Web Services Reliable Messaging
(WS-ReliableMessaging)

Working Draft 15, June 22, 2006

Document identifier:
wsrm-1.1-spec-wd-15

Location:

Editors:

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

Contributors:

TBD

Abstract:

This specification (WS-ReliableMessaging) describes a protocol that allows messages to be transferred reliably between nodes implementing this protocol 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] 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.

Status:

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

1 Introduction............................................................................................................. 4
  1.1 Notational Conventions.................................................................................... 4
  1.2 Namespace......................................................................................................... 4
  1.3 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......................................................................................... 14
  3.3 Sequence Termination...................................................................................... 16
  3.4 Sequences.......................................................................................................... 17
  3.5 Request Acknowledgement............................................................................... 18
  3.6 Sequence Acknowledgement........................................................................... 19
  3.7 MakeConnection............................................................................................... 22
  3.8 MessagePending............................................................................................... 23

4 Faults....................................................................................................................... 25
  4.1 SequenceFault Element................................................................................... 26
  4.2 Sequence Terminated....................................................................................... 27
  4.3 Unknown Sequence.......................................................................................... 27
  4.4 Invalid Acknowledgement............................................................................... 28
  4.5 Message Number Rollover............................................................................... 28
  4.6 Create Sequence Refused................................................................................ 28
  4.7 Sequence Closed.............................................................................................. 28
  4.8 WSRM Required............................................................................................... 29
  4.9 Unsupported Selection...................................................................................... 29

5 Security Considerations............................................................................................ 30
  6 References............................................................................................................ 32
    6.1 Normative........................................................................................................ 32
    6.2 Non-Normative............................................................................................... 32
  6.3 A. Schema......................................................................................................... 34
  6.4 B. WSDL............................................................................................................ 41
  6.5 C. Message Examples....................................................................................... 44
    6.5.1 C.1 Create Sequence................................................................................... 44
    6.5.2 C.2 Initial Transmission............................................................................ 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 transfer of messages. It defines a messaging protocol to identify, track, and manage the reliable transfer of messages between a source and a destination. It also defines a SOAP binding that is required for interoperability. Additional bindings can 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-Security], WS-Policy [WS-Policy], and other Web services specifications. Combined, these allow for a broad range of reliable, secure messaging options.

1.1 Notational Conventions

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

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

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

1.2 Namespace

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

[http://docs.oasis-open.org/ws-rx/wsrm/200604]

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

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 normative schema for WS-ReliableMessaging can be found at:

http://docs.oasis-open.org/ws-rx/wsrn/200604/wsrn-1.1-schema-200604.xsd

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

### 1.3 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 1.2) within SOAP Envelopes unless it is compliant with this specification.

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

<table>
<thead>
<tr>
<th>Prefix</th>
<th>Namespace</th>
</tr>
</thead>
<tbody>
<tr>
<td>S</td>
<td>(Either SOAP 1.1 or 1.2)</td>
</tr>
<tr>
<td>S11</td>
<td><a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a></td>
</tr>
<tr>
<td>S12</td>
<td><a href="http://www.w3.org/2003/05/soap-envelope">http://www.w3.org/2003/05/soap-envelope</a></td>
</tr>
<tr>
<td>wsrn</td>
<td><a href="http://docs.oasis-open.org/ws-rx/wsrn/200604">http://docs.oasis-open.org/ws-rx/wsrn/200604</a></td>
</tr>
<tr>
<td>wsra</td>
<td><a href="http://www.w3.org/2005/08/addressing">http://www.w3.org/2005/08/addressing</a></td>
</tr>
<tr>
<td>xs</td>
<td><a href="http://www.w3.org/2001/XMLSchema">http://www.w3.org/2001/XMLSchema</a></td>
</tr>
</tbody>
</table>
2 Reliable Messaging Model

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

The WS-ReliableMessaging specification defines an interoperable protocol that requires a Reliable Messaging (RM) Source and Reliable Messaging 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 regarding receipt of the messages transmitted. Note that this specification places no restriction on the scope of the RM Source or RM Destination entities. For example, either can span multiple WSDL Ports or endpoints.

The protocol enables the implementation of a broad range of reliability features which include ordered delivery, duplicate elimination, and guaranteed receipt. The protocol can also be implemented with a range of robustness characteristics ranging from in-memory persistence that is scoped to a single process lifetime, to replicated durable storage that is recoverable in all but the most extreme circumstances. It is expected that the endpoints will implement as many or as few of these reliability characteristics as necessary for the correct operation of the application using the protocol. Regardless of which of the reliability features is enabled, 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 transfer. The Reliable Messaging 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.

2.1 Glossary

The following definitions are used throughout this specification:

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

**Application Destination:** The endpoint to which a message is Delivered.
**Application Source**: The endpoint that sends a message.

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

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

**Endpoint references** convey the information needed to address a Web service endpoint.

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

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

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

**Send**: The act of submitting a message to the RM Source for reliable transfer.

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

### 2.2 Protocol Preconditions

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

1. For any single message exchange the RM Source MUST have an endpoint reference that uniquely identifies the RM Destination endpoint.
2. The RM Source MUST have successfully created a Sequence with the RM Destination.
3. The RM Source MUST be capable of formulating messages that adhere to the RM Destination's policies.
4. If a secure exchange of messages is REQUIRED, then the RM Source and RM Destination MUST have a security context.

### 2.3 Protocol Invariants

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

1. The RM Source MUST assign each message within a Sequence a message number (defined below) beginning at 1 and increasing by exactly 1 for each subsequent message. These numbers MUST be assigned in the same order in which messages are sent by the Application Source.
2. Within every acknowledgement it issues, the RM Destination MUST include one or more acknowledgement ranges that contain the message number of every message successfully received by the RM Destination. The RM Destination MUST exclude the message numbers of any messages it has not received.

### 2.4 Example Message Exchange

Figure 2 illustrates a possible message exchange between two reliable messaging endpoints A and B.
1. The protocol preconditions are established. These include policy exchange, endpoint resolution, and establishing trust.
2. The RM Source requests creation of a new Sequence.
3. The RM Destination creates a new Sequence and returns its unique identifier.
4. The RM Source begins transmitting messages in the Sequence beginning with MessageNumber 1. In the figure above, the RM Source sends 3 messages in the Sequence.
5. The 2nd message in the Sequence is lost in transit.
6. The 3rd message is the last in this Sequence and the RM Source includes an AckRequested header to ensure that it gets a timely SequenceAcknowledgement for the Sequence.
7. The RM Destination acknowledges receipt of message numbers 1 and 3 as a result of receiving the RM Source’s AckRequested header.
8. The RM Source retransmits the unacknowledged message with MessageNumber 2. This is a new message from the perspective of the underlying transport, but it has the same Sequence Identifier and MessageNumber so the RM Destination can recognize it as a duplicate of the earlier message, in case the original and retransmitted messages are both received. The RM Source includes an AckRequested header in the retransmitted message so the RM Destination will expedite an acknowledgement.
9. The RM Destination receives the second transmission of the message with MessageNumber 2 and acknowledges receipt of message numbers 1, 2, and 3.

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

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

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

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

The following protocol elements define extensibility points at various places. Implementations MAY add child elements and/or attributes 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.

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

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

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

   http://docs.oasis-open.org/ws-rx/wsrm/200602/CreateSequence

2. When an endpoint generates a SequenceAcknowledgement message that has no element content in the SOAP body, then the value of the wsa:Action IRI MUST be:

   http://docs.oasis-open.org/ws-rx/wsrm/200602/SequenceAcknowledgement

3. When an endpoint generates an AckRequested message that has no element content in the SOAP body, then the value of the wsa:Action IRI MUST be:

   http://docs.oasis-open.org/ws-rx/wsrm/200602/AckRequested

4. When an endpoint generates an RM fault as defined in section 4 below, the value of the wsa:Action IRI MUST be as defined in section 4 below.

3.1 Sequence Creation

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

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

The following exemplar defines the CreateSequence syntax:

```
<wsrm:CreateSequence ...>
  <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>
  <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
  <wsrm:Offer ...>
    <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
    <wsrm:Endpoint> wsa:EndpointReferenceType </wsrm:Endpoint>
    <wsrm:Expires ...> xs:duration </wsrm:Expires> ?
    <wsrm:IncompleteSequenceBehavior>
      wrm:IncompleteSequenceBehaviorType
    </wsrm:IncompleteSequenceBehavior>
  </wsrm:Offer>
</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. The RM Source MUST NOT send this element as a header block. The RM Destination MUST respond either with a CreateSequenceResponse response message or a CreateSequenceRefused fault.

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

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 "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever send Sequence Acknowledgements.

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'.

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

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.

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

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

An RM Source MUST include this element, of type wsa:EndpointReferenceType (as specified by WS-Addressing) This element specifies the endpoint reference to which WS-RM protocol messages related to the offered Sequence are to be sent.
Implementations MUST NOT use an endpoint reference in the Endpoint element that would prevent the
sending of WS-RM protocol messages. For example, using the WS-Addressing
"http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever
send WS-RM protocol messages (e.g. TerminateSequence) to the RM Source for the Offered
Sequence. Implementations MAY use the WS-RM anonymous URI template and doing so implies that
messages will be retrieved using a mechanism such as the MakeConnection message (see section
3.7).

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

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

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

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

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

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

/wsrm:CreateSequence/wsrm:Offer/wsrm:IncompleteSequenceBehavior

This element, if present, specifies the behavior that the destination will exhibit upon the closure or
termination of an incomplete Sequence. For the purposes of defining the values used, the term 'discard'
refers to behavior equivalent to the Application Destination never processing a particular message.
A value of "DiscardEntireSequence" indicates that the entire Sequence MUST be discarded if the
Sequence is closed, or terminated, when there are one or more gaps in the final
SequenceAcknowledgement.

A value of "DiscardFollowingFirstGap" indicates that messages in the Sequence beyond the first gap
MUST be discarded when there are one or more gaps in the final SequenceAcknowledgement.
The default value of "NoDiscard" indicates that no acknowledged messages in the Sequence will be
discarded.

/wsrm:CreateSequence/{any}

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

/wsrm:CreateSequence/@{any}

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

A CreateSequenceResponse is sent in the body of a response message by an RM Destination in
response to receipt of a CreateSequence request message. It carries the Identifier of the created
Sequence and indicates that the RM Source can begin sending messages in the context of the identified
Sequence.
The following exemplar defines the CreateSequenceResponse syntax:

```xml
<wsrm:CreateSequenceResponse ...
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
  <wsrm:Expires> xs:duration </wsrm:Expires> ?
  <wsrm:IncompleteSequenceBehavior>
    <wsrm:IncompleteSequenceBehaviorType/>
  </wsrm:IncompleteSequenceBehavior> ?
  <wsrm:Accept ...>
    <wsrm:AcksTo ...> wsa:EndpointReferenceType </wsrm:AcksTo>
    ...
  </wsrm:Accept> ?
  ...
</wsrm:CreateSequenceResponse>
```

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

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

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

This element, if present, of type xs:duration accepts or refines the RM Source's requested duration for the Sequence. A value of 'PT0S' indicates that the Sequence will never expire. Absence of the element indicates an implied value of 'PT0S'. The RM Destination MUST set the value of this element to be equal to or less than the value requested by the RM Source in the corresponding CreateSequence message.

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

This element, if present, specifies the behavior that the RM Destination will exhibit upon the closure of an incomplete Sequence.

A value of "DiscardEntireSequence" indicates that the entire Sequence will be discarded by the RM Destination if the Sequence is closed when there are one or more gaps in the final SequenceAcknowledgement.

A value of "DiscardFollowingFirstGap" indicates that messages in the Sequence beyond the first gap will be discarded by the RM Destination when there are one or more gaps in the final SequenceAcknowledgement.

The default value of "NoDiscard" indicates that no acknowledged messages in the Sequence will be discarded by the RM Destination.
This element, if present, enables an RM Destination to accept the offer of a corresponding Sequence for the reliable exchange of messages transmitted from RM Destination to RM Source.

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

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

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 "http://www.w3.org/2005/08/addressing/none" IRI would make it impossible for the RM Destination to ever send Sequence Acknowledgements.

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

/\wsrm:CreateSequenceResponse/\wsrm:Accept/@\{any\}
This is an extensibility mechanism to allow different (extensible) types of information, based on a schema, to be passed.

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

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

### 3.2 Closing A Sequence

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

If the RM Source wishes to close the Sequence, then it sends a CloseSequence element, in the body of a message, to the RM Destination. This message indicates that the RM Destination MUST NOT receive any new messages for the specified Sequence, other than those already received at the time the CloseSequence element is interpreted by the RM Destination. Upon receipt of this message, or subsequent to the RM Destination closing the Sequence of its own volition, the RM Destination MUST include a final SequenceAcknowledgement (within which the RM Destination MUST include the Final element) header block on any messages associated with the Sequence destined to the RM Source, including the CloseSequenceResponse message or on any Sequence fault transmitted to the RM Source.
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 is RECOMMENDED that it close the Sequence. Please see Final and the SequenceClosed fault. Whenever possible 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:

```
<wrsm:CloseSequence ...>
  <wrsm:Identifier ...> xs:anyURI </wrsm:Identifier>
  ...
</wrsm: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 already closed.

```
<wrsm:CloseSequence/wsrm:Identifier/@{any}>
This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the element.
```

```
<wrsm:CloseSequence/{any}>
This is an extensibility mechanism to allow different (extensible) types of information, based on a schema, to be passed.
```

```
<wrsm:CloseSequence@{any}>
This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the element.
```

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

The following exemplar defines the CloseSequenceResponse syntax:

```
<wrsm:CloseSequenceResponse ...>
  <wrsm:Identifier ...> xs:anyURI </wrsm:Identifier>
  ...
</wrsm:CloseSequenceResponse>
```

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

```
<wrsm:CloseSequenceResponse/wsrm:Identifier>
```
The RM Destination MUST include this element in any CloseSequenceResponse message it sends. The RM Destination MUST set the value of this element to the absolute URI (conformant with RFC3986) of the Sequence that is being closed.

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

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

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

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

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

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

### 3.3 Sequence Termination

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

The following exemplar defines the `TerminateSequence` syntax:

```xml
<wsrm:TerminateSequence ...>
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
  ...
</wsrm:TerminateSequence>
```

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

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

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

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

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

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

This is an extensibility mechanism to allow different (extensible) types of information, based on a schema, to be passed.
This is an extensibility mechanism to allow additional attributes, based on schemas, to be added to the
element.

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

The following exemplar defines the TerminateSequenceResponse syntax:

```
<wsrmi:TerminateSequenceResponse ...>
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
  ...
</wsrm:TerminateSequenceResponse>
```

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

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

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

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

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

On receipt of a TerminateSequence message an RM Destination MUST respond with a corresponding
TerminateSequenceResponse message or generate a fault.

### 3.4 Sequences

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

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

A following exemplar defines its syntax:

```
<wsrmi:Sequence ...>
  <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
```

Copyright © OASIS Open 2006. All Rights Reserved.
The following describes the content model of the Sequence header block.

```xml
<wsrm:MessageNumber> wsrm:MessageNumberType </wsrm:MessageNumber>
```

The following example illustrates a Sequence header block.

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

3.5 Request Acknowledgement

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

The RM Source MAY request an acknowledgement message from the RM Destination at any time by including an AckRequested header block in any message targeted to the RM Destination. An RM Destination that receives a message that contains an AckRequested header block MUST send a
message containing a `SequenceAcknowledgement` header block to the `AcksTo` endpoint reference (see Section 3.1). 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. It is RECOMMENDED that the RM Destination return a `AcknowledgementRange` or `None` element instead of a `Nack` element (see Section 3.6).

The following exemplar defines its syntax:

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

This element requests an acknowledgement for the identified Sequence.

```
/wsrm:AckRequested/wsrm:Identifier
```

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

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

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

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

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

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

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

### 3.6 Sequence Acknowledgement

The RM Destination informs the RM Source of successful message receipt using a `SequenceAcknowledgement` header block. The RM Destination MAY transmit the `SequenceAcknowledgement` header block independently or it MAY include the `SequenceAcknowledgement` header block on any message targeted to the `AcksTo` EPR. Acknowledgements can be explicitly requested using the `AckRequested` directive (see Section 3.5). 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.

A RM Destination MAY include a `SequenceAcknowledgement` header block on any SOAP envelope targeted to the endpoint referenced by the `AcksTo` EPR. During creation of a Sequence the RM Source MAY specify the WS-Addressing anonymous IRI as the address of the `AcksTo` EPR for that Sequence. When the RM Source specifies the WS-Addressing anonymous IRI as the address of the `AcksTo` EPR, the RM Destination MUST transmit any `SequenceAcknowledgement` headers for the created Sequence in a SOAP envelope to be transmitted on the protocol binding-specific channel. Such a channel is provided by the context of a received message.
containing a SOAP envelope that contains a Sequence header block and/or a AckRequested header block for that same Sequence identifier.

The following exemplar defines its syntax:

```
<wsrm:SequenceAcknowledgement ...>
   <wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
   [ [ [ <wsrm:AcknowledgementRange ...
   Upper="wsrm:MessageNumberType"
   Lower="wsrm:MessageNumberType"/> +
   [ <wsrm:Nack> wsrm:MessageNumberType </wsrm:Nack> + ]
   <wsrm:Final/> ? ]
   | <wsrm:Nack> wsrm:MessageNumberType </wsrm:Nack> + ]
...  
</wsrm:SequenceAcknowledgement>
```

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

/wsrm:SequenceAcknowledgement
This element contains the Sequence acknowledgement information.

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

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

/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange
The RM Destination MAY include one or more instances of this element within a SequenceAcknowledgement header block. It contains a range of Sequence MessageNumbers successfully received by the RM Destination. The ranges SHOULD NOT overlap. The RM Destination MUST NOT include this element if a sibling Nack or None element is also present as a child of SequenceAcknowledgement.

/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Upper
The RM Destination MUST set the value of this attribute equal to the message number of the highest contiguous message in a Sequence range received by the RM Destination.

/wsrm:SequenceAcknowledgement/wsrm:AcknowledgementRange/@Lower
The RM Destination MUST set the value of this attribute equal to the message number of the lowest contiguous message in a Sequence range received by the RM Destination.

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

/wsrm:SequenceAcknowledgement/wsrm:Final
The RM Destination MAY include this element within a SequenceAcknowledgement header block. This element 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. The RM Destination MUST include this element when the Sequence is closed. The RM Destination MUST NOT include this element when sending a Nack; it can only be used when sending AcknowledgementRange elements or a None.

/wsrm:SequenceAcknowledgement/wsrm:Nack

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

/wsrm:SequenceAcknowledgement/wsrm:None

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

/wsrm:SequenceAcknowledgement/\{any\}

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

/wsrm:SequenceAcknowledgement/@\{any\}

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

The following examples illustrate SequenceAcknowledgement elements:

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

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

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

  ```xml
  <wsrm:SequenceAcknowledgement>
    <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
    <wsrm:AcknowledgementRange Upper="2" Lower="1"/>
    <wsrm:AcknowledgementRange Upper="6" Lower="4"/>
    <wsrm:AcknowledgementRange Upper="10" Lower="8"/>
  </wsrm:SequenceAcknowledgement>
  ```

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

  ```xml
  <wsrm:SequenceAcknowledgement>
    <wsrm:Identifier>http://example.com/abc</wsrm:Identifier>
    <wsrm:Nack>3</wsrm:Nack>
  </wsrm:SequenceAcknowledgement>
  ```
When an endpoint is not directly addressable (e.g. behind a firewall or not able to allow incoming connections), an anonymous URI in the EPR address property can indicate such an endpoint. The WS-Addressing anonymous URI is one such anonymous URI. This specification defines a URI template (the WS-RM anonymous URI) which may be used to uniquely identify anonymous endpoint.

This URI template in an EPR indicates a protocol-specific back-channel will be established through a mechanism such as MakeConnection, defined below. When using this URI template, "{uuid}" MUST be replaced by a UUID value as defined by RFC4122[UUID]. This UUID value uniquely distinguishes the endpoint. A sending endpoint SHOULD transmit messages at endpoints identified with the URI template using a protocol-specific back-channel, including but not limited to those established with a MakeConnection message. Note, this URI is semantically similar to the WS-Addressing anonymous URI if a protocol-specific back-channel is available.

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

The following exemplar defines the MakeConnection syntax:

```xml
<wsrm:MakeConnection ...>
  <wsrm:Identifier> xs:anyURI </wsrm:Identifier> ?
  <wsrm:Address> xs:anyURI </wsrm:Address> ?
  ...
</wsrm:MakeConnection>
```

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

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

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

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

/wsrm:MakeConnection/@{any}

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

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

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

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

3.8 MessagePending

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

The following exemplar defines the MessagePending syntax:

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

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

/wsrm:MessagePending@pending

This attribute, when set to 'true', indicates that there is at least one message waiting to be retrieved. When this attribute is set to 'false' it indicates there are currently no messages waiting to be retrieved.

/wsrm:MessagePending/{any}

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

/wsrm:MessagePending/@{any}

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

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

Faults for the CreateSequence message exchange 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 or terminated Sequences are detected. WSRMRequired is a fault generated an RM Destination that requires the use of WS-RM on a received message that did not use the protocol. 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. Entities that generate Sequence faults SHOULD send those faults to the same [destination] as SequenceAcknowledgement messages. These faults are correlated using the Sequence identifier carried in the detail.

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

```
http://docs.oasis-open.org/ws-rx/wsrmm/200604/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 6 of WS-Addressing SOAP Binding.

The definitions of faults use the following properties:

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

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

<table>
<thead>
<tr>
<th>SOAP Version</th>
<th>Sender</th>
<th>Receiver</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOAP 1.1</td>
<td>S11:Client</td>
<td>S11:Server</td>
</tr>
<tr>
<td>SOAP 1.2</td>
<td>S:Sender</td>
<td>S:Receiver</td>
</tr>
</tbody>
</table>

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

```
<S:Envelope>
  <S:Header>
    <wsa:Action>
      http://docs.oasis-open.org/ws-rx/wsrmm/200604/fault
    </wsa:Action>
    <!-- Headers elided for clarity. -->
  </S:Header>
  <S:Body>
    <S:Fault>
      <S:Code>
        <S:Value> [Code] </S:Value>
      </S:Code>
      <S:Subcode>
        <S:Value> [Subcode] </S:Value>
      </S:Subcode>
      <S:Reason>
        <S:Text xml:lang="en"> [Reason] </S:Text>
      </S:Reason>
      <S:Detail>
```

wsrm-1.1-spec-wd-15
22 June 2006
Copyright © OASIS Open 2006. All Rights Reserved.
The properties above bind to a SOAP 1.1 fault as follows when the fault is triggered by processing an RM header block:

```
<S11:Envelope>
  <S11:Header>
    <wsrm:SequenceFault>
      <wsrm:Detail> [Detail] </wsrm:Detail>
    </wsrm:SequenceFault>
  </S11:Header>
  <S11:Body>
    <S11:Fault>
      <faultcode> [Code] </faultcode>
      <faultstring> [Reason] </faultstring>
    </S11:Fault>
  </S11:Body>
</S11:Envelope>
```

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

```
<S11:Envelope>
  <S11:Body>
    <S11:Fault>
      <faultcode> [Subcode] </faultcode>
      <faultstring> [Reason] </faultstring>
    </S11:Fault>
  </S11:Body>
</S11:Envelope>
```

4.1 SequenceFault Element

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

The following exemplar defines its syntax:

```
<wsrm:SequenceFault ...>
  <wsrm:Detail> ... </wsrm:Detail> ?
  ...
</wsrm:SequenceFault>
```

The following describes the content model of the SequenceFault element.

/wsrm:SequenceFault

This is the element containing Sequence information for WS-ReliableMessaging

/wsrm:SequenceFault/wsrm:FaultCode
WS-ReliableMessaging nodes that generate a \texttt{SequenceFault} MUST set the value of this element to a qualified name from the set of fault [Subcodes] defined below.

\texttt{/wsrm:SequenceFault/wsrm:Detail}

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

\texttt{/wsrm:SequenceFault/wsrm:Detail/{any}}

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

\texttt{/wsrm:SequenceFault/wsrm:Detail/@{any}}

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

\texttt{/wsrm:SequenceFault/{any}}

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

\texttt{/wsrm:SequenceFault/@{any}}

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

### 4.2 Sequence Terminated

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

Receipt of SequenceTerminated by either the RM Destination or the RM Source SHALL terminate the Sequence if it is not otherwise terminated.

Properties:

[Code] Sender or Receiver

[Subcode] \texttt{wsrm:SequenceTerminated}

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

[Detail]

\begin{verbatim}
<wsrm:Identifier ...> xs:anyURI </wsrm:Identifier>
\end{verbatim}

### 4.3 Unknown Sequence

This fault is generated by either the RM Source or the RM Destination in response to a message containing an unknown or terminated Sequence identifier. Receipt of UnknownSequence by either the RM Destination or the RM Source SHALL terminate the Sequence if it is not otherwise terminated.

Properties:

[Code] Sender

[Subcode] \texttt{wsrm:UnknownSequence}

[Reason] The value of \texttt{wsrm:Identifier} is not a known Sequence identifier.

[Detail]
4.4 Invalid Acknowledgement

This fault is generated by the RM Source in response to a SequenceAcknowledgement that violates the cumulative acknowledgement invariant. An example of such a violation would be a SequenceAcknowledgement covering messages that have not been sent.

[Code] Sender
[Subcode] wsrm:InvalidAcknowledgement
[Reason] The SequenceAcknowledgement violates the cumulative acknowledgement invariant.
[Detail]

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

4.5 Message Number Rollover

This fault is generated to indicate that message numbers for a Sequence have been exhausted.

Properties:
[Code] Sender
[Subcode] wsrm:MessageNumberRollover
[Reason] The maximum value for wsrm:MessageNumber has been exceeded.
[Detail]

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

4.6 Create Sequence Refused

This fault is generated in response to a create Sequence request that cannot be satisfied.

Properties:
[Code] Sender
[Subcode] wsrm:CreateSequenceRefused
[Reason] The create Sequence request has been refused by the RM Destination.
[Detail]

```
xs:any
```

4.7 Sequence Closed

This fault is generated by an RM Destination to indicate that the specified Sequence has been closed. This fault MUST be generated when an RM Destination is asked to receive a message for a Sequence that is closed or when an RM Destination is asked to close a Sequence that is already closed.

Properties:
[Code] Sender
[Subcode] wsrm:SequenceClosed
The Sequence is closed and can not receive new messages.

4.8 WSRM Required

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

Properties:

[Code] Sender

[Subcode] wsrm:WSRMRequired

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

[Detail]

4.9 Unsupported Selection

This fault is generated to indicate that endpoint processing the MakeConnection message does not support the selection criteria included in the extensibility section of the MakeConnection message. The QName of the unsupported element(s) are included in the detail.

Properties:

[Code] Receiver

[Subcode] wsrm:UnsupportedSelection

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

[Detail]
5 Security Considerations

It is strongly recommended that the communication between services be secured using the mechanisms described in WS-Security. In order to properly secure messages, the body and all relevant headers need to be included in the signature. Specifically, the Sequence header needs to be signed with the body in order to "bind" the two together. The 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[Trust] and WS-SecureConversation[SecureConversation]. If a Sequence is bound to a specific destination, then the security context needs to be established or shared with the destination servicing the Sequence. While the context can be established at any time, it is critical that the messages establishing the Sequence be secured even if they precede security context establishment. However, it is recommended that the security context be established first. Security contexts are independent of reliable messaging Sequences. Consequently, security contexts can come and go independent of the lifetime of the Sequence. In fact, it is recommended that the lifetime of a security context be less than the lifetime of the Sequence unless the Sequence is very short-lived.

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

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

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

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

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

- **Message alteration** – Alteration is prevented by including signatures of the message information using WS-Security.
- **Message disclosure** – Confidentiality is preserved by encrypting sensitive data using WS-Security.
• **Key integrity** – Key integrity is maintained by using the strongest algorithms possible (by comparing secured policies – see WS-Policy and WS-SecurityPolicy).

• **Authentication** – Authentication is established using the mechanisms described in WS-Security and WS-Trust. Each message is authenticated using the mechanisms described in WS-Security.

• **Accountability** – Accountability is a function of the type of and string of the key and algorithms being used. In many cases, a strong symmetric key provides sufficient accountability. However, in some environments, strong PKI signatures are required.

• **Availability** – All reliable messaging services are subject to a variety of availability attacks. Replay detection is a common attack and it is recommended that this be addressed by the mechanisms described in WS-Security. (Note that because of legitimate message replays, detection should include a differentiator besides message id such as a timestamp). Other attacks, such as network-level denial of service attacks are harder to avoid and are outside the scope of this specification. That said, care should be taken to ensure that minimal state is saved prior to any 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]

[SOAP 1.2]

[URI]

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

[XML]

[XML-ns]

[XML-Schema Part1]

[XML-Schema Part2]

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

[WS-Addressing]

6.2 Non-Normative

[RDDL 2.0]
Johnathan Borden, Tim Bray, eds. “Resource Directory Description Language (RDDL) 2.0,” January 2004

[WS-Policy]
W3C Member Submission, "Web Services Policy Framework (WS-Policy)," April 2006.
[WS-PolicyAttachment]

[WS-Security]

[RTTM]

[SecurityPolicy]

[SecureConversation]

[Trust]
A. Schema

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

http://docs.oasis-open.org/ws-rx/wsrm/200604/wsrm-1.1-schema-200604.xsd

The following copy is provided for reference.
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns:wsa="http://www.w3.org/2005/08/addressing"
  xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
  targetNamespace="http://docs.oasis-open.org/ws-rx/wsrm/200604"
  elementFormDefault="qualified" attributeFormDefault="unqualified">
  <!-- Protocol Elements -->
  <xs:complexType name="SequenceType">
    <xs:sequence>
      <xs:element ref="wsrm:Identifier"/>
      <xs:element name="MessageNumber" type="wsrm:MessageNumberType"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>
  <xs:element name="Sequence" type="wsrm:SequenceType"/>
  <xs:element name="SequenceAcknowledgement">
    <xs:complexType>
      <xs:sequence>
        <xs:element ref="wsrm:Identifier"/>
        <xs:choice>
          <xs:sequence>
            <xs:choice>
              <xs:element name="AcknowledgementRange" maxOccurs="unbounded">
            </xs:element>
          </xs:choice>
        </xs:choice>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
<xs:sequence/>
<xs:attribute name="Upper" type="xs:unsignedLong" use="required"/>
<xs:attribute name="Lower" type="xs:unsignedLong" use="required"/>
<xs:anyAttribute namespace="#other" processContents="lax"/>
</xs:complexType>
</xs:element>
<xs:element name="None" minOccurs="0">
<xs:complexType>
<xs:sequence/>
</xs:complexType>
</xs:element>
</xs:choice>
<xs:element name="Final" minOccurs="0">
<xs:complexType>
<xs:sequence/>
</xs:complexType>
</xs:element>
</xs:sequence>
<xs:element name="Nack" type="xs:unsignedLong" maxOccurs="unbounded"/>
</xs:choice>
<xs:element name="AckRequested" type="wsrm:AckRequestedType"/>
<xs:complexType name="wsrm:MessagePendingType">
<xs:sequence>
<xs:any namespace="#other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:attribute name="pending" type="xs:boolean"/>
<xs:anyAttribute namespace="#other" processContents="lax"/>
</xs:complexType>
</xs:element>
<xs:element name="Identifier">
<xs:complexType>
<xs:annotation>
<xs:documentation>
This type is for elements whose [children] is an anyURI and can have arbitrary attributes.
</xs:documentation>
</xs:annotation>
<xs:simpleContent>
<xs:extension base="xs:anyURI">
<xs:anyAttribute namespace="#other" processContents="lax"/>
</xs:extension>
</xs:complexType>
</xs:element>
</xs:complexType>
</xs:element>
<xs:complexType>
  <xs:extension base="xs:anyURI">
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:extension>
</xs:complexType>

<xs:element name="MakeConnectionType">
  <xs:complexType>
    <xs:sequence>
      <xs:element ref="wsrm:Identifier" minOccurs="0" maxOccurs="1"/>
      <xs:element ref="wsrm:Address" minOccurs="0" maxOccurs="1"/>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:sequence>
    <xs:anyAttribute namespace="##other" processContents="lax"/>
  </xs:complexType>
</xs:element>

<!-- Fault Container and Codes -->
<xs:complexType name="FaultCodes">
  <xs:restriction base="xs:QName">
    <xs:enumeration value="wsrm:SequenceTerminated"/>
    <xs:enumeration value="wsrm:UnknownSequence"/>
    <xs:enumeration value="wsrm:InvalidAcknowledgement"/>
    <xs:enumeration value="wsrm:MessageNumberRollover"/>
    <xs:enumeration value="wsrm:CreateSequenceRefused"/>
    <xs:enumeration value="wsrm:SequenceClosed"/>
    <xs:enumeration value="wsrm:WSRMRequired"/>
    <xs:enumeration value="wsrm:UnsupportedSelection"/>
  </xs:restriction>
</xs:complexType>

<xs:element name="SequenceFault" type="wsrm:SequenceFaultType"/>

<xs:element name="CreateSequence" type="wsrm:CreateSequenceType"/>

<xs:element name="CreateSequenceResponse" type="wsrm:CreateSequenceResponseType"/>

<xs:element name="CloseSequence" type="wsrm:CloseSequenceType"/>

<xs:element name="CloseSequenceResponse" type="wsrm:CloseSequenceResponseType"/>

<xs:element name="TerminateSequence" type="wsrm:TerminateSequenceType"/>

<xs:element name="TerminateSequenceResponse" type="wsrm:TerminateSequenceResponseType"/>
<xs:complexType name="CreateSequenceType">
  <xs:sequence>
    <xs:element ref="wsrm:AcksTo"/>
    <xs:element ref="wsrm:Expires" minOccurs="0"/>
    <xs:element name="Offer" type="wsrm:OfferType" minOccurs="0" maxOccurs="unbounded">
      <xs:annotation>
        <xs:documentation>
          It is the authors intent that this extensibility be used to
          transfer a Security Token Reference as defined in WS-Security.
        </xs:documentation>
      </xs:annotation>
      <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
    </xs:element>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:complexType name="CreateSequenceResponseType">
  <xs:sequence>
    <xs:element ref="wsrm:Identifier"/>
    <xs:element ref="wsrm:Expires" minOccurs="0"/>
    <xs:element name="IncompleteSequenceBehavior" type="wsrm:IncompleteSequenceBehaviorType" minOccurs="0" maxOccurs="unbounded">
      <xs:element name="Accept" type="wsrm:AcceptType" minOccurs="0" maxOccurs="unbounded"/>
    </xs:element>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:complexType name="CloseSequenceType">
  <xs:sequence>
    <xs:element ref="wsrm:Identifier"/>
    <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>

<xs:complexType name="CloseSequenceResponseType">
  <xs:sequence>
    <xs:element ref="wsrm:Identifier"/>
    <xs:any namespace="##other" processContents="lax" maxOccurs="unbounded"/>
  </xs:sequence>
  <xs:anyAttribute namespace="##other" processContents="lax"/>
</xs:complexType>
<xs:sequence>
  <xs:element ref="wsrm:Identifier"/>
  <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:sequence>
<xs:complexType>
  <xs:element ref="wsrm:Identifier"/>
  <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
</xs:complexType>
<xs:complexType name="TerminateSequenceType">
  <xs:sequence>
    <xs:element ref="wsrm:Identifier"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="TerminateSequenceResponseType">
  <xs:sequence>
    <xs:element ref="wsrm:Identifier"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="AcksTo" type="wsa:EndpointReferenceType"/>
<xs:complexType name="OfferType">
  <xs:sequence>
    <xs:element ref="wsrm:Identifier"/>
    <xs:element ref="wsrm:Expires" minOccurs="0"/>
    <xs:element name="EndpointReference" type="wsa:EndpointReferenceType"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="AcceptType">
  <xs:sequence>
    <xs:element ref="wsrm:AcksTo" type="wsa:EndpointReferenceType"/>
    <xs:any namespace="##other" processContents="lax" minOccurs="0" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<xs:element name="Expires">
  <xs:complexType>
    <xs:simpleContent>
      <xs:extension base="xs:duration">
        <xs:anyAttribute namespace="##other" processContents="lax"/>
      </xs:extension>
    </xs:simpleContent>
  </xs:complexType>
</xs:element>
<xs:simpleType name="IncompleteSequenceBehaviorType">
  <xs:restriction base="xs:string">
    <xs:enumeration value="DiscardEntireSequence"/>
    <xs:enumeration value="DiscardFollowingFirstGap"/>
    <xs:enumeration value="NoDiscard"/>
  </xs:restriction>
</xs:simpleType>
<xs:element name="UnsupportedElement">
  <xs:simpleType>
    <xs:restriction base="xs:QName"/>
  </xs:complexType>
</xs:element>
B. WSDL

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

http://docs.oasis-open.org/ws-rx/wsrm/200604/wsdll/wsrm-1.1-wsdl-200604.wsdl

The following non-normative copy is provided for reference.
<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/
xmlns:xs="http://www.w3.org/2001/XMLSchema
targetNamespace="http://docs.oasis-open.org/ws-rx/wsrm/200604/wsdl">

<wsdl:types>
  <xs:schema>
    <xs:import namespace="http://docs.oasis-open.org/ws-rx/wsrm/200604"
                schemaLocation="http://docs.oasis-open.org/ws-rx/wsrm/200604/wsdl-1.1-schema-200604.xsd"/>

    <wsdl:message name="CreateSequence">
      <wsdl:part name="create" element="rm:CreateSequence"/>
    </wsdl:message>

    <wsdl:message name="CreateSequenceResponse">
      <wsdl:part name="createResponse" element="rm:CreateSequenceResponse"/>
    </wsdl:message>

    <wsdl:message name="CloseSequence">
      <wsdl:part name="close" element="rm:CloseSequence"/>
    </wsdl:message>

    <wsdl:message name="CloseSequenceResponse">
      <wsdl:part name="closeResponse" element="rm:CloseSequenceResponse"/>
    </wsdl:message>
  </xs:schema>
</wsdl:types>
</wsdl:definitions>
<wsdl:message name="TerminateSequence">
  <wsdl:part name="terminate" element="rm:TerminateSequence"/>
</wsdl:message>

<wsdl:message name="TerminateSequenceResponse">
  <wsdl:part name="terminateResponse" element="rm:TerminateSequenceResponse"/>
</wsdl:message>

<wsdl:message name="MakeConnection">
  <wsdl:part name="makeConnection" element="rm:MakeConnection"/>
</wsdl:message>

<wsdl:portType name="SequenceAbstractPortType">
  <wsdl:operation name="CreateSequence">
    <wsdl:input message="tns:CreateSequence" wsa:Action="http://docs.oasis-open.org/ws-rx/wsrmi/200604/CreateSequence"/>
  </wsdl:operation>

  <wsdl:operation name="CloseSequence">
  </wsdl:operation>

  <wsdl:operation name="TerminateSequence">
  </wsdl:operation>

  <wsdl:operation name="MakeConnection">
    <wsdl:input message="tns:MakeConnection" wsa:Action="http://docs.oasis-open.org/ws-rx/wsrmi/200604/MakeConnection"/>
  </wsdl:operation>
</wsdl:portType>
C. Message Examples

C.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:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/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:ReplyTo>
    </wsa:ReplyTo>
  </S:Header>
  <S:Body>
    <wsrm:CreateSequence>
      <wsrm:AcksTo>
      </wsrm:AcksTo>
    </wsrm:CreateSequence>
  </S:Body>
</S:Envelope>
```

Create Sequence Response

```
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:To>http://Business456.com/serviceA/789</wsa:To>
    <wsa:RelatesTo>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546817
    </wsa:RelatesTo>
  </S:Header>
  <S:Body>
    <wsrm:CreateSequenceResponse>
    </wsrm:CreateSequenceResponse>
  </S:Body>
</S:Envelope>
```

C.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:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>http://Business456.com/guid/71e0654e-5ce8-477b-bb9d-34f05cfcb9e</wsa:MessageID>
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <wsa:From>
    </wsa:From>
    <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
    <wsrm:Sequence>
      <wsrm:MessageNumber>1</wsrm:MessageNumber>
    </wsrm: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:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/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:From>
    <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
    <wsrm:Sequence>
      <wsrm:MessageNumber>2</wsrm:MessageNumber>
    </wsrm: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:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546819</wsa:MessageID>
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <wsa:From>
    </wsa:From>
    <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
    <wsrm:Sequence>
    </wsrm:Sequence>
  </S:Header>
  <S:Body>
    <!-- Some Application Data -->
  </S:Body>
</S:Envelope>
C.3 First Acknowledgement

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

```xml
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
            xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
            xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546810</wsa:MessageID>
    <wsa:To>http://Business456.com/serviceA/789</wsa:To>
    <wsa:From><wsa:Address>http://example.com/serviceB/123</wsa:Address></wsa:From>
  </S:Header>
  <S:Body/>
</S:Envelope>
```

C.4 Retransmission

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

```xml
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
            xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
            xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>http://Business456.com/guid/daa7d0b2-c8e0-476e-a9a4-d164154e38de</wsa:MessageID>
    <wsa:To>http://example.com/serviceA/789</wsa:To>
    <wsa:Action>http://example.com/serviceB/123/request</wsa:Action>
  </S:Header>
  <S:Body/>  
</S:Envelope>
```
C.5 Termination

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

```xml
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://example.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546811
    </wsa:MessageID>
    <wsa:To>http://Business456.com/serviceA/789</wsa:To>
    <wsa:From>
      <wsa:Address>http://example.com/serviceB/123</wsa:Address>
    </wsa:From>
    <wsa:Action>
      http://docs.oasis-open.org/ws-rx/wsrm/200604/SequenceAcknowledgement
    </wsa:Action>
    <wsrm:SequenceAcknowledgement>
      <wsrm:AcknowledgementRange Upper="3" Lower="1"/>
    </wsrm:SequenceAcknowledgement>
  </S:Header>
  <S:Body/>
</S:Envelope>
```

Terminate Sequence

```xml
<?xml version="1.0" encoding="UTF-8"?>
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
    </wsa:MessageID>
    <wsa:To>http://example.com/serviceB/123</wsa:To>
    <wsa:From>
    </wsa:From>
    <wsa:Action>
      http://docs.oasis-open.org/ws-rx/wsrm/200604/TerminateSequence
    </wsa:Action>
  </S:Header>
  <S:Body>
    <wsrm:TerminateSequence>
    </wsrm:TerminateSequence>
  </S:Body>
</S:Envelope>
```

Terminate Sequence Response
<S:Envelope xmlns:S="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsrm="http://docs.oasis-open.org/ws-rx/wsrm/200604"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
  <S:Header>
    <wsa:MessageID>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546813
    </wsa:MessageID>
    <wsa:To>http://example.com/serviceA/789</wsa:To>
    <wsa:Action>
      http://docs.oasis-open.org/ws-rx/wsrm/200604/TerminateSequenceResponse
    </wsa:Action>
    <wsa:RelatesTo>
      http://Business456.com/guid/0baaf88d-483b-4ecf-a6d8-a7c2eb546812
    </wsa:RelatesTo>
    <wsa:From>
      http://Business456.com/serviceA/789
    </wsa:From>
  </S:Header>
  <S:Body>
    <wsrm:TerminateSequenceResponse>
    </wsrm:TerminateSequenceResponse>
  </S:Body>
</S:Envelope>
D. State Tables

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

Each cell in the tables in this appendix uses the following convention:

| Legend | action to take | next state |

<table>
<thead>
<tr>
<th>Events</th>
<th>States</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>Connecting</td>
</tr>
<tr>
<td>Create Sequence</td>
<td>Transmit Create Sequence</td>
</tr>
<tr>
<td>Create Sequence Response</td>
<td>N/A</td>
</tr>
<tr>
<td>Create Sequence Refused Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>New Message</td>
<td>N/A</td>
</tr>
<tr>
<td>Retransmit of unack message</td>
<td>N/A</td>
</tr>
<tr>
<td>SeqAck (non-final)</td>
<td>N/A</td>
</tr>
<tr>
<td>Nack</td>
<td>N/A</td>
</tr>
<tr>
<td>Reached max msg number</td>
<td>N/A</td>
</tr>
<tr>
<td>Events</td>
<td>States</td>
</tr>
<tr>
<td>--------------------------------</td>
<td>---------------------------------------------</td>
</tr>
<tr>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Message Number Rollover Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>Close Sequence</td>
<td>N/A</td>
</tr>
<tr>
<td>Close Sequence Response</td>
<td>N/A</td>
</tr>
<tr>
<td>SeqAck (final)</td>
<td>N/A</td>
</tr>
<tr>
<td>Sequence Closed Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>Unknown Sequence Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>Sequence Terminated Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>Terminate Sequence</td>
<td>N/A</td>
</tr>
<tr>
<td>Terminate Sequence Response</td>
<td>N/A</td>
</tr>
<tr>
<td>Elapse Expires duration</td>
<td>N/A</td>
</tr>
</tbody>
</table>

In Table 2 above, the rows consists of events that occur at the RM Source throughout the lifetime of an RM Sequence and the columns consists of various RM Source states. Each cell in the table above lists
the action that the RM Source takes on occurrence of a particular event and the next state that it transitions.

### Table 3 RM Destination State Transition Table

<table>
<thead>
<tr>
<th>Events</th>
<th>States</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>None</td>
</tr>
<tr>
<td>Creation request not satisfied</td>
<td>N/A</td>
</tr>
<tr>
<td>Message (with message number within range)</td>
<td>N/A</td>
</tr>
<tr>
<td>Ack requested</td>
<td>N/A</td>
</tr>
<tr>
<td>Message (with message number outside of range)</td>
<td>N/A</td>
</tr>
<tr>
<td>Close Sequence</td>
<td>N/A</td>
</tr>
<tr>
<td>Close Sequence itself</td>
<td>N/A</td>
</tr>
<tr>
<td>Terminate Sequence</td>
<td>N/A</td>
</tr>
<tr>
<td>Unknown Sequence Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>Sequence Terminated Fault</td>
<td>N/A</td>
</tr>
<tr>
<td>Elapse Expires duration</td>
<td>N/A</td>
</tr>
</tbody>
</table>
In Table 3 above, the rows consists of events that occur at the RM Destination throughout the lifetime of an RM Sequence and the columns consists of various RM Destination states. Each cell in the table above lists the action that the RM Destination takes on occurrence of a particular event and the next state that it transitions.
E. Acknowledgments

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

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

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

Keith Ballinger, Microsoft, Stefan Batres, Microsoft, Rebecca Bergersen, Iona, 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, David Ingham, 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:

## F. Revision History

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>By Whom</th>
<th>What</th>
</tr>
</thead>
<tbody>
<tr>
<td>wd-01</td>
<td>2005-07-07</td>
<td>Christopher Ferris</td>
<td>Initial version created based on submission by the authors.</td>
</tr>
<tr>
<td>ws-02</td>
<td>2005-07-21</td>
<td>Doug Davis</td>
<td>i011 (PT0S) added</td>
</tr>
<tr>
<td>wd-02</td>
<td>2005-08-16</td>
<td>Anish Karmarkar</td>
<td>Trivial editorial changes</td>
</tr>
<tr>
<td>ws-03</td>
<td>2005-09-15</td>
<td>Doug Davis</td>
<td>i019 and i028 (CloseSeq) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-26</td>
<td>Gilbert Pilz</td>
<td>i005 (Source resend of nacks messages when ack already received) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i027 (InOrder delivery assurance spanning multiple sequences) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i020 (Semantics of &quot;At most once&quot; Delivery Assurance) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i034 (Fault while processing a piggy-backed RM header) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i033 (Processing model of NACKs) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i031 (AckRequested schema inconsistency) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i025 (SeqAck/None) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i029 (Remove dependency on WS-Security) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i039 (What does 'have a mU attribute' mean) added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-27</td>
<td>Doug Davis</td>
<td>i040 (Change 'optional'/'required' to 'OPTIONAL'/'REQUIRED') added</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-30</td>
<td>Anish Karmarkar</td>
<td>i017 (Change NS to <a href="http://docs.oasis-open.org/wsr%D0%BC/200510/">http://docs.oasis-open.org/wsrм/200510/</a>)</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-30</td>
<td>Anish Karmarkar</td>
<td>i045 (Include SecureConversation as a reference and move it to non-normative citation)</td>
</tr>
<tr>
<td>wd-05</td>
<td>2005-09-30</td>
<td>Anish Karmarkar</td>
<td>i046 (change the type of wsrм:FaultCode element)</td>
</tr>
<tr>
<td>wd-06</td>
<td>2005-11-02</td>
<td>Gilbert Pilz</td>
<td>Start wd-06 by changing title page from cd-01.</td>
</tr>
<tr>
<td>wd-06</td>
<td>2005-11-03</td>
<td>Gilbert Pilz</td>
<td>i047 (Reorder spec sections)</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-17</td>
<td>Gilbert Pilz</td>
<td>Start wd-07</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i071 – except for period in Appendix headings</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i10</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i030</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i037</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i038</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i041</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i043</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i044</td>
</tr>
<tr>
<td>Rev</td>
<td>Date</td>
<td>By Whom</td>
<td>What</td>
</tr>
<tr>
<td>-------</td>
<td>------------</td>
<td>-------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i048</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i051</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i053</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i059</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i062</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i063</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i065</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i067</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i068</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>i069</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-28</td>
<td>Doug Davis</td>
<td>Fix bulleted list (#2) in section 2.3</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-29</td>
<td>Gilbert Pilz</td>
<td>i074 (Use of [tcShortName] in artifact locations namespaces, etc)</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-29</td>
<td>Gilbert Pilz</td>
<td>i071 – Fixed styles and formatting for TOC. Fixed styles of the appendix headings.</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-30</td>
<td>Doug Davis</td>
<td>Removed dup definition of &quot;Receive&quot;</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-11-30</td>
<td>Gilbert Pilz</td>
<td>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.</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-12-01</td>
<td>Gilbert Pilz</td>
<td>Updated fix for i074 to eliminate trailing '/'. Added corresponding text around action IRI composition.</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-12-01</td>
<td>Gilbert Pilz</td>
<td>Use non-fixed fields for date values on both title page and body footers.</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-12-01</td>
<td>Doug Davis</td>
<td>Alphabetize the glossary</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-12-02</td>
<td>Doug Davis</td>
<td>i064</td>
</tr>
<tr>
<td>wd-07</td>
<td>2005-12-02</td>
<td>Doug Davis</td>
<td>i066</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-15</td>
<td>Doug Davis</td>
<td>Add back in RM Source to glossary</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-15</td>
<td>Steve Winkler</td>
<td>Doug added Steve's editorial nits</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Doug Davis</td>
<td>i050</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Doug Davis</td>
<td>i081</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Doug Davis</td>
<td>i080 – but i050 negates the need for any changes</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Doug Davis</td>
<td>i079</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Doug Davis</td>
<td>I076 – didn't add text about &quot;replies&quot; since the RMD to RMS sequence could be used for any message not just replies</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Umit Yalcinalp</td>
<td>Action Su03: removed wsse from Table 1</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-21</td>
<td>Umit Yalcinalp</td>
<td>I057 per Sunnyvale F2F 2005. Cleaned up some formatting errors in contributors</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-27</td>
<td>Doug Davis</td>
<td>i060</td>
</tr>
<tr>
<td>Rev</td>
<td>Date</td>
<td>By Whom</td>
<td>What</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>---------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-27</td>
<td>Gilbert Pilz</td>
<td>Moved schema and WSDL files to their own artifacts. Converted source document to OpenDocument Text format. Changed line numbers to be a single style.</td>
</tr>
<tr>
<td>wd-08</td>
<td>2005-12-28</td>
<td>Anish Karmarkar</td>
<td>Included a section link to c:\temp\wsrm-1.1-schema-200510.xsd and to c:\temp\wsrm-1.1-wsdl-200510.wsdl</td>
</tr>
<tr>
<td>wd-08</td>
<td>2006-01-04</td>
<td>Gilbert Pilz</td>
<td>Fixed formatting for included sections.</td>
</tr>
<tr>
<td>wd-08</td>
<td>2006-01-05</td>
<td>Gilbert Pilz</td>
<td>Created links for unused references. Fixed exemplars for CloseSequence and CloseSequenceResponse.</td>
</tr>
<tr>
<td>wd-09</td>
<td>2006-01-11</td>
<td>Doug Davis</td>
<td>Minor tweaks to text/typos.</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-01-23</td>
<td>Doug Davis</td>
<td>Accept all changes from wd-09 Make some minor editorial tweaks from Marc's comments.</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-14</td>
<td>Doug Davis</td>
<td>Issue 082 resolution</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-14</td>
<td>Doug Davis</td>
<td>Issue 083 resolution</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-14</td>
<td>Doug Davis</td>
<td>Issue 085 resolution</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-14</td>
<td>Doug Davis</td>
<td>Issues 086, 087 resolutions Defined MessageNumberType</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-15</td>
<td>Doug Davis</td>
<td>Issue 078 resolution</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-15</td>
<td>Doug Davis</td>
<td>Issue 094 resolution</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-15</td>
<td>Doug Davis</td>
<td>Issue 095 resolution</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-15</td>
<td>Gilbert Pilz</td>
<td>Issue 088 – added namespace URI link to namespace URI; added text explaining that this URI could be dereferenced to produce the RDDL doc; added non-normative reference to RDDL 2.0</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-17</td>
<td>Anish Karmarkar</td>
<td>Namespace changed to 200602 for both WSDL and XSD docs.</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-17</td>
<td>Anish Karmarkar</td>
<td>Issue i087 as it applies to WSRM spec.</td>
</tr>
<tr>
<td>wd-10</td>
<td>2006-02-17</td>
<td>Anish Karmarkar</td>
<td>Added titles and minor text for state table (issue i058).</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-02-22</td>
<td>Doug Davis</td>
<td>Accept all changes for new WD</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-02-23</td>
<td>Doug Davis</td>
<td>s&quot;close&quot;/close/g – per Marc Goodner</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-02-27</td>
<td>Doug Davis</td>
<td>Added first ref to [URI] – per Marc G again</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-02-27</td>
<td>Doug Davis</td>
<td>Issue i061 applied</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-02-28</td>
<td>Doug Davis</td>
<td>Fixed typo around the use of &quot;above&quot; and &quot;below&quot;</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-03-01</td>
<td>Doug Davis</td>
<td>Minor typos found by Marc Goodner</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-03-02</td>
<td>Doug Davis</td>
<td>Minor typos found by Matt Lovett</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-03-08</td>
<td>Doug Davis</td>
<td>Issue 091 applied</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-03-08</td>
<td>Doug Davis</td>
<td>Issue 092 applied</td>
</tr>
<tr>
<td>Rev</td>
<td>Date</td>
<td>By Whom</td>
<td>What</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>wd-11</td>
<td>2006-03-08</td>
<td>Doug Davis</td>
<td>Issue 100 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-03-20</td>
<td>Doug Davis</td>
<td>Added space in &quot;SOAP1.x&quot; – Paul Cotton</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 007 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 090 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 098 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 099 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 101 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 103 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 104 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 105 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 107 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 109 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-11</td>
<td>Doug Davis</td>
<td>Issue 110 applied</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-12</td>
<td>Doug Davis</td>
<td>Used &quot;generated&quot; instead of &quot;issue&quot; or &quot;send&quot; when talking about faults.</td>
</tr>
<tr>
<td>wd-12</td>
<td>2006-04-24</td>
<td>Gilbert Pilz</td>
<td>Update references to WS-Addressing to the Proposed Recommendations; update WS-RM namespace to “200604”.</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-08</td>
<td>Gilbert Pilz</td>
<td>i093 part 1; more work needed</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-10</td>
<td>Doug Davis</td>
<td>Issue 096 applied</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-26</td>
<td>Gilbert Pilz</td>
<td>i093 part 2; reflects decisions from 2006-05-25 meeting</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-28</td>
<td>Gilbert Pilz</td>
<td>Issue 106 applied</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-29</td>
<td>Gilbert Pilz</td>
<td>Issue 118 applied</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-29</td>
<td>Gilbert Pilz</td>
<td>Issue 120 applied</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-30</td>
<td>Gilbert Pilz</td>
<td>Issue 114 applied</td>
</tr>
<tr>
<td>wd-13</td>
<td>2006-05-30</td>
<td>Gilbert Pilz</td>
<td>Issue 116 applied</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-05</td>
<td>Gilbert Pilz</td>
<td>Accept all changes; bump WD number</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Applied lots of minor edits from Marc Goodner</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Change a couple of period/sp/sp to period/sp</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Added a space in &quot;URI]of&quot; – per Marc Goodner</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Issue 131 applied</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Issue 132 applied</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Issue 119 applied</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>Applied lots of minor edits from Doug Davis</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-07</td>
<td>Doug Davis</td>
<td>s/&quot;none&quot;/&quot;full-uri&quot;/ - per Marc Goodner</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-12</td>
<td>Doug Davis</td>
<td>Complete i106</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-12</td>
<td>Doug Davis</td>
<td>Issues 089 applied</td>
</tr>
<tr>
<td>wd-14</td>
<td>2006-06-12</td>
<td>Doug Davis</td>
<td>Fix for several RFC2119 keywords – per Anish</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-12</td>
<td>Doug Davis</td>
<td>Accept all changed, dump WD number</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-12</td>
<td>Doug Davis</td>
<td>Move WSDL after Schema</td>
</tr>
<tr>
<td>Rev</td>
<td>Date</td>
<td>By Whom</td>
<td>What</td>
</tr>
<tr>
<td>------</td>
<td>------------</td>
<td>-------------</td>
<td>----------------------------------------------------------------------</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-12</td>
<td>Doug Davis</td>
<td>Nits – remove tabs, extra [yyy]'s ...</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-14</td>
<td>Doug Davis</td>
<td>Remove extra &quot;OPTIONAL&quot;s – Matt Lovett</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-14</td>
<td>Doug Davis</td>
<td>Remove blank rows/columns from state table. Fix italics in state table</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-15</td>
<td>Doug Davis</td>
<td>Typo – section D was empty</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 125 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 126 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 127 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 133 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 136 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 138 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-16</td>
<td>Doug Davis</td>
<td>Issue 135 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-20</td>
<td>Doug Davis</td>
<td>Added all TC members to the ack list</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-22</td>
<td>Doug Davis</td>
<td>Issue 129 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-22</td>
<td>Doug Davis</td>
<td>Issue 130 applied</td>
</tr>
<tr>
<td>wd-15</td>
<td>2006-06-22</td>
<td>Doug Davis</td>
<td>Issue 137 applied</td>
</tr>
</tbody>
</table>
G. 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.

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.