



Web Services ReliableMessaging (WS-Reliable Messaging)

Committee Draft 02, January 13, 2006

~~Web Services Reliable Messaging (WS-Reliable Messaging)~~

~~Committee Draft 01, September 26th 2005~~

Document identifier:

wsrm-1.1-spec-cd-024

Location:

<http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1-spec-cd-024.pdf>

Editors:

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

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

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

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

[Ümit Yalçinalp, SAP](#) <umit.yalcinalp@sap.com>

TBD

Contributors:

TBD

Abstract:

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

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

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

36 **Status:**

37 This document is a Committee Draft.

38 This document was last revised or approved by the OASIS WS-RX Technical Committee on the
39 above date. The level of approval is also listed above. Check the current location noted above for
40 possible later revisions of this document.

41 For information on whether any patents have been disclosed that may be essential to
42 implementing this specification and any offers of patent licensing terms please refer to the
43 Intellectual Property Rights section of the Technical Committee web page ([http://www.oasis-](http://www.oasis-open.org/committees/ws-rx/ipr.php)
44 [open.org/committees/ws-rx/ipr.php](http://www.oasis-open.org/committees/ws-rx/ipr.php)).

Table of Contents

1	Introduction.....	4
1.1	Goals and Requirements.....	4
1.1.1	Requirements.....	4
1.2	Notational Conventions.....	4
1.3	Namespace.....	4
1.4	Compliance.....	5
2	Reliable Messaging Model.....	6
2.1	Glossary.....	6
2.2	Protocol Preconditions.....	7
2.3	Protocol Invariants.....	7
2.4	Example Message Exchange.....	7
3	RM Protocol Elements.....	10
3.1	Sequence Creation.....	10
3.2	Closing A Sequence.....	13
3.3	Sequence Termination.....	14
3.4	Sequences.....	15
3.5	Request Acknowledgement.....	16
3.6	Sequence Acknowledgement.....	16
4	Faults.....	20
4.1	SequenceFault Element.....	21
4.2	Sequence Terminated.....	22
4.3	Unknown Sequence.....	22
4.4	Invalid Acknowledgement.....	22
4.5	Message Number Rollover.....	23
4.6	Create Sequence Refused.....	23
4.7	Sequence Closed.....	23
5	Security Considerations.....	24
6	References.....	26
6.1	Normative.....	26
6.2	Non-Normative.....	26
A.	Schema.....	28
B.	Message Examples.....	33
B.1	Create Sequence.....	33
B.2	Initial Transmission.....	33
B.3	First Acknowledgement.....	35
B.4	Retransmission.....	35
B.5	Termination.....	36

83	C. WSDL.....	38
84	D. Acknowledgments.....	40
85	E. Revision History.....	41
86	F. Notices.....	44

|

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.~~ 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. 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, 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/ws-rx/wsrn/200510/rm/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.

The following namespaces are used in this document:

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

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

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

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

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

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

1.4 Compliance

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

147 ~~An implementation is not compliant with this specification if it fails to satisfy one or more of~~
148 ~~the MUST or REQUIRED level requirements defined herein. A SOAP Node MUST NOT use the~~
149 ~~XML namespace identifier for this specification (listed in Section Namespace) within SOAP~~
150 ~~Envelopes unless it is compliant with this specification.~~
151 Normative text within this specification takes precedence over normative outlines, which in turn take
152 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. 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.

—

~~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.~~

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

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

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

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

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

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

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

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

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

Figure 1 below illustrates the entities and events in a simple reliable message exchange. 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.

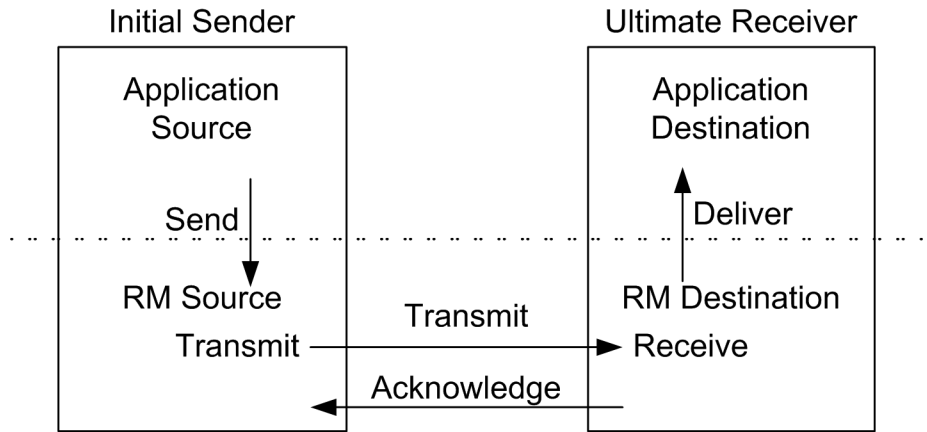


Figure 1: Reliable Messaging Model

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.

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

Application Source: The endpoint that Sends a message.

226 **Deliver:** The act of transferring a message from the RM Destination to the Application Destination. The
227 reliability guarantee is fulfilled at this point~~**Application Destination:** The endpoint to which a message is~~
228 ~~Delivered.~~

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

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

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

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

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

238 **Send:** The act of submitting a message to the RM Source for reliable delivery. The reliability guarantee
239 begins at this point~~**RM Destination:** The endpoint that receives the message.~~

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

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

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

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

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

248 2.2 Protocol Preconditions

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

- 251 • For any single message exchange the RM Source **MUST** have an endpoint reference that uniquely
252 identifies the RM Destination endpoint~~The RM Source **MUST** have an endpoint reference that~~
253 uniquely identifies the RM Destination endpoint; correlations across messages addressed to the
254 unique endpoint **MUST** be meaningful.
- 255 • The RM Source **MUST** have knowledge of the destination's policies, if any, and the RM Source
256 **MUST** be capable of formulating messages that adhere to this policy.

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

259 2.3 Protocol Invariants

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

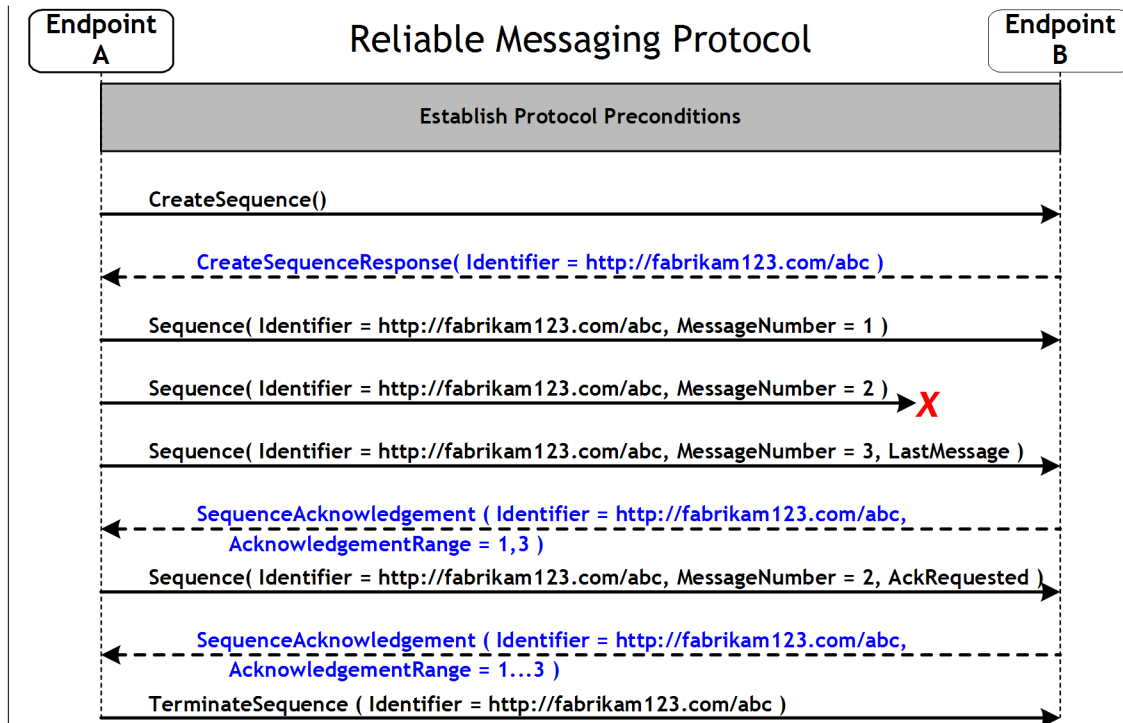
- 261 • The RM Source **MUST** assign each message within a Sequence a message number (defined
262 below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers
263 **MUST** be assigned in the same order in which messages are sent by the Application Source~~reliable~~

264 message a sequence number (defined below) beginning at 1 and increasing by exactly 1 for each
265 subsequent reliable message.

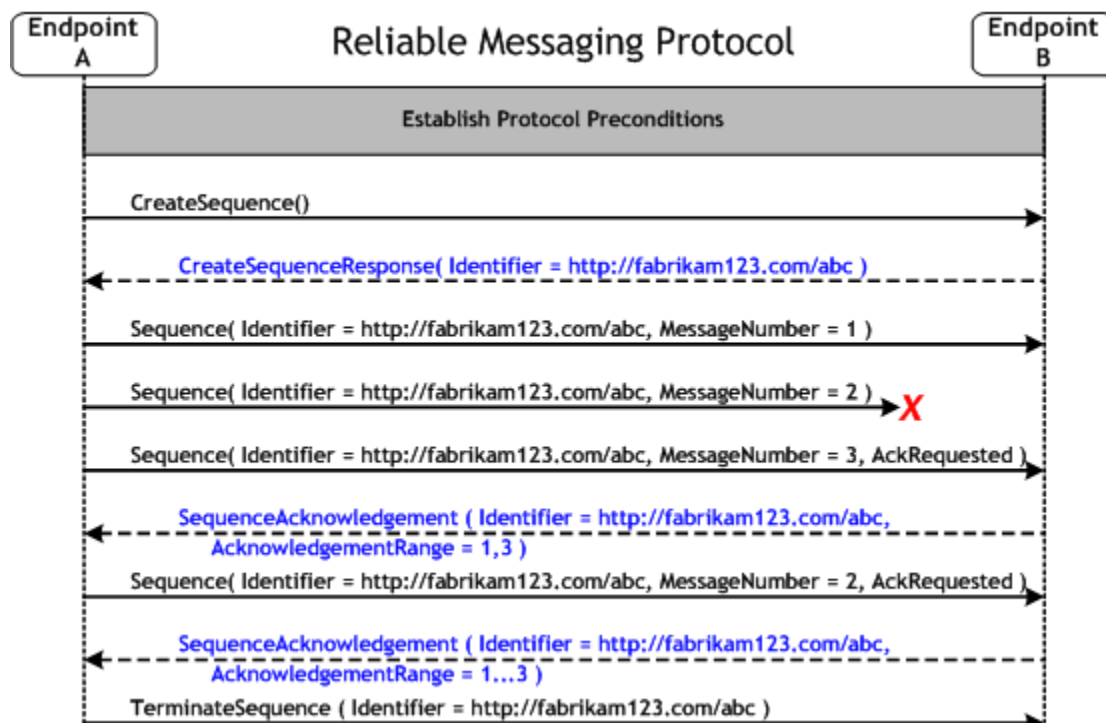
- 266 • Every acknowledgement issued by the RM Destination MUST include within an acknowledgement
267 range or ranges the sequence number of every message successfully received by the RM
268 Destination and MUST exclude sequence numbers of any messages not yet received.

269 2.4 Example Message Exchange

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



271 Figure 2: The WS-ReliableMessaging Protocol



- 272 1. The protocol preconditions are established. –These include policy exchange, endpoint resolution,
273 establishing trust.
- 274 2. The RM Source requests creation of a new Sequence.
- 275 3. The RM Destination creates a Sequence by returning a globally unique identifier.
- 276 4. The RM Source begins sending messages beginning with MessageNumber 1. [In the figure above.](#)
277 [In the figure](#) the RM Source sends 3 messages.
- 278 5. Since the 3rd message is the last in this exchange, the RM Source includes a
279 `<wsrm:AckRequested>` ~~HeaderLastMessage>~~ token.
- 280 6. The 2nd message is lost in transit.
- 281 7. The RM Destination acknowledges receipt of message numbers 1 and 3 [as a result of receiving the](#)
282 [RM Source's <wsrm:AckRequested> Header](#) ~~in response to the RM Source's~~
283 ~~<wsrm:LastMessage> token.~~
- 284 8. The RM Source retransmits the 2nd message. [This is a new message on the underlying transport.](#)
285 ~~but This is a new message on the underlying transport, but since~~ it has the same sequence
286 identifier and message number so the RM Destination can recognize it as equivalent to the earlier
287 message, in case both are received.
- 288 9. The RM Source includes an `<wsrm:AckRequested>` element so the RM Destination will expedite
289 an acknowledgement.
- 290 10. The RM Destination receives the second transmission of the message with MessageNumber 2 and
291 acknowledges receipt of message numbers 1, 2, and 3 ~~which carried the <wsrm:LastMessage>~~
292 ~~token.~~
- 293 11. The RM Source receives this acknowledgement and sends a TerminateSequence message to the
294 RM Destination indicating that the sequence is completed and reclaims any resources associated
295 with the Sequence.
- 296 12. The RM Destination receives the TerminateSequence message indicating that the RM Source will
297 not be sending any more messages, and reclaims any resources associated with the Sequence.

298 The RM Source will expect to receive acknowledgements from the RM Destination during the course of a
299 message exchange at occasions described in Section 3 below. Should the acknowledgement not be
300 received in a timely fashion, the RM Source MUST re-transmit the request since either the request or the
301 associated acknowledgement may have been lost. Since the nature and dynamic characteristics of the
302 underlying transport and potential intermediaries are unknown in the general case, the timing of re-
303 transmissions cannot be specified. Additionally, over-aggressive re-transmissions have been
304 demonstrated to cause transport or intermediary flooding which are counterproductive to the intention of
305 providing a reliable exchange of messages. Consequently, implementers are encouraged to utilize
306 adaptive mechanisms that dynamically adjust re-transmission time and the back-off intervals that are
307 appropriate to the nature of the transports and intermediaries envisioned. For the case of TCP/IP
308 transports, a mechanism similar to that described as RTTM in RFC 1323 [RTTM] should be considered.

309 Now that the basic model has been outlined, the details of the elements used in this protocol are now
310 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`

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

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

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

`/wsrm:Sequence/wsrm:MessageNumber`

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

353 /wsrm:Sequence/wsrm:LastMessage

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

356 /wsrm:Sequence/{any}

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

359 /wsrm:Sequence/@{any}

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

362 A RM Source endpoint MUST include a ~~<wsrm:LastMessage>~~ element in the
363 ~~<wsrm:Sequence>~~ element for the last message in a Sequence. An RM Destination endpoint
364 MUST respond with a ~~<wsrm:SequenceAcknowledgement>~~ upon receipt of a
365 ~~<wsrm:LastMessage>~~ element. A Sequence MUST NOT use a ~~<wsrm:MessageNumber>~~ value
366 greater than that which accompanies a ~~<wsrm:LastMessage>~~ element. An RM Destination
367 MUST generate a LastMessageNumberExceeded (See Section 4.6) fault upon receipt of such
368 a message. In the event that an RM Source needs to close a Sequence and there is no
369 application message, the RM Source MAY send a message with an empty body containing
370 ~~<wsrm:Sequence>~~ header with the ~~<wsrm:LastMessage>~~ element. In this usage, the action-
371 URI MUST be:

372 `http://docs.oasis-open.org/wsrm/200510/LastMessage`

373 in preference to the pattern defined in Section 1.2.

374 The following example illustrates a Sequence header block.

```
375 <wsrm:Sequence>  
376   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
377   <wsrm:MessageNumber>10</wsrm:MessageNumber>  
378   <wsrm:LastMessage/>  
379 </wsrm:Sequence>
```

380 3.2 Sequence Acknowledgement

381 The RM Destination informs the RM Source of successful message receipt using a
382 ~~<wsrm:SequenceAcknowledgement>~~ header block. The ~~<wsrm:SequenceAcknowledgement>~~
383 header block MAY be transmitted independently or included on return messages. The RM
384 Destination MAY send a ~~<wsrm:SequenceAcknowledgement>~~ header block at any point
385 during which the sequence is valid. The timing of acknowledgements can be advertised
386 using policy and acknowledgements can be explicitly requested using the
387 ~~<wsrm:AckRequested>~~ directive (see Section 3.3). If a non-mustUnderstand fault occurs
388 when processing an RM Header that was piggy-backed on another message, a fault MUST
389 be generated, but the processing of the original message MUST NOT be affected.

390 The following exemplar defines its syntax:

```
391 <wsrm:SequenceAcknowledgement ...>  
392   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
393   [ [ <wsrm:AcknowledgementRange ...>  
394     Upper="xs:unsignedLong"  
395     Lower="xs:unsignedLong"/> ]  
396     <wsrm:Final/> ? ]  
397   | <wsrm:Nack> xs:unsignedLong </wsrm:Nack> ]
```

```

398 | <wsrm:None/> }
399 ...
400 </wsrm:SequenceAcknowledgement>

```

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

403 ~~/wsrm:SequenceAcknowledgement~~

404 This element contains the Sequence acknowledgement information.

405 ~~/wsrm:SequenceAcknowledgement/wsrm:Identifier~~

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

408 ~~/wsrm:SequenceAcknowledgement/wsrm:Identifier/@{any}~~

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

411 ~~/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange~~

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

416 ~~/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper~~

417 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.

419 ~~/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower~~

420 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.

422 ~~/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@{any}~~

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

425 ~~/wsrm:SequenceAcknowledgement/wsrm:Final~~ —

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

432 ~~/wsrm:SequenceAcknowledgement/wsrm:Nack~~

433 This ~~OPTIONAL~~ element, if present, ~~MUST~~ contain an xs:unsignedLong representing the ~~<wsrm:MessageNumber>~~ of an unreceived message in a Sequence. This element permits the gap analysis of the ~~<wsrm:AcknowledgementRange>~~ elements to be performed at the RM Destination rather than at the RM Source which may yield performance benefits in certain environments. The ~~<wsrm:Nack>~~ element ~~MUST NOT~~ be present if either the ~~<wsrm:AcknowledgementRange>~~ or ~~<wsrm:None>~~ elements are also present as a child of ~~<wsrm:SequenceAcknowledgement>~~. Upon the

439 receipt of a Nack, an RM Source SHOULD retransmit the message identified by the Nack. The RM
440 Destination MUST NOT issue a `<wsrm:SequenceAcknowledgement>` containing a `<wsrm:Nack>` for
441 a message that it has previously acknowledged within a `<wsrm:AcknowledgementRange>`. The RM
442 Source SHOULD ignore a `<wsrm:SequenceAcknowledgement>` containing a `<wsrm:Nack>` for a
443 message that has previously been acknowledged within a `<wsrm:AcknowledgementRange>`.

444 `/wsrm:SequenceAcknowledgement/wsrm:None`

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

449 `/wsrm:SequenceAcknowledgement/{any}`

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

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

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

455 The following examples illustrate `<wsrm:SequenceAcknowledgement>` elements:

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

```
457 <wsrm:SequenceAcknowledgement>  
458   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
459   <wsrm:AcknowledgementRange Upper="10" Lower="1"/>  
460 </wsrm:SequenceAcknowledgement>
```

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

```
463 <wsrm:SequenceAcknowledgement>  
464   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
465   <wsrm:AcknowledgementRange Upper="2" Lower="1"/>  
466   <wsrm:AcknowledgementRange Upper="6" Lower="4"/>  
467   <wsrm:AcknowledgementRange Upper="10" Lower="8"/>  
468 </wsrm:SequenceAcknowledgement>
```

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

```
470 <wsrm:SequenceAcknowledgement>  
471   <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>  
472   <wsrm:Nack>3</wsrm:Nack>  
473 </wsrm:SequenceAcknowledgement>
```

474 3.3 Request Acknowledgement

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

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

479 The RM Source endpoint requests this acknowledgement by including an
480 `<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

482 ~~message containing a `<wsrm:SequenceAcknowledgement>` header block. If a non-~~
483 ~~mustUnderstand fault occurs when processing an RM Header that was piggy-backed on-~~
484 ~~another message, a fault MUST be generated, but the processing of the original message-~~
485 ~~MUST NOT be affected.~~

486 The following exemplar defines its syntax:

```
487 <wsrm:AckRequested ...>  
488   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
489   <wsrm:MessageNumber> xs:unsignedLong </wsrm:MessageNumber> ?  
490   ...  
491 </wsrm:AckRequested>
```

492 `/wsrm:AckRequested`

493 ~~This element requests an acknowledgement for the identified sequence.~~

494 `/wsrm:AckRequested/wsrm:Identifier`

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

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

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

500 `/wsrm:AckRequested/wsrm:MessageNumber`

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

505 `/wsrm:AckRequested/{any}`

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

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

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

511 3.4 Sequence Creation

512 The RM Source MUST request creation of an outbound Sequence by sending a
513 `<wsrm:CreateSequence>` element in the body of a message to the RM Destination which in turn
514 responds either with a `<wsrm:CreateSequenceResponse>` or a `CreateSequenceRefused` fault in
515 the body of the response message. `<wsrm:CreateSequence>` MAY carry an offer to create an inbound
516 sequence which is either accepted or rejected in the `<wsrm:CreateSequenceResponse>`. Note,
517 offering a Sequence within the `<wsrm:CreateSequence>` element is simply a protocol optimization.
518 There is no semantic difference between offering a Sequence, and choosing not to offer one and
519 subsequently creating a new Sequence to carry messages from the RM Destination to the RM Source.
520 ~~`<wsrm:CreateSequence>` MAY carry an offer to create an inbound sequence which is either accepted-~~
521 ~~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:

```
526 <wsrm:CreateSequence ...>
527   <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>
528   <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
529   <wsrm:Offer ...>
530     <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
531     <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
532     ...
533   </wsrm:Offer> ?
534   ...
535 </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}`

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

`/wsrm:CreateSequence/wsrm:Offer`

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

`/wsrm:CreateSequence/wsrm:Offer/wsrm:Identifier`

This REQUIRED element MUST contain an absolute URI conformant with RFC3982396 that uniquely identifies the offered Sequence.

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

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

565 /wsrm:CreateSequence/wsrm:Offer/wsrm:Expires

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

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

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

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

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

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

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

578 **OPTIONAL** /wsrm:CreateSequence/{any}

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

581 /wsrm:CreateSequence/@{any}

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

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

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

```
589 <wsrm:CreateSequenceResponse ...>  
590   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
591   <wsrm:Expires> xs:duration </wsrm:Expires> ?  
592   <wsrm:Accept ...>  
593     <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>  
594     ...  
595   </wsrm:Accept> ?  
596   ...  
597 </wsrm:CreateSequenceResponse>
```

598 /wsrm:CreateSequenceResponse

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

602 /wsrm:CreateSequenceResponse/wsrm:Identifier

603 This REQUIRED element MUST contain an absolute URI conformant with RFC3982396 of the Sequence
604 that has been created by the RM Destination.

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

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

608 /wsrm:CreateSequenceResponse/wsrm:Expires

609 This element, if present, of type `xs:duration` accepts or refines the RM Source's requested duration for
610 the Sequence. A value of 'PT0S' indicates that the Sequence will never expire. Absence of the element
611 indicates an implied value of 'PT0S'. This value MUST be equal [to or lesser-lesser](#) than the value
612 requested by the RM Source in the corresponding `<wsrm:CreateSequence>` message.

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

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

616 /wsrm:CreateSequenceResponse/wsrm:Accept

617 This element, if present, enables an RM Destination to accept the offer of a corresponding Sequence for
618 the reliable exchange of messages transmitted from RM Destination to RM Source. ~~This element MUST~~
619 ~~be present if the corresponding <wsrm:CreateSequence> message contained an <wsrm:Offer>~~
620 ~~element.~~

621 [Note: If a <wsrm:CreateSequenceResponse> is returned without a child <wsrm:Accept> in response](#)
622 [to a <wsrm:CreateSequence> that did contain a child <wsrm:Offer>, then the RM Source MAY](#)
623 [immediately reclaim any resources associated with the unused offered Sequence.](#)

624 /wsrm:CreateSequenceResponse/wsrm:Accept/wsrm:AcksTo

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

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

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

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

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

634 /wsrm:CreateSequenceResponse/{any}

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

637 /wsrm:CreateSequenceResponse/@{any}

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

640 **3.5 Closing A Sequence**

641 [There may be times during the use of an RM Sequence that the RM Source or RM Destination will wish to](#)
642 [discontinue using a Sequence. Simply terminating the Sequence discards the state managed by the RM](#)
643 [Destination, leaving the RM Source unaware of the final ranges of messages that were successfully](#)

644 delivered to the RM Destination. To ensure that the Sequence ends with a known final state both the RM
645 Source and RM Destination may choose to 'close' the Sequence before terminating it.

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

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

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

662 The following exemplar defines the CloseSequence syntax:

```
663 <wsrm:CloseSequence wsrm:Identifier="xs:anyURI" ...>  
664 ...  
665 </wsrm:CloseSequence>
```

666 /wsrm:CloseSequence

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

670 /wsrm:CloseSequence@Identifier

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

673 /wsrm:CloseSequence/{any}

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

676 /wsrm:CloseSequence@{any}

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

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

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

```
683 <wsrm:CloseSequenceResponse ...>  
684 ...  
685 </wsrm:CloseSequenceResponse>
```

686 /wsrm:CloseSequenceResponse

687 [This element is sent in the body of a response message by an RM Destination in response to receipt of a](#)
688 [<wsrm:CloseSequence> request message. It indicates that the RM Destination has closed the](#)
689 [sequence.](#)

690 [/wsrm:CloseSequenceResponse/{any}](#)

691 [This is an extensibility mechanism to allow different \(extensible\) types of information, based on a schema,](#)
692 [to be passed.](#)

693 [/wsrm:CloseSequenceResponse@{any}](#)

694 [This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the](#)
695 [element.](#)

696 3.6 Sequence Termination

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

704 The following exemplar defines the TerminateSequence syntax:

```
705 <wsrm:TerminateSequence ...>  
706   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
707   ...  
708 </wsrm:TerminateSequence>
```

709 /wsrm:TerminateSequence

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

714 /wsrm:TerminateSequence/wsrm:Identifier

715 This REQUIRED element MUST contain an absolute URI conformant with RFC[3982396](#) of the Sequence
716 that is being terminated.

717 /wsrm:TerminateSequence/wsrm:Identifier/@{any}

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

720 /wsrm:TerminateSequence/{any}

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

723 /wsrm:TerminateSequence/@{any}

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

3.7 Sequences

The RM protocol uses a `<wsrm:Sequence>` header block to track and manage the reliable delivery of messages. Messages for which a reliable delivery is required 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.

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

A following exemplar defines its syntax:

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

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

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

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

`/wsrm:Sequence/wsrm:MessageNumber`

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

`/wsrm:Sequence/{any}`

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

`/wsrm:Sequence/@{any}`

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

The following example illustrates a Sequence header block.

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


767 `<wsrm:MessageNumber>10</wsrm:MessageNumber>`
768 `</wsrm:Sequence>`

769 3.8 Request Acknowledgement

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

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

778 The following exemplar defines its syntax:

```
779 <wsrm:AckRequested ...>  
780   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>  
781   ...  
782 </wsrm:AckRequested>
```

783 `/wsrm:AckRequested`

784 This element requests an acknowledgement for the identified sequence.

785 `/wsrm:AckRequested/wsrm:Identifier`

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

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

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

791 `/wsrm:AckRequested/{any}`

792 3.9 Closing A Sequence

793 There may be times during the use of an RM Sequence that the RM Source or RM
794 Destination will wish to discontinue using a Sequence even if some of the messages have
795 not been successfully delivered to the RM Destination.

796 In the case where the RM Source wishes to discontinue use of a sequence, while it can send
797 a `TerminateSequence` to the RM Destination, since this is a one-way message and due to
798 the possibility of late arriving (or lost) messages and Acknowledgements, this would leave
799 the RM Source unsure of the final ranges of messages that were successfully delivered to
800 the RM Destination.

801 To alleviate this, the RM Source can send a `<wsrm:CloseSequence>` element, in the body of
802 a message, to the RM Destination to indicate that RM Destination MUST NOT receive any
803 new messages for the specified sequence, other than those already received at the time the
804 `<wsrm:CloseSequence>` element is interpreted by the RMD. Upon receipt of this message
805 the RM Destination MUST send a `SequenceAcknowledgement` to the RM Source. Note, this
806 `SequenceAcknowledgement` MUST include the `<wsrm:Final>` element.

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

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

The following exemplar defines the CloseSequence syntax:

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

/wsrm:CloseSequence—

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

/wsrm:CloseSequence@Identifier

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

/wsrm:CloseSequence/{any}

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

/wsrm:AckRequested/CloseSequence@{any}

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

3.10 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> + ]
  ]
```

850
851

[...](#)
[</wsrm:SequenceAcknowledgement>](#)

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

853 [/wsrm:SequenceAcknowledgement](#)

854 This element contains the Sequence acknowledgement information.

855 [/wsrm:SequenceAcknowledgement/wsrm:Identifier](#)

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

859 [/wsrm:SequenceAcknowledgement/wsrm:Identifier/@{any}](#)

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

862 [/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange](#)

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

867 [/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper](#)

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

870 [/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower](#)

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

873 [/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@{any}](#)

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

876 [/wsrm:SequenceAcknowledgement/wsrm:Final](#)

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

883 [/wsrm:SequenceAcknowledgement/wsrm:Nack](#)

884 This OPTIONAL element, if present, MUST contain an xs:unsignedLong representing the
885 <wsrm:MessageNumber> of an unreceived message in a Sequence. This element permits the gap
886 analysis of the <wsrm:AcknowledgementRange> elements to be performed at the RM Destination
887 rather than at the RM Source which may yield performance benefits in certain environments. The
888 <wsrm:Nack> element MUST NOT be present if a sibling <wsrm:AcknowledgementRange> or
889 <wsrm:None> element is also present as a child of <wsrm:SequenceAcknowledgement>. Upon the
890 receipt of a Nack, an RM Source SHOULD retransmit the message identified by the Nack. The RM

891 Destination MUST NOT issue a `<wsrm:SequenceAcknowledgement>` containing a `<wsrm:Nack>` for
892 a message that it has previously acknowledged within a `<wsrm:AcknowledgementRange>`. The RM
893 Source SHOULD ignore a `<wsrm:SequenceAcknowledgement>` containing a `<wsrm:Nack>` for a
894 message that has previously been acknowledged within a `<wsrm:AcknowledgementRange>`.

895 `/wsrm:SequenceAcknowledgement/wsrm:None`

896 This OPTIONAL element, if present, MUST be used when the RM Destination has not received any
897 messages for the specified sequence. The `<wsrm:None>` element MUST NOT be present if a sibling
898 `<wsrm:AcknowledgementRange>` or `<wsrm:Nack>` element is also present as a child of the
899 `<wsrm:SequenceAcknowledgement>`.

900 `/wsrm:SequenceAcknowledgement/{any}`

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

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

905 `<wsrm:CloseSequenceResponse>`

906 `/wsrm:CloseSequenceResponse`

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

909 `/wsrm:CloseSequenceResponse/{any}`

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

912 `/wsrm:SequenceAcknowledgement/@{any}CloseSequenceResponse@{any}`

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

915 The following examples illustrate `<wsrm:SequenceAcknowledgement>` elements:

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

917 `<wsrm:SequenceAcknowledgement>`
918 `<wsrm:Identifier>http://example.com/abc</wsrm:Identifier>`
919 `<wsrm:AcknowledgementRange Upper="10" Lower="1"/>`
920 `</wsrm:SequenceAcknowledgement>`

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

923 `<wsrm:SequenceAcknowledgement>`
924 `<wsrm:Identifier>http://example.com/abc</wsrm:Identifier>`
925 `<wsrm:AcknowledgementRange Upper="2" Lower="1"/>`
926 `<wsrm:AcknowledgementRange Upper="6" Lower="4"/>`
927 `<wsrm:AcknowledgementRange Upper="10" Lower="8"/>`
928 `</wsrm:SequenceAcknowledgement>`

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

930 `<wsrm:SequenceAcknowledgement>`
931 `<wsrm:Identifier>http://example.com/abc</wsrm:Identifier>`
932 `<wsrm:Nack>3</wsrm:Nack>`
933 `</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.](#) ~~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. 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.](#) ~~URI 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>
```

```

970     <wsa:Action>
971         http://schemas.xmlsoap.org/ws/2004/08/addressing/fault
972     </wsa:Action>
973     <!-- Headers elided for clarity. -->
974 </S:Header>
975 <S:Body>
976     <S:Fault>
977         <S:Code>
978             <S:Value> [Code] </S:Value>
979             <S:Subcode>
980                 <S:Value> [Subcode] </S:Value>
981             </S:Subcode>
982         </S:Code>
983         <S:Reason>
984             <S:Text xml:lang="en"> [Reason] </S:Text>
985         </S:Reason>
986         <S:Detail>
987             [Detail]
988             ...
989         </S:Detail>
990     </S:Fault>
991 </S:Body>
992 </S:Envelope>

```

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

```

995 <S11:Envelope>
996 <S11:Header>
997     <wsrm:SequenceFault>
998         <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>
999         ...
1000     </wsrm:SequenceFault>
1001     <!-- Headers elided for clarity. -->
1002 </S11:Header>
1003 <S11:Body>
1004     <S11:Fault>
1005         <faultcode> [Code] </faultcode>
1006         <faultstring> [Reason] </faultstring>
1007     </S11:Fault>
1008 </S11:Body>
1009 </S11:Envelope>

```

1010 The properties bind to a SOAP 1.1 fault as follows when the fault is generated as a result of processing a
1011 <wsrm:CreateSequence> request message:

```

1012 <S11:Envelope>
1013 <S11:Body>
1014     <S11:Fault>
1015         <faultcode> [Subcode] </faultcode>
1016         <faultstring xml:lang="en"> [Reason] </faultstring>
1017     </S11:Fault>
1018 </S11:Body>
1019 </S11:Envelope>

```

1020 4.1 SequenceFault Element

1021 The purpose of the <wsrm:SequenceFault> element is to carry the specific details of a fault generated
1022 during the reliable messaging specific processing of a message belonging to a Sequence. -The

1023 <wsrm:SequenceFault> container MUST only be used in conjunction with the SOAP1.1 fault
1024 mechanism. It MUST NOT be used in conjunction with the SOAP1.2 binding.

1025 The following exemplar defines its syntax:

```
1026 <wsrm:SequenceFault ...>  
1027   <wsrm:FaultCode> wsrm:FaultCodes </wsrm:FaultCode>  
1028   ...  
1029 </wsrm:SequenceFault>
```

1030 The following describes the content model of the `SequenceFault` element.

1031 /wsrm:SequenceFault

1032 This is the element containing Sequence information for WS-ReliableMessaging

1033 /wsrm:SequenceFault/wsrm:FaultCode

1034 This element, if present, MUST contain a qualified name from the set of fault [\[Subcodes\]codes](#) defined
1035 below.

1036 /wsrm:SequenceFault/{any}

1037 This is an extensibility mechanism to allow different (extensible) types of information, based on a schema,
1038 to be passed.

1039 /wsrm:SequenceFault/@{any}

1040 This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the
1041 element.

1042 4.2 Sequence Terminated

1043 This fault is sent by either the RM Source or the RM Destination to indicate that [it has either encountered](#)
1044 [an unrecoverable condition, or has detected a violation of the protocol and as a consequence, has chosen](#)
1045 [to terminate the sequence. The endpoint that generates this fault should make every reasonable effort to](#)
1046 [notify the corresponding endpoint of this decision.](#)~~the endpoint that generated the fault has either-~~
1047 ~~encountered an unrecoverable condition, or has detected a violation of the protocol and as a~~
1048 ~~consequence, has chosen to terminate the sequence. The endpoint that generates this fault should make-~~
1049 ~~every reasonable effort to notify the corresponding endpoint of this decision.~~

1050 Properties:

1051 [Code] Sender or Receiver

1052 [Subcode] wsrm:SequenceTerminated

1053 [Reason] The Sequence has been terminated due to an unrecoverable error.

1054 [Detail]-

```
1055 <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
```

1056 4.3 Unknown Sequence

1057 This fault is sent by either the RM Source or the RM Destination in response to a message containing an
1058 unknown sequence identifier.-

1059 Properties:

1060 [Code] Sender
1061 [Subcode] wsrn:UnknownSequence
1062 [Reason] The value of wsrn:Identifier is not a known Sequence identifier.
1063 [Detail]

1064 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

1065 4.4 Invalid Acknowledgement

1066 This fault is sent by the RM Source in response to a `<wsrm:SequenceAcknowledgement>` that violates
1067 the cumulative acknowledgement invariant. An example of such a violation would be a
1068 SequenceAcknowledgement covering messages that have not been sent.

1069 [Code] Sender
1070 [Subcode] wsrn:InvalidAcknowledgement
1071 [Reason] The SequenceAcknowledgement violates the cumulative acknowledgement invariant.
1072 [Detail]-

1073 `<wsrm:SequenceAcknowledgement ...> ... </wsrm:SequenceAcknowledgement>`

1074 4.5 Message Number Rollover

1075 This fault is sent to indicate that message numbers for a sequence have been exhausted.

1076 Properties:

1077 [Code] Sender
1078 [Subcode] wsrn:MessageNumberRollover
1079 [Reason] The maximum value for wsrn:MessageNumber has been exceeded.

1080 [Detail]

1081 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

1082 4.6 Last Message Number Exceeded

1083 ~~This fault is sent by an RM Destination to indicate that it has received a message that has a~~
1084 ~~`<wsrm:MessageNumber>` within a Sequence that exceeds the value of the~~
1085 ~~`<wsrm:MessageNumber>` element that accompanied a `<wsrm:LastMessage>` element for the~~
1086 ~~Sequence.~~

1087 Properties:

1088 [Code] Sender

1089 [Subcode] wsrn:LastMessageNumberExceeded

1090 [Reason] ~~The value for wsrn:MessageNumber exceeds the value of the MessageNumber~~
1091 ~~accompanying a LastMessage element in this Sequence.~~

1092 [Detail]

1093 `<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>`

1094 **4.7 Create Sequence Refused**

1095 This fault is sent in response to a create sequence request that cannot be satisfied.

1096 Properties:

1097 [Code] Sender

1098 [Subcode] wsrm:CreateSequenceRefused

1099 [Reason] The create sequence request has been refused by the RM Destination.

1100 [Detail] ~~empty~~

1101 `xs:any`

1102 **4.8 Sequence Closed**

1103 This fault is sent by an RM Destination to indicate that the specified sequence has been closed. This fault
1104 MUST be generated when an RM Destination is asked to receive a message for a sequence that is
1105 closed.

1106 Properties:

1107 [Code] Sender

1108 [Subcode] wsrm:SequenceClosed

1109 [Reason] The sequence is closed and can not receive new messages.

1110 [Detail] ~~<wsrm:Identifier...> xs:anyURI </wsrm:Identifier>~~

1111 `<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 [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. ~~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. ~~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. ~~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. ~~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

1157 ~~message (or the message is considered "processed"), then it is possible (and likely) that the security sub-~~
1158 ~~system will treat subsequent copies as replays and discard them. At the same time, the reliable~~
1159 ~~messaging sub-system will likely continue to expect and even solicit the missing message(s). That is, one~~
1160 ~~aspect of security is to prevent message replay and the core tenet of reliable messaging is to replay~~
1161 ~~messages until they are acknowledged. Consequently, if the security sub-system processes a message~~
1162 ~~but a failure occurs before the reliable messaging sub-system records the message (or the message is~~
1163 ~~considered "processed"), then it is possible (and likely) that the security sub-system will treat subsequent~~
1164 ~~copies as replays and discard them. At the same time, the reliable messaging sub-system will likely~~
1165 ~~continue to expect and even solicit the missing message(s).~~ Care should be taken to avoid and prevent
1166 this rare condition.

1167 The following list summarizes common classes of attacks that apply to this protocol and identifies the
1168 mechanism to prevent/mitigate the attacks:

- 1169 • **Message alteration** – Alteration is prevented by including signatures of the message information
1170 using WS-Security.
- 1171 • **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-Security.
- 1172 • **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by comparing
1173 secured policies – see WS-Policy and WS-SecurityPolicy).
- 1174 • **Authentication** – Authentication is established using the mechanisms described in WS-Security
1175 and WS-Trust. ~~Each message is authenticated using the mechanisms described in WS-Security.~~
- 1176 • **Accountability** – Accountability is a function of the type of and string of the key and algorithms
1177 being used. ~~In many cases, a strong symmetric key provides sufficient accountability. In many~~
1178 ~~cases, a strong symmetric key provides sufficient accountability.~~ However, in some environments,
1179 strong PKI signatures are required.
- 1180 • **Availability** – All reliable messaging services are subject to a variety of availability attacks. ~~Replay~~
1181 ~~detection is a common attack and it is recommended that this be addressed by the mechanisms~~
1182 ~~described in WS-Security. (Note that because of legitimate message replays, detection should~~
1183 ~~include a differentiator besides message id such as a timestamp). Other attacks, such as network-~~
1184 ~~level denial of service attacks are harder to avoid and are outside the scope of this specification.~~
1185 ~~Replay detection is a common attack and it is recommended that this be addressed by the~~
1186 ~~mechanisms described in WS-Security. (Note that because of legitimate message replays,~~
1187 ~~detection should include a differentiator besides message id such as a timestamp). Other attacks,~~
1188 ~~such as network-level denial of service attacks are harder to avoid and are outside the scope of this~~
1189 ~~specification.~~ That said, care should be taken to ensure that minimal state is saved prior to any
1190 authenticating 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]URI~~

W3C Recommendation, "SOAP Version 1.2 Part 1: Messaging Framework" June 2003~~T. Berners-Lee, R. Fielding, L. Masinter, "Uniform Resource Identifiers (URI): Generic Syntax," RFC 2396, MIT/LCS, U.C. Irvine, Xerox Corporation, August 1998.~~

~~[URI]XML-ns~~

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

~~[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]1~~

W3C Recommendation, "XML Schema Part 1: Structures," 2 May 2001.

~~[XML-Schema Part2]2~~

W3C Recommendation, "XML Schema Part 2: Datatypes," 2 May 2001.

~~[WSDL 1.1Security]~~

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

~~[Tanenbaum]~~

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

~~[WSDL]~~

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]~~

- 1225 D. Box, et al, "[Web Services Policy Framework \(WS-Policy\)](#)," September 2004.
- 1226 **-[WS-PolicyAttachment]**
- 1227 D. ~~D~~-Box, et al, "[Web Services Policy Attachment \(WS-PolicyAttachment\)](#)," September 2004.
- 1228 **[WS-Security]**
- 1229 [Anthony Nadalin, Chris Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds. "OASIS Web Services Security:](#)
- 1230 [SOAP Message Security 1.0 \(WS-Security 2004\)". OASIS Standard 200401, March 2004.](#)
- 1231 **[RTTM]**
- 1232 [V. Jacobson, R. Braden, D. Borman, "TCP Extensions for High Performance", RFC 1323, May](#)
- 1233 [1992.](#)
- 1234 **[SecurityPolicy]**
- 1235 G. Della-Libra, [et. al. "Web Services Security Policy Language \(WS-SecurityPolicy\)", July 2005](#)~~"Web-~~
- 1236 ~~Services Security Policy Language (WS-SecurityPolicy)," December 2002.~~
- 1237 **-[SecureConversation]**
- 1238 S. Anderson, et al, "[Web Services Secure Conversation Language \(WS-SecureConversation\)](#)," May 2004.

1239 **A. Schema**

1240 [The normative schema that is defined for WS-ReliableMessaging using \[XML-Schema Part1\] and \[XML-](#)
1241 [Schema Part2\] is located at:](#)

1242 <http://docs.oasis-open.org/ws-rx/wsrn/200510/wsrn-1.1-schema-200510.xsd>

1243 -

Appendix A. Schema

The normative schema for WS-ReliableMessaging is located at:

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

The following copy is provided for reference.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
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 © OASIS Open 2002-2006. 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 does 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.
-->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrn/200510"
targetNamespace="http://docs.oasis-open.org/ws-rx/wsrn/200510"
elementFormDefault="qualified" attributeFormDefault="unqualified">
  <xs:import
namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
  <!-- Protocol Elements -->
  <xs:complexType name="SequenceType">
    <xs:sequence>
      <xs:element ref="wsrm:Identifier"/>
      <xs:element name="MessageNumber"
type="xs:unsignedLong"/>
      <xs:any namespace="##other" processContents="lax"
minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
</xs:schema>
```

```

1301         <xs:anyAttribute namespace="##other" processContents="lax"/>
1302     </xs:complexType>
1303     <xs:element name="Sequence" type="wsrm:SequenceType"/>
1304     <xs:element name="SequenceAcknowledgement">
1305         <xs:complexType>
1306             <xs:sequence>
1307                 <xs:element ref="wsrm:Identifier"/>
1308                 <xs:choice>
1309                     <xs:sequence>
1310                         <xs:choice>
1311                             <xs:element
1312 name="AcknowledgementRange" maxOccurs="unbounded">
1313             <xs:complexType>
1314             <xs:sequence/>
1315             <xs:attribute name="Upper" type="xs:unsignedLong" use="required"/>
1316             <xs:attribute name="Lower" type="xs:unsignedLong" use="required"/>
1317             <xs:anyAttribute namespace="##other" processContents="lax"/>
1318             </xs:complexType>
1319                             </xs:element>
1320                             <xs:element
1321 name="None" minOccurs="0">
1322             <xs:complexType>
1323             <xs:sequence/>
1324             </xs:complexType>
1325                             </xs:element>
1326                             </xs:choice>
1327                             <xs:element name="Final"
1328 minOccurs="0">
1329             <xs:complexType>
1330             <xs:sequence/>
1331             </xs:complexType>
1332                             </xs:element>
1333                             </xs:sequence>
1334                             <xs:element name="Nack"
1335 type="xs:unsignedLong" maxOccurs="unbounded"/>
1336                             </xs:choice>
1337                         <xs:any namespace="##other"
1338 processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
1339                     </xs:sequence>
1340                 <xs:anyAttribute namespace="##other"
1341 processContents="lax"/>
1342             </xs:complexType>
1343         </xs:element>
1344     <xs:complexType name="AckRequestedType">
1345         <xs:sequence>
1346             <xs:element ref="wsrm:Identifier"/>
1347             <xs:any namespace="##other" processContents="lax"
1348 minOccurs="0" maxOccurs="unbounded"/>
1349         </xs:sequence>
1350         <xs:anyAttribute namespace="##other" processContents="lax"/>
1351     </xs:complexType>
1352     <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>

```



```

1353     <xs:element name="Identifier">
1354         <xs:complexType>
1355             <xs:annotation>
1356                 <xs:documentation>
1357                     This type is for elements whose [children] is an anyURI and can have
1358                     arbitrary attributes.
1359                 </xs:documentation>
1360             </xs:annotation>
1361             <xs:simpleContent>
1362                 <xs:extension base="xs:anyURI">
1363                     <xs:anyAttribute namespace="##other"
1364                     processContents="lax"/>
1365                 </xs:extension>
1366             </xs:simpleContent>
1367         </xs:complexType>
1368     </xs:element>
1369     <!-- Fault Container and Codes -->
1370     <xs:simpleType name="FaultCodes">
1371         <xs:restriction base="xs:QName">
1372             <xs:enumeration value="wsrm:UnknownSequence"/>
1373             <xs:enumeration value="wsrm:SequenceTerminated"/>
1374             <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1375             <xs:enumeration value="wsrm:MessageNumberRollover"/>
1376             <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1377         </xs:restriction>
1378     </xs:simpleType>
1379     <xs:complexType name="SequenceFaultType">
1380         <xs:sequence>
1381             <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1382             <xs:any namespace="##any" processContents="lax"
1383             minOccurs="0" maxOccurs="unbounded"/>
1384         </xs:sequence>
1385         <xs:anyAttribute namespace="##any" processContents="lax"/>
1386     </xs:complexType>
1387     <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1388     <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1389     <xs:element name="CreateSequenceResponse"
1390     type="wsrm:CreateSequenceResponseType"/>
1391     <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1392     <xs:element name="CloseSequenceResponse"
1393     type="wsrm:CloseSequenceResponseType"/>
1394     <xs:element name="TerminateSequence"
1395     type="wsrm:TerminateSequenceType"/>
1396     <xs:complexType name="CreateSequenceType">
1397         <xs:sequence>
1398             <xs:element ref="wsrm:AcksTo"/>
1399             <xs:element ref="wsrm:Expires" minOccurs="0"/>
1400             <xs:element name="Offer" type="wsrm:OfferType"
1401             minOccurs="0"/>
1402             <xs:any namespace="##other" processContents="lax"
1403             minOccurs="0" maxOccurs="unbounded">
1404                 <xs:annotation>
1405                     <xs:documentation>
1406                         It is the authors intent that this extensibility be used to
1407                         transfer a Security Token Reference as defined in WS-Security.
1408                     </xs:documentation>
1409                 </xs:annotation>
1410             </xs:any>
1411         </xs:sequence>
1412         <xs:anyAttribute namespace="##other" processContents="lax"/>
1413     </xs:complexType>
1414     <xs:complexType name="CreateSequenceResponseType">

```

```

1415         <xs:sequence>
1416             <xs:element ref="wsrm:Identifier"/>
1417             <xs:element ref="wsrm:Expires" minOccurs="0"/>
1418             <xs:element name="Accept" type="wsrm:AcceptType"
1419 minOccurs="0"/>
1420             <xs:any namespace="##other" processContents="lax"
1421 minOccurs="0" maxOccurs="unbounded"/>
1422         </xs:sequence>
1423         <xs:anyAttribute namespace="##other" processContents="lax"/>
1424     </xs:complexType>
1425     <xs:complexType name="CloseSequenceType">
1426         <xs:sequence>
1427             <xs:any namespace="##other" processContents="lax"
1428 minOccurs="0" maxOccurs="unbounded"/>
1429         </xs:sequence>
1430         <xs:attribute name="Identifier" type="xs:anyURI"
1431 use="required"/>
1432         <xs:anyAttribute namespace="##other" processContents="lax"/>
1433     </xs:complexType>
1434     <xs:complexType name="CloseSequenceResponseType">
1435         <xs:sequence>
1436             <xs:any namespace="##other" processContents="lax"
1437 minOccurs="0" maxOccurs="unbounded"/>
1438         </xs:sequence>
1439         <xs:anyAttribute namespace="##other" processContents="lax"/>
1440     </xs:complexType>
1441     <xs:complexType name="TerminateSequenceType">
1442         <xs:sequence>
1443             <xs:element ref="wsrm:Identifier"/>
1444             <xs:any namespace="##other" processContents="lax"
1445 minOccurs="0" maxOccurs="unbounded"/>
1446         </xs:sequence>
1447         <xs:anyAttribute namespace="##other" processContents="lax"/>
1448     </xs:complexType>
1449     <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1450     <xs:complexType name="OfferType">
1451         <xs:sequence>
1452             <xs:element ref="wsrm:Identifier"/>
1453             <xs:element ref="wsrm:Expires" minOccurs="0"/>
1454             <xs:any namespace="##other" processContents="lax"
1455 minOccurs="0" maxOccurs="unbounded"/>
1456         </xs:sequence>
1457         <xs:anyAttribute namespace="##other" processContents="lax"/>
1458     </xs:complexType>
1459     <xs:complexType name="AcceptType">
1460         <xs:sequence>
1461             <xs:element ref="wsrm:AcksTo"/>
1462             <xs:any namespace="##other" processContents="lax"
1463 minOccurs="0" maxOccurs="unbounded"/>
1464         </xs:sequence>
1465         <xs:anyAttribute namespace="##other" processContents="lax"/>
1466     </xs:complexType>
1467     <xs:element name="Expires">
1468         <xs:complexType>
1469             <xs:simpleContent>
1470                 <xs:extension base="xs:duration">
1471                     <xs:anyAttribute namespace="##other"
1472 processContents="lax"/>
1473                 </xs:extension>
1474             </xs:simpleContent>
1475         </xs:complexType>
1476     </xs:element>

```

```

1477 </xs:schema>
1478
1479 <xs:schema targetNamespace="http://docs.oasis-open.org/wsrn/200510/"
1480 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1481 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1482 xmlns:wsrm="http://docs.oasis-open.org/wsrn/200510/"
1483 elementFormDefault="qualified" attributeFormDefault="unqualified">
1484 <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1485 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1486 <!-- Protocol Elements -->
1487 <xs:complexType name="SequenceType">
1488 <xs:sequence>
1489 <xs:element ref="wsrm:Identifier"/>
1490 <xs:element name="MessageNumber" type="xs:unsignedLong"/>
1491 <xs:element name="LastMessage" minOccurs="0">
1492 <xs:complexType>
1493 <xs:sequence/>
1494 </xs:complexType>
1495 </xs:element>
1496 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1497 maxOccurs="unbounded"/>
1498 </xs:sequence>
1499 <xs:anyAttribute namespace="##other" processContents="lax"/>
1500 </xs:complexType>
1501 <xs:element name="Sequence" type="wsrm:SequenceType"/>
1502 <xs:element name="SequenceAcknowledgement">
1503 <xs:complexType>
1504 <xs:sequence>
1505 <xs:element ref="wsrm:Identifier"/>
1506 <xs:choice>
1507 <ws:sequence>
1508 <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
1509 <xs:complexType>
1510 <xs:sequence/>
1511 <xs:attribute name="Upper" type="xs:unsignedLong"
1512 use="required"/>
1513 <xs:attribute name="Lower" type="xs:unsignedLong"
1514 use="required"/>
1515 <xs:anyAttribute namespace="##other" processContents="lax"/>
1516 </xs:complexType>
1517 </xs:element>
1518 <ws:element name="Final" minOccurs="0">
1519 <xs:complexType>
1520 <xs:sequence/>
1521 </xs:complexType>
1522 </ws:element>
1523 </ws:sequence>
1524 <xs:element name="Nack" type="xs:unsignedLong"
1525 maxOccurs="unbounded"/>
1526 <xs:element name="None" minOccurs="0">
1527 <xs:complexType>
1528 <xs:sequence/>
1529 </xs:complexType>
1530 </xs:element>
1531 </xs:choice>
1532 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1533 maxOccurs="unbounded"/>
1534 </xs:sequence>
1535 <xs:anyAttribute namespace="##other" processContents="lax"/>
1536 </xs:complexType>
1537 </xs:element>
1538 <xs:complexType name="AckRequestedType">

```

```

1538 <xs:sequence>
1539 <xs:element ref="wsrm:Identifier"/>
1540 <xs:element name="MessageNumber" type="xs:unsignedLong" minOccurs="0"/>
1541 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1542 maxOccurs="unbounded"/>
1543 </xs:sequence>
1544 <xs:anyAttribute namespace="##other" processContents="lax"/>
1545 </xs:complexType>
1546 <xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
1547 <xs:element name="Identifier">
1548 <xs:complexType>
1549 <xs:annotation>
1550 <xs:documentation>
1551 This type is for elements whose [children] is an anyURI and can have arbitrary
1552 attributes.
1553 </xs:documentation>
1554 </xs:annotation>
1555 <xs:simpleContent>
1556 <xs:extension base="xs:anyURI">
1557 <xs:anyAttribute namespace="##other" processContents="lax"/>
1558 </xs:extension>
1559 </xs:simpleContent>
1560 </xs:complexType>
1561 </xs:element>
1562 <!-- Fault Container and Codes -->
1563 <xs:simpleType name="FaultCodes">
1564 <xs:restriction base="xs:QName">
1565 <xs:enumeration value="wsrm:UnknownSequence"/>
1566 <xs:enumeration value="wsrm:SequenceTerminated"/>
1567 <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
1568 <xs:enumeration value="wsrm:MessageNumberRollover"/>
1569 <xs:enumeration value="wsrm:CreateSequenceRefused"/>
1570 <xs:enumeration value="wsrm:LastMessageNumberExceeded"/>
1571 </xs:restriction>
1572 </xs:simpleType>
1573 <xs:complexType name="SequenceFaultType">
1574 <xs:sequence>
1575 <xs:element name="FaultCode" type="wsrm:FaultCodes"/>
1576 <xs:any namespace="##any" processContents="lax" minOccurs="0"
1577 maxOccurs="unbounded"/>
1578 </xs:sequence>
1579 <xs:anyAttribute namespace="##any" processContents="lax"/>
1580 </xs:complexType>
1581 <xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>
1582 <xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>
1583 <xs:element name="CreateSequenceResponse"
1584 type="wsrm:CreateSequenceResponseType"/>
1585 <xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>
1586 <xs:element name="CloseSequenceResponse"
1587 type="wsrm:CloseSequenceResponseType"/>
1588 <xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>
1589 <xs:complexType name="CreateSequenceType">
1590 <xs:sequence>
1591 <xs:element ref="wsrm:AcksTo"/>
1592 <xs:element ref="wsrm:Expires" minOccurs="0"/>
1593 <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0"/>
1594 <xs:any namespace="##other" processContents="lax" minOccurs="0"
1595 maxOccurs="unbounded">
1596 <xs:annotation>
1597 <xs:documentation>
1598 It is the authors intent that this extensibility be used to transfer a
1599 Security Token Reference as defined in WS-Security.
1600 </xs:documentation>

```

```

1601     </xs:annotation>
1602   </xs:any>
1603 </xs:sequence>
1604 <xs:anyAttribute namespace="##other" processContents="lax"/>
1605 </xs:complexType>
1606 <xs:complexType name="CreateSequenceResponseType">
1607   <xs:sequence>
1608     <xs:element ref="wsrm:Identifier"/>
1609     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1610     <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0"/>
1611     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1612 maxOccurs="unbounded"/>
1613   </xs:sequence>
1614   <xs:anyAttribute namespace="##other" processContents="lax"/>
1615 </xs:complexType>
1616 <xs:complexType name="CloseSequenceType">
1617   <xs:sequence>
1618     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1619 maxOccurs="unbounded"/>
1620   </xs:sequence>
1621   <xs:attribute name="Identifier" type="xs:anyURI" use="required"/>
1622   <xs:anyAttribute namespace="##other" processContents="lax"/>
1623 </xs:complexType>
1624 <xs:complexType name="CloseSequenceResponseType">
1625   <xs:sequence>
1626     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1627 maxOccurs="unbounded"/>
1628   </xs:sequence>
1629   <xs:anyAttribute namespace="##other" processContents="lax"/>
1630 </xs:complexType>
1631 <xs:complexType name="TerminateSequenceType">
1632   <xs:sequence>
1633     <xs:element ref="wsrm:Identifier"/>
1634     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1635 maxOccurs="unbounded"/>
1636   </xs:sequence>
1637   <xs:anyAttribute namespace="##other" processContents="lax"/>
1638 </xs:complexType>
1639 <xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
1640 <xs:complexType name="OfferType">
1641   <xs:sequence>
1642     <xs:element ref="wsrm:Identifier"/>
1643     <xs:element ref="wsrm:Expires" minOccurs="0"/>
1644     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1645 maxOccurs="unbounded"/>
1646   </xs:sequence>
1647   <xs:anyAttribute namespace="##other" processContents="lax"/>
1648 </xs:complexType>
1649 <xs:complexType name="AcceptType">
1650   <xs:sequence>
1651     <xs:element ref="wsrm:AcksTo"/>
1652     <xs:any namespace="##other" processContents="lax" minOccurs="0"
1653 maxOccurs="unbounded"/>
1654   </xs:sequence>
1655   <xs:anyAttribute namespace="##other" processContents="lax"/>
1656 </xs:complexType>
1657 <xs:element name="Expires">
1658   <xs:complexType>
1659     <xs:simpleContent>
1660       <xs:extension base="xs:duration">
1661         <xs:anyAttribute namespace="##other" processContents="lax"/>
1662       </xs:extension>
1663     </xs:simpleContent>

```

```
1664 </xs:complexType>
1665 </xs:element>
1666 </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:wsmr="http://docs.oasis-open.org/ws-rx/wsmr/200510rm/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/wsmr/200510/CreateSequence</wsa:Action>
    <wsa:ReplyTo>
      <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
    </wsa:ReplyTo>
  </S:Header>
  <S:Body>
    <wsmr:CreateSequence>
      <wsmr:AcksTo>
        <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
      </wsmr:AcksTo>
    </wsmr:CreateSequence>
  </S:Body>
</S:Envelope>
```

Create Sequence Response

Create Sequence Response

```
<?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/ws-rx/wsmr/200510rm/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/wsmr/200510/CreateSequenceResponse
    </wsa:Action>
  </S:Header>
  <S:Body>
    <wsmr:CreateSequenceResponse>
      <wsmr:Identifier>http://Business456.com/RM/ABC</wsmr:Identifier>
    </wsmr: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:

Message 1

```
<?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/200510rm/200510/"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfc9e
    </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>
    <wsm:Sequence>
      <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
      <wsm:MessageNumber>1</wsm:MessageNumber>
    </wsm:Sequence>
  </S:Header>
  <S:Body>
    <!-- Some Application Data -->
  </S:Body>
</S:Envelope>
```

Message 2

```
<?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/200510rm/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>
    <wsm:Sequence>
      <wsm:Identifier>http://Business456.com/RM/ABC</wsm:Identifier>
      <wsm:MessageNumber>2</wsm:MessageNumber>
    </wsm:Sequence>
  </S:Header>
  <S:Body>
    <!-- Some Application Data -->
  </S:Body>
</S:Envelope>
```

Message 3

```
<?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/200510rm/200510/"
  xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
```



```

1769 <S:Header>
1770   <wsa:MessageID>
1771     http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819
1772   </wsa:MessageID>
1773   <wsa:To>http://example.com/serviceB/123</wsa:To>
1774   <wsa:From>
1775     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1776   </wsa:From>
1777   <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1778   <wsrm:Sequence>
1779     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1780     <wsrm:MessageNumber>3</wsrm:MessageNumber>
1781   </wsrm:Sequence>
1782   <wsrm:AckRequested>
1783     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1784   </wsrm:AckRequested>
1785 </S:Header>
1786 <S:Body>
1787   <!-- Some Application Data -->
1788 </S:Body>
1789 </S:Envelope>

```

1791 B.3 First Acknowledgement

1792 Message number 2 has not been received by the RM Destination due to some transmission error so it
 1793 responds with an acknowledgement for messages 1 and 3:

```

1794 <?xml version="1.0" encoding="UTF-8"?>
1795 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1796   xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200510rm/200510/"
1797   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1798   <S:Header>
1799     <wsa:MessageID>
1800       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810
1801     </wsa:MessageID>
1802     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1803     <wsa:From>
1804       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1805     </wsa:From>
1806     <wsa:Action>
1807       http://docs.oasis-open.org/ws-rx/wsrm/200510/SequenceAcknowledgement
1808     </wsa:Action>
1809     <wsrm:SequenceAcknowledgement>
1810       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1811       <wsrm:AcknowledgementRange Upper="1" Lower="1"/>
1812       <wsrm:AcknowledgementRange Upper="3" Lower="3"/>
1813     </wsrm:SequenceAcknowledgement>
1814   </S:Header>
1815   <S:Body/>
1816 </S:Envelope>

```

1817 B.4 Retransmission

1818 The [RM Sourcesending endpoint](#) discovers that message number 2 was not received so it resends the
 1819 message and requests an acknowledgement:

```

1820 <?xml version="1.0" encoding="UTF-8"?>
1821 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1822   xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200510rm/200510/"
1823   xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">

```

```

1824 <S:Header>
1825   <wsa:MessageID>
1826     http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de
1827   </wsa:MessageID>
1828   <wsa:To>http://example.com/serviceB/123</wsa:To>
1829   <wsa:From>
1830     <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1831   </wsa:From>
1832   <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
1833   <wsrm:Sequence>
1834     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1835     <wsrm:MessageNumber>2</wsrm:MessageNumber>
1836   </wsrm:Sequence>
1837   <wsrm:AckRequested>
1838     <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1839   </wsrm:AckRequested>
1840 </S:Header>
1841 <S:Body>
1842   <!-- Some Application Data -->
1843 </S:Body>
1844 </S:Envelope>

```

1845 B.5 Termination

1846 The RM Destination now responds with an acknowledgement for the complete sequence which can then
 1847 be terminated:

```

1848 <?xml version="1.0" encoding="UTF-8"?>
1849 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1850 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200510rm/200510/"
1851 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1852   <S:Header>
1853     <wsa:MessageID>
1854       http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
1855     </wsa:MessageID>
1856     <wsa:To>http://Business456.com/serviceA/789</wsa:To>
1857     <wsa:From>
1858       <wsa:Address>http://example.com/serviceB/123</wsa:Address>
1859     </wsa:From>
1860     <wsa:Action>
1861       http://docs.oasis-open.org/ws-rx/wsrm/200510/SequenceAcknowledgement
1862     </wsa:Action>
1863     <wsrm:SequenceAcknowledgement>
1864       <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1865       <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
1866     </wsrm:SequenceAcknowledgement>
1867   </S:Header>
1868   <S:Body/>
1869 </S:Envelope>

```

1870 Terminate Sequence

```

1871 <?xml version="1.0" encoding="UTF-8"?>
1872 <S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
1873 xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200510rm/200510/"
1874 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing">
1875   <S:Header>
1876     <wsa:MessageID>
1877       http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
1878     </wsa:MessageID>
1879     <wsa:To>http://example.com/serviceB/123</wsa:To>
1880     <wsa:Action>

```

```
1881      http://docs.oasis-open.org/ws-rx/wsr/200510/TerminateSequence
1882      </wsa:Action>
1883      <wsa:From>
1884        <wsa:Address>http://Business456.com/serviceA/789</wsa:Address>
1885      </wsa:From>
1886    </S:Header>
1887    <S:Body>
1888      <wsrm:TerminateSequence>
1889        <wsrm:Identifier>http://Business456.com/RM/ABC</wsrm:Identifier>
1890      </wsrm:TerminateSequence>
1891    </S:Body>
1892  </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-wsdl-200510-rm/200510/wsd/wsrn.wsdl>

The following non-normative copy is provided for reference.

```
<?xml version="1.0" encoding="utf-8"?>
<!--
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 © OASIS Open 2002-2006. 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 does 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.
-->
<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:definitions>
```

```

1949         </wsdl:message>
1950         <wsdl:message name="CreateSequenceResponse">
1951             <wsdl:part name="createResponse"
1952 element="rm:CreateSequenceResponse"/>
1953         </wsdl:message>
1954         <wsdl:message name="CloseSequence">
1955             <wsdl:part name="close" element="rm:CloseSequence"/>
1956         </wsdl:message>
1957         <wsdl:message name="CloseSequenceResponse">
1958             <wsdl:part name="closeResponse"
1959 element="rm:CloseSequenceResponse"/>
1960         </wsdl:message>
1961         <wsdl:message name="TerminateSequence">
1962             <wsdl:part name="terminate" element="rm:TerminateSequence"/>
1963         </wsdl:message>
1964         <wsdl:portType name="SequenceAbstractPortType">
1965             <wsdl:operation name="CreateSequence">
1966                 <wsdl:input message="tns:CreateSequence"
1967 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/CreateSequence"/>
1968                 <wsdl:output message="tns:CreateSequenceResponse"
1969 wsa:Action="http://docs.oasis-open.org/ws-
1970 rx/wsrn/200510/CreateSequenceResponse"/>
1971             </wsdl:operation>
1972             <wsdl:operation name="CloseSequence">
1973                 <wsdl:input message="tns:CloseSequence"
1974 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/CloseSequence"/>
1975                 <wsdl:output message="tns:CloseSequenceResponse"
1976 wsa:Action="http://docs.oasis-open.org/ws-
1977 rx/wsrn/200510/CloseSequenceResponse"/>
1978             </wsdl:operation>
1979             <wsdl:operation name="TerminateSequence">
1980                 <wsdl:input message="tns:TerminateSequence"
1981 wsa:Action="http://docs.oasis-open.org/ws-rx/wsrn/200510/TerminateSequence"/>
1982             </wsdl:operation>
1983         </wsdl:portType>
1984     </wsdl:definitions>

1985 <wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
1986 xmlns:xs="http://www.w3.org/2001/XMLSchema"
1987 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1988 xmlns:rm="http://docs.oasis-open.org/wsrn/200510/"
1989 xmlns:tns="http://docs.oasis-open.org/wsrn/200510/wsdl"
1990 targetNamespace="http://docs.oasis-open.org/wsrn/200510/wsdl"><wsdl:types>
1991     <xs:schema
1992     <xs:import namespace="http://docs.oasis-open.org/wsrn/200510/"
1993 schemaLocation="http://docs.oasis-open.org/wsrn/200510/wsrn.xsd"/>
1994     <xs:import namespace="http://schemas.xmlsoap.org/ws/2004/08/addressing"
1995 schemaLocation="http://schemas.xmlsoap.org/ws/2004/08/addressing"/>
1996     </xs:schema>
1997 </wsdl:types>
1998     <wsdl:message name="CreateSequence">
1999         <wsdl:part name="create" element="rm:CreateSequence"/>
2000     </wsdl:message>
2001     <wsdl:message name="CreateSequenceResponse">
2002         <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
2003     </wsdl:message>
2004     <wsdl:message name="CloseSequence">
2005         <wsdl:part name="close" element="rm:CloseSequence"/>
2006     </wsdl:message>
2007     <wsdl:message name="CloseSequenceResponse">
2008         <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
2009     </wsdl:message>

```

```

2010 <wsdl:message name="TerminateSequence">
2011 <wsdl:part name="terminate" element="rm:TerminateSequence"/>
2012 </wsdl:message>
2013 <wsdl:portType name="SequenceAbstractPortType">
2014 <wsdl:operation name="CreateSequence">
2015 <wsdl:input message="tns:CreateSequence" wsa:Action="http://docs.oasis-
2016 open.org/wsrn/200510/CreateSequence"/>
2017 <wsdl:output message="tns:CreateSequenceResponse"
2018 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CreateSequenceResponse"/>
2019 </wsdl:operation>
2020 <wsdl:operation name="CloseSequence">
2021 <wsdl:input name="tns:CloseSequence" wsa:Action="http://docs.oasis-
2022 open.org/wsrn/200510/CloseSequence"/>
2023 <wsdl:output name="tns:CloseSequenceResponse"
2024 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CloseSequenceResponse"/>
2025 </wsdl:operation>
2026 <wsdl:operation name="TerminateSequence">
2027 <wsdl:input message="tns:TerminateSequence"
2028 wsa:Action="http://docs.oasis-open.org/wsrn/200510/CreateSequenceResponse"/>
2029 </wsdl:operation>
2030 </wsdl:portType>
2031 </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~~

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.
wd-08	2006-01-05	Gilbert Pilz	Created links for unused references. Fixed exemplars for CloseSequence and CloseSequenceResponse .
wd-09	2006-01-11	Doug Davis	Minor tweaks to text/typos.
cd-02	2006-01-13	Gilbert Pilz	Titles, boilerplate, etc. for cd-02
Rev	Date	By Whom	What
wd-01	2005-07-07	Christopher Ferris	Initial version created based on submission by the authors.
ws-02	2005-07-21	Doug Davis	i011 (PTOS) added
wd-02	2005-08-16	Anish Karmarkar	Trivial editorial changes
ws-03	2005-09-15	Doug Davis	i019 and i028 (CloseSeq) added
wd-05	2005-09-26	Gilbert Pilz	i005 (Source resend of nacks messages when ack already received) added.
wd-05	2005-09-27	Doug Davis	i027 (InOrder delivery assurance spanning multiple sequences) added
wd-05	2005-09-27	Doug Davis	i020 (Semantics of "At most once" Delivery Assurance) added
wd-05	2005-09-27	Doug Davis	i034 (Fault while processing a piggy-backed RM header) added
wd-05	2005-09-27	Doug Davis	i033 (Processing model of NACKs) added
wd-05	2005-09-27	Doug Davis	i031 (AckRequested schema inconsistency) added
wd-05	2005-09-27	Doug Davis	i025 (SeqAck/None) added
wd-05	2005-09-27	Doug Davis	i029 (Remove dependency on WS-Security) added
wd-05	2005-09-27	Doug Davis	i039 (What does 'have a MU attribute' mean) added
wd-05	2005-09-27	Doug Davis	i040 (Change 'optiona'/'required' to 'OPTIONAL'/'REQUIRED') added
wd-05	2005-09-30	Anish Karmarkar	i017 (Change NS to http://docs.oasis-open.org/wsrm/200510/)
wd-05	2005-09-30	Anish Karmarkar	i045 (Include SecureConversation as a reference and move it to non-normative citation)
wd-05	2005-09-30	Anish Karmarkar	i046 (change the type of wsrm:FaultCode element)

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 (2006). 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.

Copyright (C) OASIS Open (2005). All Rights Reserved.
The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

~~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.