   </S:Header>
1130   <S:Body>
1131     <!-- Some Application Data -->
1132   </S:Body>
1133 </S:Envelope>
```

1134 Message 3

```
1135 <?xml version="1.0" encoding="UTF-8"?>
1136 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1137 xmlns:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510"
1138 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1139   <S:Header>
1140     <wsa:MessageID>
1141       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1142     </wsa:MessageID>
1143     <wsa:To>http://example.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://example.com/serviceB/123/request</wsa:Action>
1148     <wsrm:Sequence>
1149       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1150       <wsrm:MessageNumber>3</wsrm:MessageNumber>
1151     </wsrm:Sequence>
1152     <wsrm:AckRequested>
1153       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1154     </wsrm:AckRequested>
1155   </S:Header>
1156   <S:Body>
1157     <!-- Some Application Data -->
1158   </S:Body>
1159 </S:Envelope>

```

1160 B.3 First Acknowledgement

1161 Message number 2 has not been received by the RM Destination due to some transmission error so it
 1162 responds with an acknowledgement for messages 1 and 3:

```

1163 <?xml version="1.0" encoding="UTF-8"?>
1164 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1165   xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1166   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1167   <S:Header>
1168     <wsa:MessageID>
1169       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1170     </wsa:MessageID>
1171     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1172     <wsa:From>
1173       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1174     </wsa:From>
1175     <wsa:Action>
1176       http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement
1177     </wsa:Action>
1178     <wsrm:SequenceAcknowledgement>
1179       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1180       <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1181       <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1182     </wsrm:SequenceAcknowledgement>
1183   </S:Header>
1184   <S:Body/>
1185 </S:Envelope>

```

1186 B.4 Retransmission

1187 The RM Sourcediscovers that message number 2 was not received so it resends the message and
 1188 requests an acknowledgement:

```

1189 <?xml version="1.0" encoding="UTF-8"?>
1190 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1191   xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1192   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1193   <S:Header>
1194     <wsa:MessageID>
1195       http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1196     </wsa:MessageID>
1197     <wsa:To>http://example.com/serviceB/123</wsa:To>
1198     <wsa:From>
1199       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1200     </wsa:From>

```

```

1201 <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1202 <wsrm:Sequence>
1203   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1204   <wsrm:MessageNumber>2</wsrm:MessageNumber>
1205 </wsrm:Sequence>
1206 <wsrm:AckRequested>
1207   <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1208 </wsrm:AckRequested>
1209 </S:Header>
1210 <S:Body>
1211   <!-- Some Application Data -->
1212 </S:Body>
1213 </S:Envelope>

```

1214 B.5 Termination

1215 The RM Destination now responds with an acknowledgement for the complete sequence which can then
 1216 be terminated:

```

1217 <?xml version="1.0" encoding="UTF-8"?>
1218 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1219 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1220 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1221   <S:Header>
1222     <wsa:MessageID>
1223       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1224     </wsa:MessageID>
1225     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1226     <wsa:From>
1227       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1228     </wsa:From>
1229     <wsa:Action>
1230       http://docs.oasis-open.org/ws-rx/wsrn/200510/SequenceAcknowledgement
1231     </wsa:Action>
1232     <wsrm:SequenceAcknowledgement>
1233       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1234       <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1235     </wsrm:SequenceAcknowledgement>
1236   </S:Header>
1237   <S:Body/>
1238 </S:Envelope>

```

1239 Terminate Sequence

```

1240 <?xml version="1.0" encoding="UTF-8"?>
1241 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1242 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
1243 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1244   <S:Header>
1245     <wsa:MessageID>
1246       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1247     </wsa:MessageID>
1248     <wsa:To>http://example.com/serviceB/123</wsa:To>
1249     <wsa:Action>
1250       http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence
1251     </wsa:Action>
1252     <wsa:From>
1253       <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1254     </wsa:From>
1255   </S:Header>
1256   <S:Body>
1257     <wsrm:TerminateSequence>

```

1258	<wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1259	</wsrm:TerminateSequence>
1260	</S:Body>
1261	</S:Envelope>

C. WSDL

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

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

The following non-normative copy is provided for reference.

```
<?xml version="1.0" encoding="utf-8"?>
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:rm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
xmlns:tns="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd"
targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510/wsd">
  <wsdl:types>
    <xs:schema
      <xs:import namespace="http://docs.oasis-open.org/ws-
rx/wsrn/200510" schemaLocation="http://docs.oasis-open.org/ws-
rx/wsrn/200510/wsrn-1.1-schema-200510.xsd"/>
    </xs:schema>
  </wsdl:types>
  <wsdl:message name="CreateSequence">
    <wsdl:part name="create" element="rm:CreateSequence"/>
  </wsdl:message>
  <wsdl:message name="CreateSequenceResponse">
    <wsdl:part name="createResponse"
element="rm:CreateSequenceResponse"/>
  </wsdl:message>
  <wsdl:message name="CloseSequence">
    <wsdl:part name="close" element="rm:CloseSequence"/>
  </wsdl:message>
  <wsdl:message name="CloseSequenceResponse">
    <wsdl:part name="closeResponse"
element="rm:CloseSequenceResponse"/>
  </wsdl:message>
  <wsdl:message name="TerminateSequence">
    <wsdl:part name="terminate" element="rm:TerminateSequence"/>
  </wsdl:message>
  <wsdl:portType name="SequenceAbstractPortType">
    <wsdl:operation name="CreateSequence">
      <wsdl:input message="tns:CreateSequence"
wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/CreateSequence"/>
      <wsdl:output message="tns:CreateSequenceResponse"
wsa:Action="http://docs.oasis-open.org/ws-
rx/wsrn/200510/CreateSequenceResponse"/>
    </wsdl:operation>
    <wsdl:operation name="CloseSequence">
      <wsdl:input message="tns:CloseSequence"
wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/CloseSequence"/>
      <wsdl:output message="tns:CloseSequenceResponse"
wsa:Action="http://docs.oasis-open.org/ws-
rx/wsrn/200510/CloseSequenceResponse"/>
    </wsdl:operation>
    <wsdl:operation name="TerminateSequence">
      <wsdl:input message="tns:TerminateSequence"
wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence"/>
    </wsdl:operation>
  </wsdl:portType>
```

1317

```
</wsdl:definitions>
```

D. Acknowledgments

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

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

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

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

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

TBD

E. Revision History

Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.
ws-02	2005-07-21	Doug Davis	I011 (PT0S) added
wd-02	2005-08-16	Anish Karmarkar	Trivial editorial changes
ws-03	2005-09-15	Doug Davis	I019 and i028 (CloseSeq) added
wd-05	2005-09-26	Gilbert Pilz	i005 (Source resend of nacks messages when ack already received) added.
wd-05	2005-09-27	Doug Davis	i027 (InOrder delivery assurance spanning multiple sequences) added
wd-05	2005-09-27	Doug Davis	i020 (Semantics of "At most once" Delivery Assurance) added
wd-05	2005-09-27	Doug Davis	i034 (Fault while processing a piggy-backed RM header) added
wd-05	2005-09-27	Doug Davis	i033 (Processing model of NACKs) added
wd-05	2005-09-27	Doug Davis	i031 (AckRequested schema inconsistency) added
wd-05	2005-09-27	Doug Davis	i025 (SeqAck/None) added
wd-05	2005-09-27	Doug Davis	i029 (Remove dependency on WS-Security) added
wd-05	2005-09-27	Doug Davis	i039 (What does 'have a mU attribute' mean) added
wd-05	2005-09-27	Doug Davis	i040 (Change 'optiona'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrn/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrn:FaultCode element)
wd-06	2005-11-02	Gilbert Pilz	Start wd-06 by changing title page from cd-01.
wd-06	2005-11-03	Gilbert Pilz	i047 (Reorder spec sections)
wd-07	2005-11-17	Gilbert Pilz	Start wd-07
wd-07	2005-11-28	Doug Davis	i071 – except for period in Appendix headings
wd-07	2005-11-28	Doug Davis	i10
wd-07	2005-11-28	Doug Davis	i030
wd-07	2005-11-28	Doug Davis	i037
wd-07	2005-11-28	Doug Davis	i038
wd-07	2005-11-28	Doug Davis	i041
wd-07	2005-11-28	Doug Davis	i043
wd-07	2005-11-28	Doug Davis	i044

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

Rev	Date	By Whom	What
			OpenDocument Text format. Changed line numbers to be a single style.
wd-08	2005-12-28	Anish Karmarkar	Included a section link to c:\temp\wsrm-1.1-schema-200510.xsd and to c:\temp\wsrm-1.1-wsdl-200510.wsdl
wd-08	2006-01-04	Gilbert Pilz	Fixed formatting for included sections.

F. Notices

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

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

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

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

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

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