WSRM – Section 2:

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 enables a Reliable Messaging (RM) Source to accurately determine the disposition of each message it Transmits as perceived by the RM Destination, so as to allow it to resolve any in-doubt status regarding receipt of the message Transmitted. The protocol also enables an RM Destination to efficiently determine which of those messages it Receives have been previously Received, enabling it to filter out duplicate message transmissions caused by the retransmission, by the RM Source, of an unacknowledged message. It also enables an RM Destination to Deliver the messages it Receives to the Application Destination in the order in which they were sent by an Application Source, in the event that they are Received out of order. 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 accepting the message, the RM Destination Acknowledges it. Finally, the RM Destination Delivers the message to the Application Destination. The exact roles the entities play and the complete meaning of the events will be defined throughout this specification.

Figure 1: Reliable Messaging Model

2.1. Glossary

The following definitions are used throughout this specification:

Accept: The act of qualifying a message by the RM Destination such that it becomes eligible for Delivery and acknowledgement**.**

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

Acknowledgement Message: A message containing a `SequenceAcknowledgement` header block. Acknowledgement Messages may or may not contain a SOAP body.

Acknowledgement Request: A message containing an `AckRequested` header. Acknowledgement Requests may or may not contain a SOAP body.

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

Application Source: The Endpoint that Sends a message.

Back-channel: When the underlying transport provides a mechanism to return a transport-protocol specific response, capable of carrying a SOAP message, without initiating a new connection, this specification refers to this mechanism as a back-channel.

Deliver: The act of transferring responsibility for 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 (EPRs) convey the information needed to address a Web service Endpoint.

Receive: The act of reading a message from a network connection and accepting it.

RM Destination: The Endpoint that Receives messages Transmitted reliably from an RM Source.

RM Protocol Header Block: One of `Sequence`, `SequenceAcknowledgement`, or `AckRequested`.

RM Source: The Endpoint that Transmits messages reliably to an RM Destination.

Send: The act of transferring a message from the Application Source to the RM Source for reliable transfer.

Sequence Lifecycle Message: A message that contains one of: `CreateSequence`, `CreateSequenceResponse`, `CloseSequence`, `CloseSequenceResponse`, `TerminateSequence`, `TerminateSequenceResponse` as the child element of the SOAP body element.

Sequence Traffic Message: A message containing a `Sequence` header block.

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:

- For any single message exchange the RM Source MUST have an endpoint reference that uniquely identifies the RM Destination Endpoint.
- The RM Source MUST have successfully created a Sequence with the RM Destination.
- The RM Source MUST be capable of formulating messages that adhere to the RM Destination's policies.
- 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, the following invariants are REQUIRED for correctness:

- 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.
- Within every Acknowledgement Message it issues, the RM Destination MUST include one or more `AcknowledgementRange` child elements that contain, in their collective ranges, the message number of every message accepted by the RM Destination. The RM Destination MUST exclude, in the `AcknowledgementRange` elements, the message numbers of any messages it has not accepted. If no messages have been received the RM Destination MUST return `None` instead of an `AcknowledgementRange(s)`. The RM Destination MAY transmit a `Nack` for a specific message or messages in stead of an `AcknowledgementRange(s)`.
- While the Sequence is not closed or terminated, the RM Source SHOULD retransmit unacknowledged messages.

2.4 Delivery Assurances

This section defines a number of Delivery Assurance assertions, which can be supported by RM Sources and RM Destinations. These assertions can be specified as policy assertions using the WS-Policy framework [WS-Policy]. For details on this see the WS-RM Policy specification [WS-RM Policy]

AtLeastOnce

Each message is to be delivered at least once, or else an error MUST be raised by the RM Source and/or RM Destination. The requirement on an RM Source is that it SHOULD retry transmission of every message sent by the Application Source until it receives an acknowledgement from the RM Destination. The requirement on the RM Destination is that it SHOULD retry the transfer to the Application Destination of any message that it accepts from the RM Source, until that message has been successfully delivered. There is no requirement for the RM Destination to apply duplicate message filtering.

AtMostOnce
Each message is to be delivered at most once. The RM Source MAY retry transmission of unacknowledged messages, but is NOT REQUIRED to do so. The requirement on the RM Destination is that it MUST filter out duplicate messages, i.e. that it MUST NOT deliver a duplicate of a message that has already been delivered.

ExactlyOnce
Each message is to be delivered exactly once; if a message cannot be delivered then an error MUST be raised by the RM Source and/or RM Destination. The requirement on an RM Source is that it SHOULD retry transmission of every message sent by the Application Source until it receives an acknowledgement from the RM Destination. The requirement on the RM Destination is that it SHOULD retry the transfer to the Application Destination of any message that it accepts from the RM Source until that message has been successfully delivered, and that it MUST NOT deliver a duplicate of a message that has already been delivered.

InOrder
Messages from each individual sequence are to be delivered in the same order they have been sent by the Application Source. The requirement on an RM Source is that it MUST ensure that the ordinal position of each message in the sequence (as indicated by a message sequence number) is consistent with the order in which the messages have been sent from the Application Source. The requirement on the RM Destination is that it MUST deliver received messages for each sequence in the order indicated by the message numbering. This DeliveryAssurance can be used in combination with any of the AtLeastOnce, AtMostOnce or ExactlyOnce assertions, and the requirements of those assertions MUST also be met. In particular if the AtLeastOnce or ExactlyOnce assertion applies and the RM Destination detects a gap in the sequence then the RM Destination MUST NOT deliver any subsequent messages from that sequence until the missing messages are received or until the sequence is closed.

2.5 Example Message Exchange

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

Figure 2: The WS-ReliableMessaging Protocol

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. The TerminateSequence message indicates that message number 3 was the last message in the Sequence. The RM Destination then 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.

WS-RMP – Section 2:

2. RM Policy Assertions

WS-Policy Framework and WS-Policy Attachment [WS-PolicyAttachment] collectively define a framework, model and grammar for expressing the requirements, and general characteristics of entities in an XML Web services-based system. To enable an RM Destination and an RM Source to describe their requirements for a given Sequence, this specification defines a single RM policy assertion that leverages the WS-Policy framework.

2.1. Assertion Model

The RM policy assertion indicates that the RM Source and RM Destination MUST use WS-ReliableMessaging to ensure reliable delivery of messages. Specifically, the WS-ReliableMessaging protocol determines invariants maintained by the reliable messaging endpoints and the directives used to track and manage the delivery of a Sequence of messages.

2.2. Normative Outline

The normative outline for the RM assertion is:

```
<wsrmp:RMAssertion [wsp:Optional="true"]? ... >

<wsp:Policy>

[ <wsrmp:SequenceSTR/> |

<wsrmp:SequenceTransportSecurity/> ] ?

 <wsrmp:DeliveryAssurance>

    <wsp:Policy>

    [ <wsrmp:ExactlyOnce/> |

      <wsrmp:AtLeastOnce/> |

      <wsrmp:AtMostOnce> ]

      <wsrmp:InOrder/> ?

    </wsp:Policy>
```

```
   </wsrmp:DeliveryAssurance> ?
</wsp:Policy>
...
</wsrmp:RMAssertion>
```

The following describes the content model of the `RMAssertion` element.

/wsrmp:RMAssertion

A policy assertion that specifies that WS-ReliableMessaging protocol MUST be used when sending messages.

/wsrmp:RMAssertion/@wsp:Optional="true"

Per WS-Policy, this is compact notation for two policy alternatives, one with and one without the assertion. The intuition is that the behavior indicated by the assertion is optional, or in this case, that WS-ReliableMessaging MAY be used.

/wsrmp:RMAssertion/wsp:Policy

This required element allows for the inclusion of nested policy assertions.

/wsrmp:RMAssertion/wsp:Policy/wsrmp:SequenceSTR

When present, this assertion defines the requirement that an RM Sequence MUST be bound to an explicit token that is referenced from a wsse:SecurityTokenReference in the CreateSequence message. See section 2.5.1.

/wsrmp:RMAssertion/wsp:Policy/wsrmp:SequenceTransportSecurity

When present, this assertion defines the requirement that an RM Sequence MUST be bound to the session(s) of the underlying transport-level protocol used to carry the `CreateSequence` and `CreateSequenceResponse` message. See section 2.5.2.

/wsrmp:RMAssertion/wsp:Policy/wsrmp:DeliveryAssurance

This expression, which may be omitted, describes the message delivery quality of service between the RM and application layer. When used by an RM Destination it expresses the delivery assurance in effect between the RM Destination and its corresponding application destination, and it also indicates requirements on any RM Source that transmits messages to this RM destination. Conversely when used by an RM Source it expresses the delivery assurance in effect between the RM Source and its corresponding application source, as well as indicating requirements on any RM Destination that receives messages from this RM Source. In either case the delivery assurance does not affect the messages transmitted on the wire. Absence of this expression from a wsrmp:RMAssertion policy assertion simply means that the endpoint has chosen not to advertise its delivery assurance characteristics.

/wsrmp:RMAssertion/wsp:Policy/wsrmp:DeliveryAssurance/wsp:Policy

This required element identifies additional requirements for the use of the wsrmp:DeliveryAssurance.

/wsrmp:RMAssertion/wsp:Policy/wsrmp:DeliveryAssurance/wsp:Policy/wsrmp:ExactlyOnce

This expresses the ExactlyOnce Delivery Assurance defined in [WS-RM].

/wsrmp:RMAssertion/wsp:Policy/wsrmp:DeliveryAssurance/wsp:Policy/wsrmp:AtLeastOnce

This expresses the AtLeastOnce Delivery Assurance defined in [WS-RM].

/wsrmp:RMAssertion/wsp:Policy/wsrmp:DeliveryAssurance/wsp:Policy/wsrmp:AtMostOnce

This expresses the AtMostOnce Delivery Assurance defined in [WS-RM].

/wsrmp:RMAssertion/wsp:Policy/wsrmp:DeliveryAssurance/wsp:Policy/wsrmp:InOrder

This expresses the InOrder Delivery Assurance defined in [WS-RM].

/wsrmp:RMAssertion/{any}

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

/wsrmp:RMAssertion/@{any}

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

2.3. Assertion Attachment

The RM policy assertion is allowed to have the following Policy Subjects [WS-PolicyAttachment]:

- Endpoint Policy Subject
- Message Policy Subject

WS-PolicyAttachment defines a set of WSDL/1.1 policy attachment points for each of the above Policy Subjects. Since an RM policy assertion specifies a concrete behavior, it MUST NOT be attached to the abstract WSDL policy attachment points.

The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects allowed for an RM policy assertion but which MUST NOT have RM policy assertions attached:

- wsdl:message
- wsdl:portType/wsdl:operation/wsdl:input
- wsdl:portType/wsdl:operation/wsdl:output
- wsdl:portType/wsdl:operation/wsdl:fault
- wsdl:portType

The following is the list of WSDL/1.1 elements whose scope contains the Policy Subjects allowed for an RM policy assertion and which MAY have RM policy assertions attached:

- wsdl:port
- wsdl:binding
- wsdl:binding/wsdl:operation/wsdl:input
- wsdl:binding/wsdl:operation/wsdl:output
- wsdl:binding/wsdl:operation/wsdl:fault

If an RM policy assertion is attached to any of:

- wsdl:binding/wsdl:operation/wsdl:input
- wsdl:binding/wsdl:operation/wsdl:output
- wsdl:binding/wsdl:operation/wsdl:fault

then an RM policy assertion, specifying wsp:Optional=true MUST be attached to the corresponding wsdl:binding or wsdl:port, indicating that the endpoint supports WS-RM. Any messages, regardless of whether they have an attached Message Policy Subject RM policy assertion, MAY be sent to that endpoint using WS-RM. Additionally, the receiving endpoint MUST NOT reject any message belonging to a Sequence, simply because there was no Message Policy Subject RM policy assertion attached to that message. There might be certain RM implementations that are incapable of applying RM Quality of Service (QoS) semantics on a per-message basis. In order to ensure the broadest interoperability, when an endpoint decorates its WSDL with RM policy assertions using Message Policy Subject, it MUST also be prepared to accept that all messages sent to that endpoint might be sent within the context of an RM Sequence, regardless of whether the corresponding wsdl:input, wsdl:output or wsdl:fault had an attached RM policy assertion.

Rather than turn away messages that were unnecessarily sent with RM semantics, the receiving endpoint described by the WSDL MUST accept these messages.

By attaching an RM policy assertion that specifies wsp:Optional="true" to the corresponding endpoint that has attached RM policy assertions at the Message Policy Subject level, the endpoint is describing the above constraint in policy.

In the case where an optional RM Assertion applies to an output message, there is no requirement on the client to support an RM Destination implementation

2.4. Assertion Example

Table 2 lists an example use of the RM policy assertion.

Table 2: Example policy with RM policy assertion

```
(01)<wsdl:definitions

(02) targetNamespace="example.com"

(03) xmlns:tns="example.com"

(04) xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"

(05) xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy"

(06) xmlns:wsrmp="http://docs.oasis-open.org/ws-rx/wsrmp/200608"

(07) xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
200401-wss-wssecurity-utility-1.0.xsd">

(08)

(09) <wsp:UsingPolicy wsdl:required="true" />

(10)

(11) <wsp:Policy wsu:Id="MyPolicy" >

(12) <wsrmp:RMAssertion>

(13) <wsp:Policy/>

(14) </wsrmp:RMAssertion>

(15) <!-- omitted assertions -->

(16) </wsp:Policy>

(17)
```

```
(18) <!-- omitted elements -->

(19)

(20) <wsdl:binding name="MyBinding" type="tns:MyPortType" >

(21) <wsp:PolicyReference URI="#MyPolicy" />

(22) <!-- omitted elements -->

(23) </wsdl:binding>

(24)

(25)</wsdl:definitions>
```

Line (09) in Table 2 indicates that WS-Policy is in use as a required extension.

Lines (11-16) are a policy expression that includes a RM policy assertion (lines 12-14) to indicate that WS-ReliableMessaging must be used.

Lines (20-23) are a WSDL binding. Line (21) indicates that the policy in lines (11-16) applies to this binding, specifically indicating that WS-ReliableMessaging must be used over all the messages in the binding.

2.5. Sequence Security Policy

WS-SecurityPolicy [SecurityPolicy] provides a framework and grammar for expressing the security requirements and characteristics of entities in a XML web services based system. The following assertions MAY be used in conjunction with WS-SecurityPolicy to express additional security requirements particular to RM Sequences.

2.5.1. RM Assertion with Sequence STR Assertion

This version of the RM assertion includes the requirement that an RM Sequence MUST be bound to an explicit token that is referenced from a `wsse:SecurityTokenReference` in the `CreateSequence` message.

This assertion MUST apply to [Endpoint Policy Subject]. The normative outline for this form of the Sequence STR Assertion is:

```
<wsrmp:RMAssertion [wsp:Optional="true"]? ...>

<wsp:Policy>

<wsrmp:SequenceSTR/>
```

```
<wsp:Policy>

</wsrmp:RMAssertion>
```

The following describes the content model of the SequenceSTR element.

/wsrmp:SequenceSTR

A policy assertion that specifies security requirements which MUST be used with an RM Sequence that are particular to WS-RM and beyond what can be expressed in WS-SecurityPolicy.

2.5.2. RM Assertion with Sequence Transport Security Assertion

This version of the RM assertion includes the requirement that an RM Sequence MUST be bound to the session(s) of the underlying transport-level security protocol (e.g. SSL/TLS) used to carry the CreateSequence and CreateSequenceResponse messages.

This assertion MUST apply to [Endpoint Policy Subject]. This assertion is effectively meaningless unless it occurs in conjunction with the sp:TransportBinding assertion that requires the use of some transport-level security mechanism (e.g. sp:HttpsToken).

The normative outline for this form of the RM Assertion with the Sequence Transport Security Assertion is:

```
<wsp:Policy>

<wsp:>ExactlyOne>

<wsp:All>

<wsrm:RMAssertion [wsp:Optional="true"]> ...>

<wsp:Policy>

<wsrmp:SequenceTransportSecurity/>

</wsp:Policy>

</wsrm:RMAssertion>

<sp:TransportBinding ...>

...

</sp:TransportBinding>
```

```
<wsp:All>

<wsp:ExactlyOne>

</wsp:Policy>
```

The following describes the content model of the `SequenceTransportSecurity` element.

/wsrmp:SequenceTransportSecurity

A policy assertion that specifies that any Sequences targeted to the indicated endpoint MUST be bound to the underlying session(s) of the transport-level security used to carry messages related to the Sequence.

This form of the RM Assertion says that an endpoint MAY have RM as an option but always requires HTTPS to be used. All the SequenceTransportSecurity assertion indicates is that RM's rules for protecting the Sequence over TLS are followed.