Web Services Reliable Messaging TC WS-Reliability 1.1

OASIS Standard, 15 November 2004

Document identifier:

wsrm-ws_reliability-v1.1-spec-os

Location:

http://docs.oasis-open.org/wsrm/ws-reliability/v1.1

Editors:

Principal editor:
Kazunori Iwasa, Fujitsu Limited <kiwasa@jp.fujitsu.com>
Assisting editors:
Jacques Durand, Fujitsu Software Corporation <jdurand@us.fujitsu.com>
Tom Rutt, Fujitsu Software Corporation <trutt@us.fujitsu.com>
Mark Peel, Novell, Inc. <mpeel@novell.com>
Sunil Kunisetty, Oracle Corporation <Sunil.Kunisetty@oracle.com>
Doug Bunting, Sun Microsystems <Doug.Bunting@sun.com>

Abstract:

Web Services Reliability (WS-Reliability) is a SOAP-based protocol for exchanging SOAP messages with guaranteed delivery, no duplicates, and guaranteed message ordering. WS-Reliability is defined as SOAP header extensions and is independent of the underlying protocol. This specification contains a binding to HTTP.

Status:

This document is an OASIS Standard.

Committee members should send comments on this specification to the wsrm@lists.oasis-open.org list. Others should use the comment form at http://www.oasis-open.org/committees/comments/form.php?wg_abbrev=wsrm.

For information on whether any patents that may be essential to implementing this specification have been disclosed and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Web Services Reliable Messaging TC web page (http://www.oasis-open.org/committees/wsrm/).

If necessary, the errata page for this version of of the specification will be located at http://www.oasis-open.org/committees/wsrm/documents/errata/1.1/index.html.

Table of Contents

List of Tables

1 Introduction

1.1 Purpose of WS-Reliability

WS-Reliability is a SOAP-based ([SOAP 1.1] and [SOAP 1.2 Part 1]) specification that fulfills reliable messaging requirements critical to some applications of Web Services. SOAP over HTTP [RFC2616] is not sufficient when an application-level messaging protocol must also guarantee some level of reliability and security. This specification defines reliability in the context of current Web Services standards. This specification has been designed for use in combination with other complementary protocols (see **Section** 1.4) and builds on previous experiences (e.g., ebXML Message Service [ebMS].)

1.2 Definition and Scope of Reliable Messaging

Reliable Messaging (RM) is the execution of a transport-agnostic, SOAP-based protocol providing quality of service in the reliable delivery of messages. There are two aspects to Reliable Messaging; both must be equally addressed when specifying RM features:

(1) **The "wire" protocol** aspect. RM is a protocol, including both specific message headers and specific message choreographies, between a sending party and a receiving party.

(2) **The quality of service** (QoS) aspect. RM defines a quality of messaging service to the communicating parties, viz., the users of the messaging service. This assumes a protocol between these users and the provider of this service (i.e., the reliable messaging middleware). This protocol is defined by a set of abstract operations: Submit, Deliver, Notify, Respond (defined in **Section** 1.5).

Reliable messaging requires the definition and enforcement of contracts between:

- The Sending and Receiving message processors (contracts about the wire protocol)
- The messaging service provider and the users of the messaging service (contracts about quality of service).

Each major RM feature will be defined as a composition of these two types of contract.

Example: Guaranteed message delivery is defined as both (1) a messaging protocol involving Acknowledgment Indications and specific message headers and (2) as a rule guaranteeing if "Submit" completes successfully for a payload on the sending side, "Deliver" completes successfully for this payload on the receiving side or "Notify" (of failure) will be invoked on the sending side.

Figure 1 shows all of the reliability contracts (both QoS and protocol) binding the Reliable Messaging entities (a producer of reliable messages, a consumer of reliable messages, and the two Reliable Messaging Processors or RMPs). The direction of the arrows for the QoS contract abstract operations, shown in **Figure 1**, represents the direction of information flow associated with the operation.

Note:

This specification does not make any assumption about the implementation of a messaging service user component (Producer or Consumer components in **Figure 1**): such a component could be an application, a queuing or logging system, a database, a SOAP node, or the next handler in the message processing chain. The QoS contracts concern only the conditions of invocation of the "Deliver", "Submit", "Respond" and "Notify" operations. The interpretation of these operations is a matter of implementation.

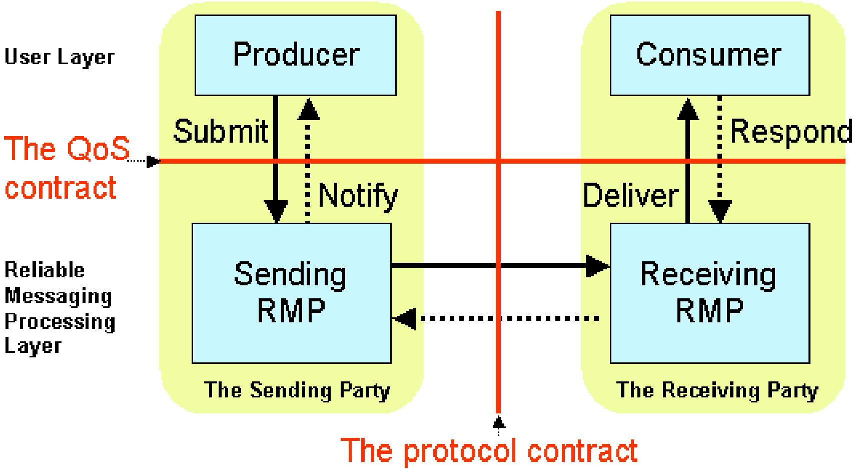

Figure 1 Reliable Messaging Contracts

The current specification defines the following reliability features:

- Guaranteed message delivery, or At-Least-Once delivery semantics.
- Guaranteed message duplicate elimination, or At-Most-Once delivery semantics.
- Guaranteed message delivery and duplicate elimination, or Exactly-Once delivery semantics.
- Guaranteed message ordering for delivery within a group of messages.

Some messaging features are out of scope for this specification. They are:

- Routing features. This specification addresses end-to-end reliability and is not concerned with intermediaries. The mechanisms described are orthogonal to routing techniques and can be used in combination with them.
- Transactions. Transactional messaging ensures the integrity of exchange patterns that involve possibly several messages. Failure conditions may involve application-level decisions based on message payload interpretation. This specification is concerned with the reliability of individual messages from submission to delivery; it ignores any interpretation of these messages.

Reliability is often associated with quantitative measures in QoS areas other than Web services (e.g., networking). Thresholds such as rate of failures, minimal size of persistent store, average latency, and quantitative measures that may appear in service level agreements (SLAs) are out of scope for this version.

1.3 Notational Conventions

This document occasionally uses terms that appear in capital letters. When the terms "MUST", “REQUIRED”, “SHALL”, "SHOULD", "RECOMMENDED", “MAY”, “OPTIONAL”, "MUST NOT", “NOT REQUIRED”, “SHALL NOT” and "SHOULD NOT" appear capitalized, they are being used to indicate particular requirements of this specification. An interpretation of the meanings of these terms appears in [RFC2119].

All text in this specification is normative, except the following:

- examples
- notes (identified with a preceding “**Note**” header)

- appendices not explicitly identified as normative

Section 4 includes tables to explain each message header element. The meaning of the labels in these tables is as follows:

Label	***Meaning***
Cardinality	A constraint on the number of instances of the element, as allowed in its enclosing element (e.g., “0 or 1” means means the element may be either absent or present only once in its enclosing element).
Value	A type or format for a value of the element.
Attributes	Attribute names for the element. The type or format for the attribute value is included in parentheses.
Child elements	Elements allowed as direct descendants of the element.

Table 1 Labels

This specification uses the following namespace prefixes:

Prefix	***Namespace***
soap	http://schemas.xmlsoap.org/soap/envelope/
soap12	http://www.w3.org/2003/05/soap-envelope
wsrm	http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd
xs	http://www.w3.org/2001/XMLSchema/
wsdl11	http://schemas.xmlsoap.org/wsdl/
fnp	http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd
wsrmfp	http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd
ref	http://docs.oasis-open.org/wsrm/2004/06/reference-1.1.xsd

Table 2 Prefixes

The choice of any namespace prefix is arbitrary and not semantically significant.

XPath [XPath 1.0] is used to refer to header elements, in particular in **Section** 4.

1.4 Relation to Other Specifications

- **W3C SOAP 1.1/1.2**: SOAP 1.1 [SOAP 1.1] and SOAP 1.2 [SOAP 1.2 Part 1] are the base protocols for this specification. This specification defines reliable messaging protocol features expressed as extension header blocks embedded in the SOAP Header.
- **OASIS ebXML Message Service Specification 2.0**: The reliable messaging mechanism defined in the ebXML Message Service Specification 2.0 [ebMS] is implemented in a number of products and open source efforts, many of which have undergone interoperability testing. WS-Reliability borrows from this technology.

- **OASIS Web Services Security: SOAP Message Security 1.0**: This specification defines reliability independently from security, each of these features mapping to different SOAP header extensions. Although both features can be used in combination, the specification does not attempt to compose them in a more intricate way, nor does it attempt to profile their combination. This specification can be used with OASIS Web Services Security: SOAP Message Security 1.0 [WSS].
- **WS-I Basic Profile 1.1**: This specification defines how to use reliability in compliance with WS-I Basic Profile 1.1 [WS-I BP 1.1].

1.5 Terminology

Some of these definitions may reference other definitions, either within or outside of the terminology section.

Reliable Messaging (RM):

The act of processing the set of transport-agnostic SOAP Features defined by WS-Reliability, which results in a protocol supporting quality of service features such as guaranteed delivery, duplicate message elimination, and message ordering.

Reliable Messaging Processor (RMP):

A SOAP processor and other infrastructure capable of performing Reliable Messaging as described by this specification. With regard to the transmission of a Reliable Message from one RMP to another, the former is referred to as the Sending RMP and the latter as the Receiving RMP. An RMP may act in both roles.

Reliable Message:

A SOAP message containing a <wsrm:Request> header block.

Payload:

A subset of the message data intended for the Consumer or Producer of the Reliable Message and provided by the Producer or Consumer respectively.

Producer (or Payload Producer)

An abstract component that produces the payload of a message to be sent. An example of a Producer is an application component able to invoke an RMP to send a payload.

Consumer (or Payload Consumer)

An abstract component that consumes the payload of a received message after it has been processed by the Receiving RMP. Examples of Consumers are: an application component called back when a message is received, a queuing device storing received payloads.

Deliver:

An abstract operation that transfers a payload from Receiving RMP to Consumer.

Submit:

An abstract operation that transfers a payload from Producer to Sending RMP – for example, a request to the Sending RMP to handle the payload subject to a reliability agreement.

Respond:

An abstract operation that transfers a payload from Consumer to Receiving RMP as a response to a previously received Reliable Message.

Notify:

An abstract operation that makes available to the Producer a failure status of a previously sent message (e.g., a notification the Sending RMP failed to send a Reliable Message) or transfers a payload received as a response from Sending RMP to Producer.

RMP Operations:

Deliver, Submit, Respond and Notify are also called “RMP operations”. These abstract operations control the transfer of payload data (and, in one case, failure information) between the RMP and a user component (Producer or Consumer). An RMP operation is not necessarily implemented by an RMP, but it must be either supported in some way by an RMP or invoked by the RMP.

Message Identifier:

A message header value or a combination of message header values that uniquely identifies a Reliable Message. This identifier is meaningful only to the reliability features described here.

Duplicate Message:

A message is a duplicate of another message if it has same Message Identifier.

Message Delivery:

Completion of the Deliver operation for a Reliable Message.

Acknowledgment Indication:

An indication that refers to a previous message delivered by the Receiving RMP. An Acknowledgment Indication signals that the acknowledged message has been successfully delivered (that is, the message has satisfied all of the reliability requirements placed on it for delivery).

Reliable Messaging Fault Indication (RM Fault):

An indication referring to a previous message that encountered a Reliable Messaging fault condition at the Receiving RMP: it signals to the Sending RMP of the referred message that there was a failure to invoke the Deliver operation for the message.

Reliable Messaging Reply (RM-Reply):

An indication – either an Acknowledgment Indication or a Reliable Messaging Fault Indication – referring to a previous Reliable Message.

Response, Callback and Poll RM-Reply Patterns:

See **Section** 2.5.

PollRequest Message:

A message from the Sending RMP to the Receiving RMP that requests RM-Replies for its identified set of previously sent Reliable Messages.

Intermediary:

A SOAP node between a Sending RMP and a Receiving RMP.

Publish (an RM-Reply):

The set of mechanisms that make an RM-Reply available to the Sending RMP. The particular mechanism used for a given Publish operation depends on the RM-Reply Pattern (**Section** 2.5) requested within the Reliable Message that elicited the Publish.

2 Messaging Model

2.1 Messaging Context

The Reliable Messaging Model described in this document makes the following assumptions about SOAP messaging and its relation to the RMP behavior:

- **Intermediary transparency**. SOAP Intermediaries do not play any active role in the reliability mechanisms. They can be abstracted from the communication between Sending RMP and Receiving RMP: the RMPs are the only parties involved in implementing the RM protocol (e.g., for handling RM-Replies). There is no role for an RMP other than Receiving RMP or Sending RMP. **Figure 2** illustrates this model.

- **Message integrity**. For the reliability mechanisms described here to fulfill the reliability contract, this specification strongly RECOMMENDS that message header integrity be guaranteed end-to-end by using adequate security options such as those described in Web Services Security: SOAP Message Security 1.0 [WSS].

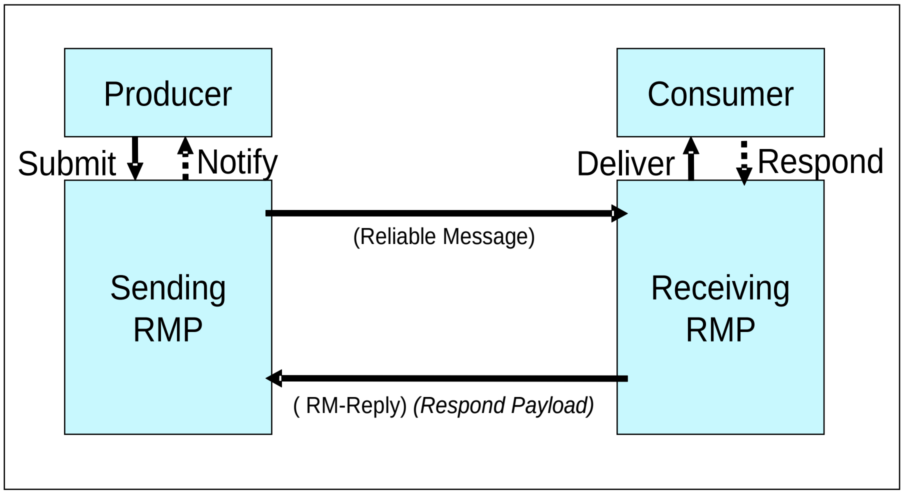

Figure 2 Messaging Model

2.2 RMP Operations and Their Invocation

Four operations (Submit, Deliver, Respond and Notify) are used to model the reliability contracts between an RMP and its users (Producer and Consumer components).

These operations and executable components are defined abstractly to simplify discussion of the WS-Reliability protocol, not to imply a particular API or component separation. No requirement is made herein about how these operations should be implemented, which component should implement them, or whether an implementation should explicitly represent them. The operations themselves describe a transfer of information (payload or failure notice) between an RMP and associated external components (Producer, Consumer).

The separations assumed here between the RMPs and their external components indicate the expected value of placing WS-Reliability support within an infrastructure component. However, any implementation choice leading to the externally observable properties describe in this specification is equally valid.

For example, a Receiving RMP could put a received payload in a queue; later, an application component gets the payload from that queue. This situation could be modeled in two different ways: (1) the queuing middleware is the Consumer, in which case the delivery is over when the payload is placed in the queue, (2) the application component is the Consumer, in which case the delivery is over when the payload is read by the application. Note that the reliability contracts will differ in each case and that it is an implementation choice to decide the precise point at which the reliability contract is considered fulfilled.

The following requirements are associated with the use of RMP operations:

- For every valid and non-expired message it receives, a Receiving RMP MUST invoke the Deliver operation after the associated reliability requirements (ordering, duplicate elimination) have been satisfied.
- The Sending RMP is NOT REQUIRED to invoke the Notify operation for communicating the status of every Reliable Message to a Producer. Only the failure status and available Consumer payload cases need be reported.
- An invocation of Deliver is not always matched by an invocation of Respond; the Consumer is NOT REQUIRED to invoke Respond for every Reliable Message delivered. A Receiving RMP MUST be capable of mapping a pair of Deliver and Respond invocations to an instance of SOAP Request-response MEP (See 2.3)

The basic exchange patterns described in the following section derive from the above messaging assumptions. Reliability features defined in this specification will in turn rely on these patterns.

2.2.1 Binding between WSDL Operation Types and RMP Invocations

This specification supports Reliable Messaging capabilities for WSDL 1.1 [WSDL 1.1] One-way and Request-response operation types only. That is, a WSDL instance describing the Consumer interface would use one of these two operations. Assuming a Sending RMP (or S-RMP) and a Receiving RMP (or R-RMP), the operations in such a WSDL instance MUST bind with the RMP operations in the following way:

- A successful WSDL One-way operation maps to a sequence of RMP invocations of the form: S-RMP.Submit(p) + R-RMP.Deliver(p), where (p) is the payload sent in the request (input message) of the operation described in WSDL.
- A successful WSDL Request-response operation maps to a sequence of RMP invocations of the form: S-RMP.Submit(p) + R-RMP.Deliver(p) + R-RMP.Respond(p2) + S-RMP.Notify(p2), where (p) is the payload sent in the request and (p2) is the payload returned in the response (output message) of the operation described in WSDL.

2.3 Assumed SOAP Message Exchange Patterns

Although SOAP [SOAP 1.1] was initially defined as a one-way messaging protocol, support for other exchange patterns [SOAP 1.1], message exchange patterns (MEPs) [SOAP 1.2 Part 2], and operations [WSDL 1.1] has been described. For example, SOAP over HTTP was principally described in terms of a request-response exchange pattern in [SOAP 1.1], bound to either One-way or Request-response operations in [WSDL 1.1] and restricted (especially with regard to the meaning of a One-way operation) in [WS-I BP 1.1]. Described below are two MEPs – called here SOAP MEPs – of interest for the RM features specified herein and derived from the terminology in those specifications. We use these terms to describe how the RMPs send and receive SOAP messages over the underlying transfer protocol.

An RMP MUST know which SOAP MEP is in use when sending or receiving a Reliable Message. A WSDL instance is just one way among many to specify to an RMP a message's binding to a SOAP MEP.

SOAP One-way MEP:

From an RMP perspective, support for this MEP assumes the following:

- The Sending RMP (as a SOAP node) is able to initiate the sending of a SOAP envelope over the underlying protocol (i.e., not as a result of a previous protocol action such as an HTTP GET or POST).
- No response containing a SOAP envelope is sent back – although a non-SOAP response (e.g., an HTTP error code) may be returned.

SOAP Request-response MEP:

From an RMP perspective, support for this MEP assumes the following:

- The Sending RMP is able to initiate the sending of a SOAP envelope over the underlying protocol.
- The Receiving RMP can send back a message with a SOAP envelope (called a response) after somehow associating the response with the request.

2.4 Message Reply Patterns

There are three ways to publish an RM-Reply (Acknowledgment Indication or Fault Indication):

2.4.1 Response RM-Reply Pattern

When the Response RM-Reply Pattern is in use, the following sequence of exchanges MUST occur:

Step 1: The Sending RMP sends the Reliable Message in a request of a SOAP Request-response MEP instance.

Step 2: The Receiving RMP sends the RM-Reply in the response message of the same SOAP MEP instance.

Figure 3 shows this reply pattern.

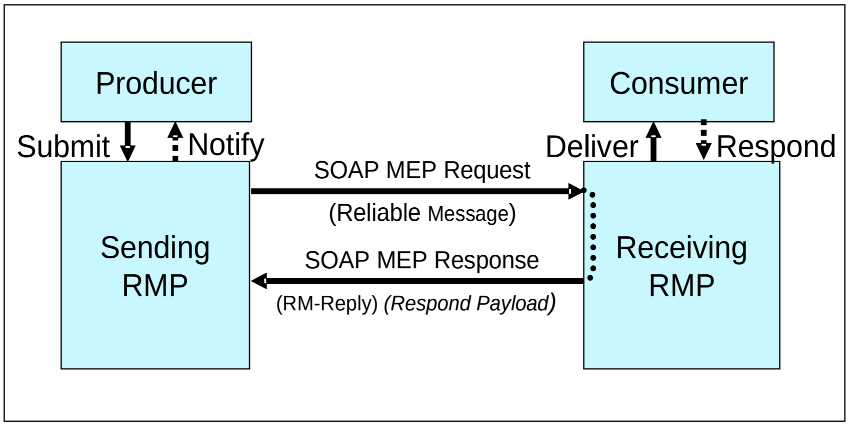

Figure 3 Response RM-Reply Pattern

The Response RM-Reply Pattern MUST NOT be used for WSDL One-way operations to the Consumer.

2.4.2 Callback RM-Reply Pattern

When the Callback RM-Reply Pattern is in use, the following sequence of exchanges MUST occur:

Step 1: The Sending RMP sends the Reliable Message in the SOAP MEP instance required by this Producer-Consumer exchange. This MEP instance may be either Request-response or One-way.

Step 2: The Receiving RMP sends the RM-Reply. Except when the RM Reply is bundled with a Reliable Message (as described in **Section** 4.4), the RMP MUST send this RM-Reply using a SOAP One-way MEP.

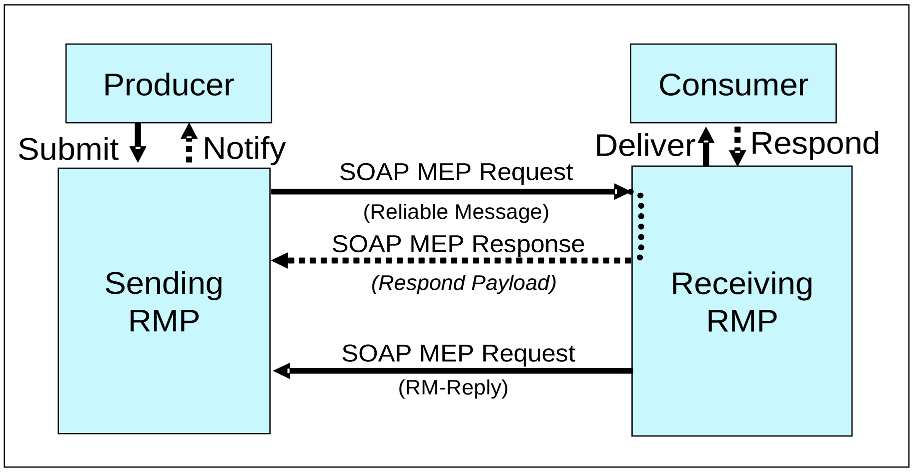

Figure 4 Callback RM-Reply Pattern

Figure 4 shows this reply pattern. The dashed arrows indicate the SOAP message returned when a SOAP Request-response MEP is used to send the Reliable Message.

2.4.3 Poll RM-Reply Pattern

When the Poll RM-Reply Pattern is in use, the following sequence of exchanges MUST occur:

Step 1: The Sending RMP sends the Reliable Message in the SOAP MEP instance required by this Producer-Consumer exchange. This MEP instance may be either Request-response or One-way.

Step 2: The Sending RMP issues a message with a PollRequest element in a new SOAP MEP instance; this acts as a request for Acknowledgment. This message MUST NOT contain a payload (as defined in **Section** 1.5). The Sending RMP MUST use the request of a SOAP Request-response MEP instance for a synchronous PollRequest and MUST use a SOAP One-way MEP for an asynchronous PollRequest.

Step 3: The Receiving RMP sends the RM-Reply either (if synchronous polling) in the response message of the same SOAP instance that carried the PollRequest or (if asynchronous polling) in a message from a SOAP One-way MEP instance. This message MUST NOT contain a payload.

When the Sending RMP of Reliable Messages cannot receive underlying protocol requests (e.g., due to security restrictions), it may use the synchronous version of this reply pattern. The Sending RMP MAY also use this reply pattern (steps 2 and 3 above) to extend other RM-Reply Patterns. **Figure 5** illustrates the synchronous variant, **Figure 6** the asynchronous.

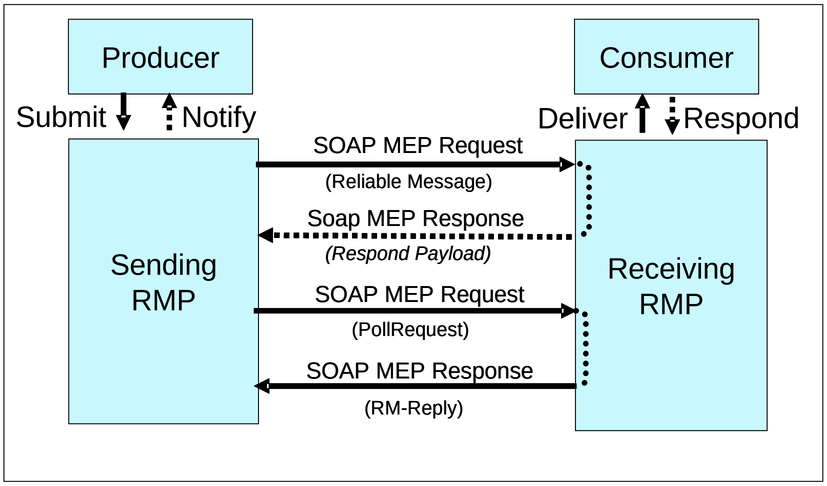

Figure 5 Synchronous Poll RM-Reply Pattern

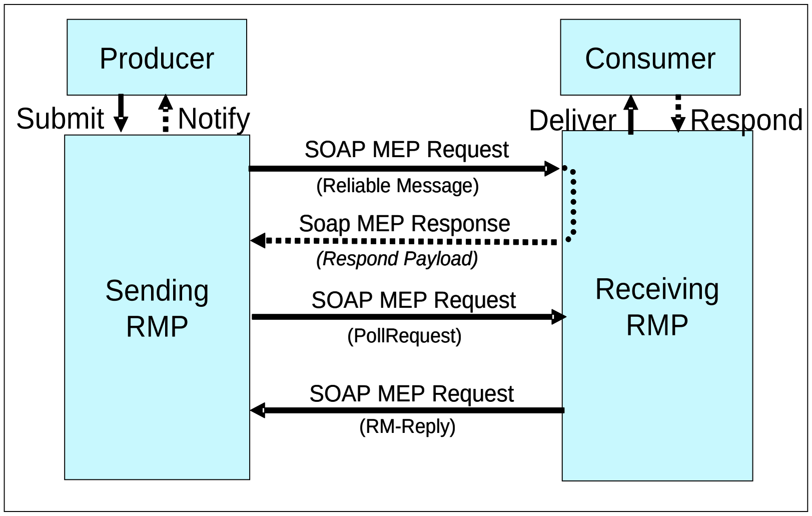

Figure 6 Asynchronous Poll RM-Reply Pattern

2.5 Message Identification and Grouping

A Reliable Message contains an Identifier that is globally unique and relies on the notion of a group. A Reliable Message always belongs to a group. The Sending RMP sends a group of messages to the Receiving RMP as a sequence of individual messages. The Reliable Message Identifier is a combination of a group ID and an optional sequence number; a sequence number, if present, is an integer that is unique within a group. More precisely, a message is uniquely identified as follows:

1) When there is only one message in the group: the group ID, which is a globally unique group identifier, may be used alone as Message Identifier. No sequence number is required, although one is allowed.

2) When the message belongs to a group of several messages: the message is identified by the group ID and a unique sequence number.

3 Reliability Agreement and Features

3.1 RM Agreement

3.1.1 Definition

An agreement for messaging reliability, or RM Agreement, describes which reliability features a sending party and a receiving party have agreed to use when exchanging a set of messages. The RM Agreement can be seen as a contract at two levels: (1) quality of service (QoS), about the conditions and quality of message delivery to the Consumer and (2) protocol features, including timing parameters and details about choreography between the Sending and Receiving RMPs.

3.1.2 RM Agreement Items

An RM Agreement is a list of Agreement Items.

A Sending RMP MUST be capable of (1) taking knowledge (whether by configuration, an API call, a message, the result of an algorithm or any other means) of a set of values that represent the RM Agreement Items described in this specification and (2) processing them according to the semantics described in this specification.

A Receiving RMP MUST be capable of (1) taking knowledge of the Agreement items as they are communicated via the header elements of Reliable Messages and (2) processing them according to the semantics described in this specification.

Table 3 shows the Agreement Items this specification uses. Each item is listed with its possible values:

Name	***Value***	***Definition***
GuaranteedDelivery	enabled/disabled	For setting Guaranteed Delivery (see **Section** 3.2.1 for details).
NoDuplicateDelivery	enabled/disabled	For setting message delivery without duplicates or Duplicate Elimination (see **Section** 3.2.2 for details).
OrderedDelivery	enabled/disabled	For setting Guaranteed Message Ordering (see **Section** 3.2.3 for details).
GroupMaxIdleDuration	number of seconds	For setting the elapsed time limit from the last message sent or received in a group, after which the group can be terminated. The value MUST NOT be zero or smaller.
GroupExpiryTime	date/time	For setting the date and time after which the group can be terminated.
ExpiryTime	date/time	For setting the date and time after which a message must not be delivered to the Consumer.
ReplyPattern	"Response", "Callback", "Poll"	For setting the mode of response for Acknowledgments or Faults.

Table 3 RM Agreement Items

3.1.3 Scope of an Agreement Item

There are two scopes to consider:

- Group scope: All messages sent within a group.
- Message Scope: A single message.

Agreement Items relate to a particular scope: for example, ExpiryTime affects each message separately, while GroupExpiryTime is an Agreement Item about groups.

Agreement items applying to the Message Scope MAY be applied to the Group Scope. For example, an RMP implementation may decide to specify the same ExpiryTime value for all messages of a group and not support setting different values for messages in a group. The default scope of applicability for each RM Agreement item is:

Message scope:

- ExpiryTime
- ReplyPattern

Group scope:

- OrderedDelivery
- GuaranteedDelivery
- NoDuplicateDelivery
- GroupExpiryTime
- GroupMaxIdleDuration

An RMP MUST NOT allow most Agreement items applicable at Group scope to vary between messages of a group. For example, a Sending RMP MUST NOT use different guaranteed delivery modes for different messages of a group. However, it is allowed to dynamically change the value of GroupExpiryTime or GroupMaxIdleDuration pertaining to a group (See **Section** 5.1.2).

3.1.4 Rules

When defining an RM Agreement instance, there are some dependencies between the items of the agreement that must be respected:

- If OrderedDelivery is enabled for a group, GuaranteedDelivery and NoDuplicateDelivery MUST also be enabled for that group.
- If GroupExpiryTime is used for a group, the item GroupMaxIdleDuration MUST NOT be used for this group and vice versa.

3.1.5 Creation, Representation and Deployment of RM Agreements

The concrete representation of an RM Agreement is beyond the scope of this specification, as this may be part of a more general agreement that covers other matters as well as the reliability aspect. However, the RM Agreement determines the use of the reliability protocol and the behavior of RMPs. For these reasons, this specification references the RM Agreement in an abstract way, showing it as a simple list of (name, value) pairs called Agreement Items. This allows a description of the concrete effect of each Agreement Item on the message content and flow. Once there is a broad enough consensus for using a particular representation for agreements, a future version of this specification will define a corresponding binding for RM Agreements.

The way RM Agreements are established or communicated to each party is out of scope. However, one of the principles of this specification is that it should not be necessary to deploy an RM Agreement on both RMPs prior to executing business transactions. Only the Sending RMP needs to have knowledge of the RM Agreement initially. No prior communication of the agreement to the receiving party (an RMP and its user) is required. The only input the Receiving RMP will need in order to enforce the reliability requirements will be obtained from the header elements of received messages.

3.1.6 RM Capability

As a way to support the creation of RM Agreements, it may be useful for Web services providers to advertise somehow the reliability features (or RM Agreement Item values) supported by a deployed Web service. In contrast with agreements involving both parties, such reliability features – called RM Capabilities – may conveniently be associated with WSDL definitions. In support of this option, this specification proposes a concrete representation for these capabilities (see **Appendix** B).

3.2 Main Reliability Features

The main reliability features mentioned in **Section** 1 are formally described here in terms of requirements. This specification provides the means to enforce these requirements. A detailed description of the protocol features implementing these means is given in **Section** 4 and beyond.

3.2.1 Guaranteed Delivery

Quality of Service requirements:

When the GuaranteedDelivery Agreement Item is enabled, one of the two following outcomes SHALL occur for each Submit invocation: either (1) the Receiving RMP successfully delivers (Deliver invocation) the submitted payload to its associated Consumer or (2) the Sending RMP notifies (Notify invocation) the Producer associated with that payload of a delivery failure.

Notes:

- This QoS feature guarantees only that the sender will always be notified of a delivery failure when a message is not delivered. It is, however, impossible to guarantee this while at the same time guaranteeing that (1) and (2) will never occur together for the same message. A proper usage by an implementation of the protocol options described in this specification will, however, greatly reduce situations where both (1) and (2) occur.

- The GuaranteedDelivery agreement is defined for messages resulting from invocations of the Submit operation. An extension of this agreement to messages resulting from invocations of the Respond operation is out of scope for this specification.

Protocol requirements:

For all messages sent with the GuaranteedDelivery agreement, a Receiving RMP MUST publish the RM-Reply of each such message that has been either delivered or faulted. The Sending RMP MUST poll for all of its sent messages that requested the Poll RM-Reply Pattern.

A message resending technique combined with the acknowledgment and fault mechanism described here MUST be used in case of a delivery failure. Parameters that control the resending policy (number of retries, frequency, etc.) are out of the scope of this specification. These parameters may be added to an RM Agreement, although the resending policy may need to be dynamically adjusted depending on network conditions. When resending a message, the message contents must not change.

A Receiving RMP MUST NOT publish a Reliable Messaging Fault for a delivered Message. The RMP MUST NOT deliver a message for which a Reliable Messaging Fault has been published.

A Sending RMP MUST NOT resend a message for which an RM-Reply with a Fault type other than MessageProcessingFailure has been received and MUST instead notify its Producer of a delivery failure.

3.2.2 Duplicate Elimination

Quality of Service requirements:

When the NoDuplicateDelivery Agreement Item is enabled, a message resulting from a Submit invocation SHALL NOT be delivered twice or more to the Consumer.

Note:

In the current specification, the NoDuplicateDelivery agreement is defined for messages resulting from invocations to the Submit operation. An extension of this agreement to messages resulting from invocations to the Respond operation is out of scope for this specification.

Protocol requirements:

An implementation of this specification must ensure the following invariants:

- Message instances resulting from separate invocations of Submit MUST NOT share the same Message Identifier.

- When resending a message, the message contents must not change.

As a corollary to the above requirements, a Receiving RMP MUST ensure that once a message under this agreement has been delivered to a Consumer, no message with the same identifier received afterward will be delivered to this Consumer.

When the Response RM-Reply Pattern is requested with Duplicate Elimination for a Reliable Message, the Receiving RMP cannot deliver that message to the Consumer again (because it is a duplicate of a previously delivered message), and a Consumer response payload is expected, the response of the SOAP MEP instance MUST contain one (but not both) of the following:

- a copy of the original response payload returned for that Message (in the SOAP Body) in addition to the Acknowledgment Indication (in the SOAP Header) or

- a SOAP server Fault (in the SOAP Body) in addition to the Acknowledgment Indication (in the SOAP Header).

The Sending RMP and Producer expect either a complete response or a SOAP Fault when using the Response RM-Reply Pattern; these two allowed behaviors satisfy that expectation.

3.2.3 Guaranteed Message Ordering

Quality of Service requirements:

When the OrderedDelivery Agreement Item is enabled, messages resulting from a sequence of Submit invocations SHALL be delivered in the same order to the Consumer. In addition, when the Receiving RMP delivers one of these messages, all previous messages submitted in the sequence MUST already have been delivered (no missing message allowed).

Note:

In the current specification, the OrderedDelivery agreement is defined for messages resulting from invocations of the Submit operation on the Sending RMP. An extension of this agreement to messages resulting from invocations of the Respond operation is out of scope for this specification.

Protocol requirements:

Ordering is supported only over messages of the same group.

An implementation of this specification must ensure the following invariants, regarding the usage of sequence numbers (SequenceNum element):

- The Sending RMP MUST reflect the order of the Submit invocations on this RMP in the sequence numbers of the corresponding messages sent.

- The Receiving RMP MUST deliver the messages received according to the order expressed by their sequence numbers, which is the same as the submission order.

An RMP will terminate the group as specified in **Section** 5.1.3.5 (T5) when those conditions arise.

609 # 4 Message Format

610 ## 4.1 Structure

611 **Figure 7** shows the structure of reliability SOAP header blocks in the SOAP Envelope, as specified by the WS-Reliability protocol. On the left side of the figure, a Reliable Message is characterized by the presence of the wsrm:Request element. On the right side a response to a Reliable Message contains a wsrm:Response element. Both wsrm:Request and wsrm:Response elements may be found in the same message.

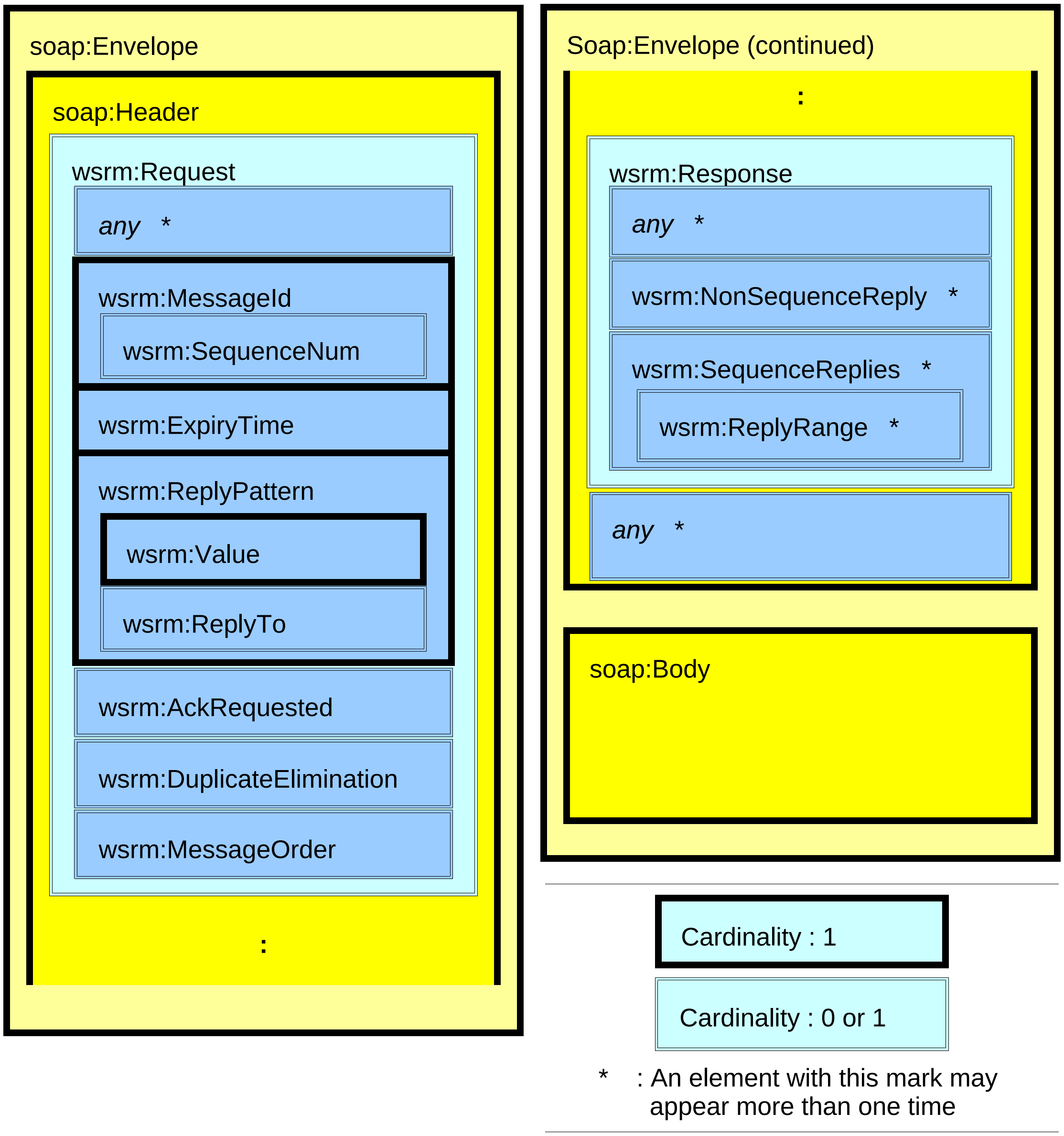

Figure 7 Structure of WS-Reliability elements

Figure 8 shows the structure of PollRequest message embedded in the SOAP Envelope.

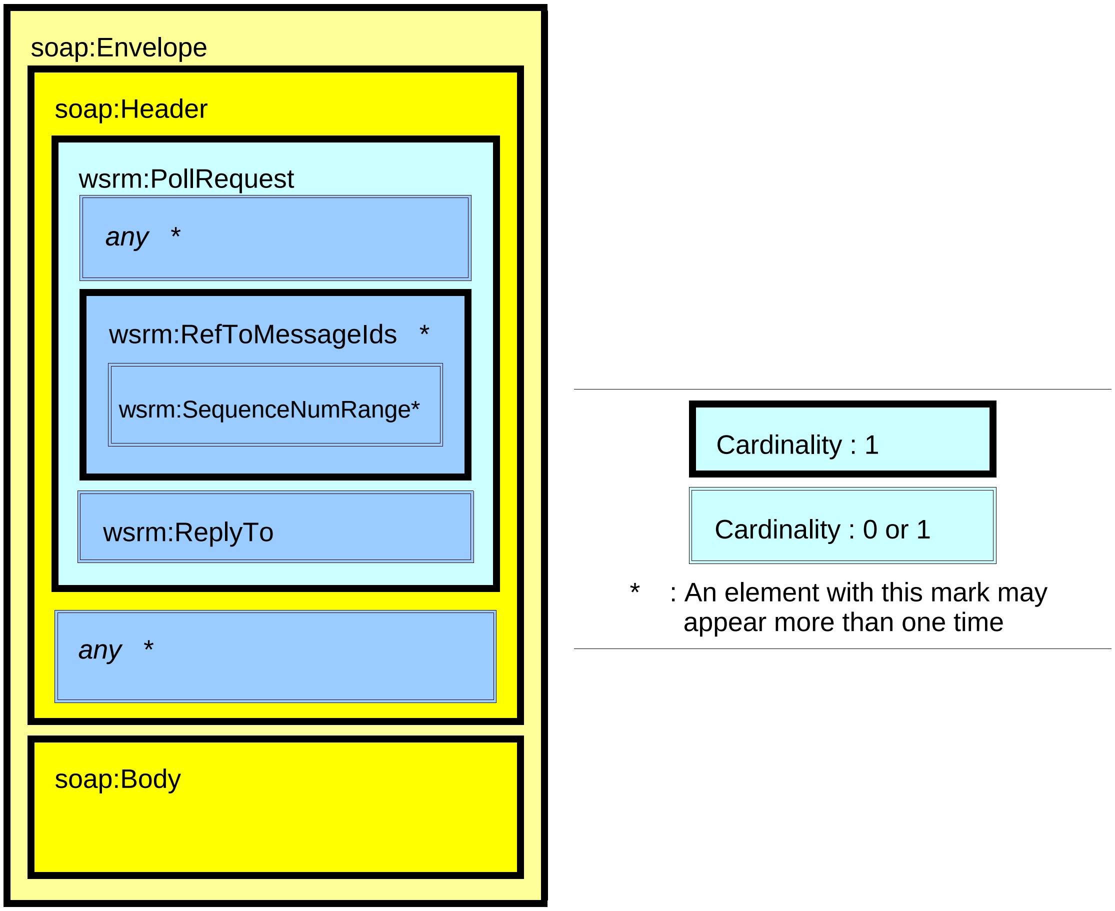

Figure 8 Structure of PollRequest message elements

The namespace [XML Namespaces] for reliable messaging defined in this specification is:

http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd

When the text of the specification is shown to be in conflict with schema statements, the schema statements prevail in the absence of an errata addressing the conflict.

The schema for some of the elements specified in this section includes the specification of extensibility elements and attributes. The extensibility features expressed formally in the schema are specified in **Section** 4.6.

If a message contains additional elements or attributes not described in this specification, the Reliable Messaging Processor MAY ignore them.

Any of the following three elements can be a direct child element of the SOAP Header:

- **Request** element
- **PollRequest** element
- **Response** element

4.2 Request Element

The Request element conveys information about the agreement items that apply to the containing Reliable Message. This element includes the following attribute and child elements (see the description of each child element for cardinality requirements):

- SOAP **mustUnderstand** attribute (see **Appendix** A for details)
- **MessageId** element
- **ExpiryTime** element
- **ReplyPattern** element
- **AckRequested** element
- **DuplicateElimination** element
- **MessageOrder** element

Cardinality	0 or 1
Value	None
Attributes	soap:mustUnderstand (Boolean)
Child elements	MessageId ExpiryTime ReplyPattern AckRequested DuplicateElimination MessageOrder

Table 4 Request Element

Example 1 shows an instance of a Request element.

Example 1 Request Element

```
<Request
 xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
 xmlns:soap12="http://www.w3.org/2003/05/soap-envelope"
 soap12:mustUnderstand="1">
  <MessageId groupId="mid://20040202.103832@wsr-sender.org">
    <SequenceNum number="0"
     groupExpiryTime="2005-02-02T03:00:33-31:00" />
  </MessageId>
  <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
  <ReplyPattern>
    <Value>Response</Value>
  </ReplyPattern>
  <AckRequested/>
  <DuplicateElimination/>
  <MessageOrder/>
</Request>
```

4.2.1 Element: Request/MessageId

This element includes the following attribute:

- a **groupId** attribute

Cardinality	1
Value	None
Attributes	groupId (xs:anyURI)
Child elements	SequenceNum

Table 5 MessageId Element

4.2.1.1 Attribute: Request/MessageId@groupId

This attribute identifies a message group. The Sending RMP MUST use a distinct globally unique @groupId value for each distinct group of messages. Within any such group, all messages will have the same value for @groupId. This identification (the value) is of type URI as defined in [RFC2396]. It is RECOMMENDED that implementations use the Message-ID schema defined in [RFC2392].

4.2.1.2 Element: Request/MessageId/SequenceNum

The Sending RMP MUST include the SequenceNum element in all Reliable Messages of a group with more than one message.

The SequenceNum element carries the sequence number as well as other attributes that may alter the Receiving RMP's processing of the group. When a message includes a MessageOrder element, the sequence number is used in support of message ordering (**Section** 3.2.3).

This element includes the following attributes:

- a **groupExpiryTime** attribute
- a **groupMaxIdleDuration** attribute
- a **number** attribute
- a **last** attribute

In a request message, the sender MAY include either (but not both) @groupExpiryTime or @groupMaxIdleDuration (see **Section** 5.1.2).

Example 2 illustrates the SequenceNum element with some message fragments:

Example 2 SequenceNum Element

1) First message

```
<MessageId groupId="mid://20040202.103832@wsr-sender.org">
  <SequenceNum number="0"
   groupExpiryTime="2005-02-02T03:00:33-31:00" />
</MessageId>
```

2) Second message

```
<MessageId groupId="mid://20040202.103832@wsr-sender.org">
  <SequenceNum number="1"
   groupExpiryTime="2005-02-02T03:00:33-31:00" />
</MessageId>
```

3) The last message for the group

```
<MessageId groupId="mid://20040202.103832@wsr-sender.org">
  <SequenceNum number="2"
   groupExpiryTime="2005-02-02T03:00:33-31:00" last="true" />
</MessageId>
```

Cardinality	1
Value	None
Attributes	groupExpiryTime (dateTime) groupMaxIdleDuration (duration) number (unsignedLong) last (Boolean)
Child elements	None

Table 6 SequenceNum Element

4.2.1.2.1 Attribute: Request/MessageId/SequenceNum@groupExpiryTime

This attribute represents the GroupExpiryTime agreement item (**Section** 3.1.2, **Table 3**). It specifies the the date and time at which the sender wishes the group to terminate. The @groupExpiryTime value is expressed as UTC and conforms to [XML Schema Part 2] dateTime.

The Cardinality of this attribute is 0 or 1. Constraints on the use of this attribute are specified in **Section** 5.

4.2.1.2.2 Attribute: Request/MessageId/SequenceNum@groupMaxIdleDuration

This attribute represents the GroupMaxIdleDuration agreement item (**Section** 3.1.2, **Table 3**). It specifies the maximum idle time for a group. The @groupMaxIdleDuration value conforms to [XML Schema Part 2] duration. The Cardinality of this attribute is 0 or 1. Constraints on the use of this attribute are specified in **Section** 5.

4.2.1.2.3 Attribute: Request/MessageId/SequenceNum@number

This attribute contains the sequence number, which identifies the message within its group (**Section** 2.6) and is used in support of message ordering (**Section** 3.2.3). @number conforms to [XML Schema Part 2] unsignedLong.

The Sending RMP MUST set this value to 0 for the first message of a group. The Sending RMP thereafter MUST increment this value by 1 for each message submitted in this group. Once the value reaches the maximum (18446744073709551615, the maximum value for this data type), the group is terminated (see **Section** 5).

4.2.1.2.4 Attribute: Request/MessageId/SequenceNum@last

This attribute indicates whether or not the containing message is the last in a group. The Cardinality of this attribute is 0 or 1. When this attribute is present, its Boolean value has the following meaning:

- **false:** Indicates the message is not the last message of the group or is not known to be the last message of the group.

- **true:** Indicates the message is known to be the last message sent within a group of messages.

When this attribute is not present, its value defaults to false.

4.2.2 Element: Request/ExpiryTime

The ExpiryTime element represents the ExpiryTime agreement item (**Section** 3.1.2, **Table 3**). It indicates the ultimate date and time after which the Receiving RMP MUST NOT invoke the Deliver operation for the received message. The message is considered expired if the current time, expressed in UTC, is greater than the value of the ExpiryTime element. When a message expires on the Sending RMP before being successfully sent, a Sending RMP MUST NOT send or resend it and MUST communicate a delivery failure to the Producer. The time is expressed as UTC and conforms to [XML Schema Part 2] dateTime.

Cardinality	1
Value	xs:dateTime
Attributes	None
Child elements	None

Table 7 ExpiryTime Element

4.2.3 Element: Request/ReplyPattern

A Sending RMP MUST include the ReplyPattern element in a Request element. The ReplyPattern element includes the following child elements:

- a **Value** element
- a **ReplyTo** element

Cardinality	1
Value	None
Attributes	None
Child elements	Value ReplyTo

Table 8 ReplyPattern Element

4.2.3.1 Element: Request/ReplyPattern/Value

The Value element indicates which reply pattern the Sending RMP requests. This element specifies whether the Receiving RMP should send the Acknowledgment Indication or RM Fault Indication back in the response to the reliable message, in a separate callback request, or in the response to a separate poll request. A Sending RMP MUST include the Value element in a ReplyPattern element. This element has one of the following three values:

- **Response**
- **Callback**
- **Poll**

These values respectively indicate which of the RM-Reply Patterns – Response, Callback or Poll – is in use, as described in **Section** 2.5.

Cardinality	1
Value	xs:string: Response, Callback or Poll
Attributes	None
Child elements	None

Table 9 Value Element

4.2.3.2 Element: Request/ReplyPattern/ReplyTo

If the value of the Request/ReplyPattern/Value element is "Callback", the Sending RMP MUST include this element in the Reliable Message. For all other values ("Poll" and "Response") of Request/ReplyPattern/Value element, the Sending RMP MUST NOT include this element. This element specifies the endpoint where the Sending RMP expects to receive a callback containing RM-Reply information.

If present, the reference-scheme attribute specifies the format of the single child element of the ReplyTo element. If the attribute is omitted, the default content of the ReplyTo element is BareURI.

Cardinality	0 or 1
Value	None
Attributes	reference-scheme
Child elements	*{xs:anyType}* (an element representing the reference)

Table 10 ReplyTo Element

4.2.3.2.1 Attribute: Request/ReplyPattern/ReplyTo@reference-scheme

This attribute specifies the format or schema of the child element of Request/ReplyPattern/ReplyTo. The Sending RMP MUST omit this attribute when the child element of Request/ReplyPattern/ReplyTo is BareURI. The type of this attribute is xs:anyURI.

4.2.3.2.2 Element: Request/ReplyPattern/ReplyTo/BareURI

This element provides one of the simplest referencing options, the URI of the callback recipient's endpoint. It is the default content of the Request/ReplyPattern/ReplyTo and PollRequest/ReplyTo (see **Section** 4.3.1) elements, though the Sending RMP MAY use any other element and scheme supported by the Receiving RMP. This location (the value) is of type URI as defined in [RFC2396].

Section 6 provides additional information about the specific case for which the content of a BareURI in a Request or PollRequest element uses the HTTP URI scheme.

Cardinality	0 or 1
Value	xs:anyURI
Attributes	None
Child elements	None

Table 11 BareURI Element

4.2.4 Element: Request/AckRequested

A Sending RMP MUST include the AckRequested element in a message if and only if that message is subject to the GuaranteedDelivery Agreement Item (refer to **Section** 3.2.1 for details); as described in **Section** 3.1.4, this condition includes all messages subject to the OrderedDelivery Agreement Item. The Sending RMP uses this element to request the Receiving RMP to publish an Acknowledgment after the message is delivered to the consumer party or else to publish an RM Fault Indication. The Receiving RMP MUST publish this information, even for received messages that are duplicates of previously delivered messages. For example, if the RM-Reply Pattern is Callback and no fault occurs, an Acknowledgment Indication SHALL be sent back.

The Receiving RMP MAY publish an RM Fault Indication for a Reliable Message, even if the AckRequested element is not present in the Request element for that message.

The pattern used to send the Acknowledgment or RM Fault Indication is determined by the value of the ReplyPattern element.

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

Table 12 AckRequested Element

4.2.5 Element: Request/DuplicateElimination

A Sending RMP MUST include the DuplicateElimination element in a message if and only if that message is subject to the NoDuplicateDelivery Agreement Item (refer to **Section** 3.2.2 for details); as described in **Section** 3.1.4, this condition includes all messages subject to the OrderedDelivery Agreement Item.

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

Table 13 DuplicateElimination Element

4.2.6 Element: Request/MessageOrder

A Sending RMP MUST include the MessageOrder element if and only if that message is subject to the OrderedDelivery Agreement Item (refer to **Section** 3.2.3 for details).

If the MessageOrder element appears in the message received, the Receiving RMP MUST NOT deliver the message until all messages with the same Request/MessageId@groupId value and a lower Request/MessageId/SequenceNum@number value have been delivered.

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

Table 14 MessageOrder Element

4.2.7 Example

The HTTP message below uses the Request element to specify (among other things) that all three reliability features should be used: GuaranteedDelivery ("AckRequested" element), NoDuplicateDelivery ("DuplicateElimination" element), and OrderedDelivery ("MessageOrder" element). The reply pattern is “Poll”, meaning that no Acknowledgment or Fault will be sent back unless explicitly requested by another message containing a PollRequest header.

Example 3 Reliable Message with Request header

```
POST /abc/servlet/wsrEndpoint HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.100
SOAPAction: ""
Content-Length: 736

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
  <Request
   xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
   soap:mustUnderstand="1">
    <MessageId groupId="mid://20040202.103832@wsr-sender.org">
      <SequenceNum number="0"
       groupExpiryTime="2005-02-02T03:00:33-31:00" />
    </MessageId>
      <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
      <ReplyPattern>
        <Value>Poll</Value>
      </ReplyPattern>
      <AckRequested/>
      <DuplicateElimination/>
      <MessageOrder/>
    </Request>
  </soap:Header>
  <soap:Body>
    <Request xmlns="http://example.org/wsr">Request Message</Request>
  </soap:Body>
</soap:Envelope>
```

4.3 PollRequest Element

A PollRequest Message requests an RM-Reply for a Reliable Message that had “Poll” as the value of the Request/ReplyPattern/Value element and included the Request/AckRequested element. However, PollRequest Messages can also solicit delivery status for messages that were originally sent with “Response” or “Callback” as the value of the Request/ReplyPattern/Value element and that included the Request/AckRequested element.

If a Receiving RMP does not support the use of PollRequest as a general status query mechanism, it MAY return a FeatureNotSupported fault in response to a PollRequest when the relevant ReplyPattern Agreement Item does not have the value “Poll”.

A Receiving RMP that receives a supported form of PollRequest MUST publish RM-Reply information relevant to non-expired messages identified in that request.

This element includes the following attribute and child elements:

- SOAP **mustUnderstand** attribute (see **Appendix** A for details)
- a **ReplyTo** element

- a **RefToMessageIds** element

Cardinality	0 or 1
Value	None
Attributes	soap:mustUnderstand (Boolean)
Child elements	ReplyTo RefToMessageIds

Table 15 PollRequest Element

Example 4 PollRequest Element

```
<PollRequest
 xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 soap:mustUnderstand="1">
  <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
    <SequenceNumRange from="0" to="5"/>
    <SequenceNumRange from="15" to="20"/>
  </RefToMessageIds>
  <RefToMessageIds groupId="mid://20040202.103811@wsr-sender.org" />
  <RefToMessageIds groupId="mid://20040202.103807@wsr-sender.org">
    <SequenceNumRange from="713" to="6150"/>
  </RefToMessageIds>
</PollRequest>
```

4.3.1 Element: PollRequest/ReplyTo

The Receiving RMP MUST send the RM-Reply information in a new request to the endpoint specified by PollRequest/ReplyTo whenever this element is present. If it is not present, the Receiving RMP MUST send back the RM-Reply on the response to the PollRequest message.

Section 4.2.3.2 provides additional information about the very similar Request/ReplyPattern/ReplyTo element.

Cardinality	0 or 1
Value	None
Attributes	reference-scheme
Child elements	*{xs:anyType}* (an element representing the reference)

Table 16 ReplyTo Element

4.3.1.1 Attribute: PollRequest/ReplyTo@reference-scheme

Section 4.2.3.2.1 provides additional information about the similar Request/ReplyPattern/ReplyTo@reference attribute.

4.3.1.2 Element: PollRequest/ReplyTo/BareURI

Section 4.2.3.2.2 provides additional information about the similar Request/ReplyPattern/ReplyTo/BareURI element.

Cardinality	0 or 1
Value	xs:anyURI
Attributes	None
Child elements	None

Table 17 BareURI Element

4.3.2 Element: PollRequest/RefToMessageIds

The RefToMessageIds element contains the identifiers of groups and messages whose status the Sending RMP is requesting. This element includes @groupId and zero or more SequenceNumRange elements as follows:

- a **groupId** attribute
- zero or more **SequenceNumRange** elements

Cardinality	1 or more
Value	None
Attributes	groupId (URI)
Child elements	SequenceNumRange

Table 18 RefToMessageIds Element

When this RefToMessageIds element does not include a SequenceNumRange element, the Receiving RMP MUST return RM-Replies for non-expired messages that were delivered or faulted in that group.

When the RefToMessageIds element includes one or more SequenceNumRange element(s), the Receiving RMP MUST return RM-Replies for the non-expired messages that were delivered or faulted in the identified subset of that group. The identified subset includes all Reliable Messages whose MessageId/SequenceNum@number values fall in the range(s) specified in the RefToMessageIds/SequenceNumRange element(s) of the PollRequest.

A Sending RMP MAY include multiple RefToMessageIds elements (one for each @groupId value) in a single PollRequest Message to request RM-Replies for multiple groups.

4.3.2.1 Attribute: PollRequest/RefToMessageIds@groupId

The @groupId specifies the group of messages whose status the Sending RMP is requesting. This identification (the value) is of type URI as defined in [RFC2396].

4.3.2.2 Element: PollRequest/RefToMessageIds/SequenceNumRange

The SequenceNumRange element specifies those messages in a group for which the Sending RMP requests status. Attributes @from and @to of this element express an inclusive range for SequenceNum values. This element contains the following two attributes:

- a **from** attribute
- a **to** attribute

When these attributes have the same value, the range is limited to a single message.

Cardinality	0 or more
Value	None
Attributes	from (unsignedLong) to (unsignedLong)
Child elements	None

Table 19 SequenceNumRange Element

4.3.2.2.1 Attribute: PollRequest/RefToMessageIds/SequenceNumRange@from

This attribute specifies the lowest SequenceNum@number value of the message range. The value of @from is of type unsignedLong and SHALL be less than or equal to the value of @to.

4.3.2.2.2 Attribute: PollRequest/RefToMessageIds/SequenceNumRange@to

This attribute specifies the highest SequenceNum@number value of the message range. The value of @to is of type unsignedLong and SHALL be greater than or equal to the value of @from.

4.3.3 Example

The HTTP message below uses the PollRequest reliability element, polling the Receiving RMP for the status of messages within the range of sequence numbers 0 to 20 of a particular group. The response to this PollRequest will identify which of those messages have been delivered (Acknowledged).

Example 5 PollRequest Message embedded in HTTP Request

```
POST /abc/servlet/wsrEndpoint HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.100
SOAPAction: ""
Content-Length: 432

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <PollRequest
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
     soap:mustUnderstand="1">
      <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
        <SequenceNumRange from="0" to="20"/>
      </RefToMessageIds>
    </PollRequest>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

4.4 Response Element

The Response element indicates Acknowledgments and Faults for Reliable Messages. This element includes the following attributes:

- SOAP **mustUnderstand** attribute (see **Appendix** A for details)

The Response element SHALL include a list one or more elements in length containing a choice or choices from the following:

- **NonSequenceReply element(s)**
- **SequenceReplies element(s)**

When the Response occurs under the Response RM-Reply Pattern, the first element in this list describes the status of the received Reliable Message. In this case, when the SequenceReplies element is used, the first contained ReplyRange element will include the received Reliable Message within its range.

The Receiving RMP MAY bundle a Response element with a Request element when responding to a message that used the Callback RM-Reply Pattern. In this case, the response and the new Reliable Message MUST share a common destination URI. This enables the combination of an Acknowledgment Indication and the business response to the original message. This also allows a Receiving RMP to bundle an Acknowledgment Indication with another unrelated message to the Sending RMP to reduce network traffic. When combined in a single message, the Request and Response elements are treated separately from the perspective of the abstract model (**Section** 2); a Receiving RMP component handles the Request element and payload while a Sending RMP handles the Response element.

Cardinality	0 or 1
Value	None
Attributes	soap:mustUnderstand (Boolean)
Child elements	NonSequenceReply SequenceReplies

Table 20 Response Element

Example 6 shows an instance of the Response element.

Example 6 Response Element

```
<Response
 xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 soap:mustUnderstand="1">
  <NonSequenceReply groupId="mid://20040202.103832@wsr-sender.org" />
  <SequenceReplies groupId="mid://20040202.103807@wsr-sender.org">
    <ReplyRange from="1" to="4" />
    <ReplyRange from="5" to="5" fault="wsrm:InvalidRequest" />
    <ReplyRange from="6" to="42" />
  </SequenceReplies>
  <NonSequenceReply groupId="mid://20040202.103811@wsr-sender.org"
   fault="wsrm:PermanentProcessingFailure" />
</Response>
```

4.4.1 Element: Response/NonSequenceReply

An RM-Reply for a message that does not have a sequence number SHALL include a NonSequenceReply element. This element includes the following attributes:

- a **groupId** attribute
- a **fault** attribute

The @fault indicates a particular fault for the identified message. Without this attribute, the NonSequenceReply element is an Acknowledgment Indication for the message.

Cardinality	0 or more
Value	None
Attributes	groupId (URI) fault (QName)
Child elements	None

Table 21 NonSequenceReply Element

4.4.1.1 Attribute: Response/NonSequenceReply@groupId

This attribute specifies the group identifier of a message that did not have a sequence number. A NonSequenceReply element SHALL include the message's @groupId. This identification (the value) is of type URI as defined in [RFC2396].

4.4.1.2 Attribute: Response/NonSequenceReply@fault

This attribute indicates the code of a Reliable Messaging Fault encountered while processing the message. The Cardinality of this attribute is 0 or 1.

4.4.2 Element: Response/SequenceReplies

An RM-Reply for a group (or a subset thereof) whose messages had sequence numbers SHALL include a SequenceReplies element. This element contains a @groupId and 1 or more ReplyRange elements.

Cardinality	0 or more
Value	None
Attributes	groupId (URI)
Child elements	ReplyRange

Table 22 SequenceReplies Element

4.4.2.1 Attribute: Response/SequenceReplies@groupId

The @groupId specifies the message group for which its SequenceReplies element carries the status. A SequenceReplies element SHALL include the group's @groupId. This identification (the value) is of type URI as defined in [RFC2396].

4.4.2.2 Element: Response/SequenceReplies/ReplyRange

The ReplyRange element indicates a range of sequence numbers with a shared delivery status. The @fault indicates a particular, common fault all messages in the range share. Without this attribute, the ReplyRange element is an Acknowledgment Indication for all messages in the range.

Cardinality	1 or more
Value	None
Attributes	from (unsigned Long) to (unsigned Long) fault (QName)
Child elements	None

Table 23 ReplyRange Element

4.4.2.2.1 Attribute: Response/SequenceReplies/ReplyRange@from

This attribute has same type and semantics as in the PollRequest element.

4.4.2.2.2 Attribute: Response/SequenceReplies/ReplyRange@to

This attribute has same type and semantics as in the PollRequest element.

4.4.2.2.3 Attribute: Response/SequenceReplies/ReplyRange@fault

This attribute indicates the code of a Reliable Messaging Fault encountered while processing all of the messages in the identified range. The Cardinality of this attribute is 0 or 1.

4.4.3 Example

The message below uses the Response reliability element, which in this case is carrying the response of a previous PollRequest element. The response acknowledges a message specified by the group identifier “mid://20040202.103811@wsr-sender.org” and messages for a group specified by the group identifier “mid://20040202.103832@wsr-sender.org” within the ranges of sequence numbers 0 to 14 and 16 to 20. The response also reports an RM Fault for a message with sequence number 15 for the group.

Example 7 RM-Reply message embedded in HTTP Response

```
HTTP/1.0 200 OK
Server: WS-ReliabilityServer
Date: Mon, 02 Feb 2004 10:38:32 GMT
Content-Language: en
Content-Type: text/xml; charset=utf-8
Content-Length: 593

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <NonSequenceReply groupId="mid://20040202.103811@wsr-sender.org"/>
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="0" to="14"/>
        <ReplyRange from="15" to="15" fault="InvalidRequest"/>
        <ReplyRange from="16" to="20"/>
      </SequenceReplies>
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

4.5 Fault Codes For Reliable Messaging Failures

The protocol defines two fault categories:

- The Message Format fault set, which includes all faults generated because of a malformed Reliable Message header.
- The Message Processing fault set, which includes all faults generated while processing the message.

They are explained in detail in the following sections. The Receiving RMP returns these protocol-specific fault codes within the Response header element. Reliable Message Faults are carried in the SOAP Header and do not rely exclusively on the SOAP Fault model for the following reasons:

- The SOAP Fault model does not allow batching of several faults in the same message.
- RM Faults may be carried along with business messages that are unrelated to these faults; they should not affect the processing of the SOAP body in such messages.

The rules for processing faults are:

- The Receiving RMP MUST NOT deliver a message for which an RM Fault is published. Therefore, the Receiving RMP MUST NOT send an Acknowledgment Indication for such a message.
- If a Reliable Message sent over a SOAP Request-response MEP cannot be delivered to the Consumer, the response of the SOAP MEP instance SHALL contain a SOAP Fault (in the SOAP Body) in addition to the appropriate RM Fault (in the SOAP Header). If the specific RM Fault encountered was due to a problem with the request header element, the Receiving RMP MUST set the value of the soap:Fault@faultcode attribute to "soap:Client" (for SOAP 1.1 messages) or the soap12:Fault/Code/Value element to "soap12:Sender" (for SOAP 1.2 messages). If the specific RM Fault encountered was due to a problem with processing by the Receiving RMP, the Receiving RMP MUST set the value of the soap:Fault@faultcode attribute to "soap:Server" (for SOAP 1.1 messages) or the soap12:Fault/Code/Value element to "soap12:Receiver" (for SOAP 1.2 messages). The Sending RMP and Producer expect either a complete response or a SOAP Fault when using the SOAP Request-response MEP; this requirement satisfies those expectations. More details are given in **Section** 3.2 and in the HTTP Binding section (**Section** 6).
- When a Reliable Message sent over a SOAP One-way MEP cannot be delivered to the Consumer due to a failure in processing the RM headers, a SOAP Fault SHALL NOT be returned. The HTTP binding section (**Section** 6) gives more details on the recommended behavior in such case.

The Fault codes described in **Sections** 4.5.1 and 4.5.2 are allowed values for @fault in a Response element.

4.5.1 Message Format Faults

The Receiving RMP publishes these faults when the message format of the Reliable Messaging Headers is either invalid or wrong.

Local part name	Description and Cause(s)
InvalidRequest	The Request element is wrong or invalid. Examples are: 1.Any of the mandatory elements such as MessageId, ExpiryTime or ReplyPattern are missing. 2.AckRequested, DuplicateElimination or MessageOrder elements appear twice. 3.The soap:mustUnderstand attribute is missing.
InvalidPollRequest	The PollRequest element is wrong or invalid. Examples are: 1.The soap:mustUnderstand attribute is missing. 2. The RefToMessageIds element is missing.
InvalidMessageId	Used in any of the following cases: 1. @groupId (for MessageId or RefToMessageIds) is not present or is present with an invalid value. 2. @number in SequenceNum element is not present or is present with an invalid value. 3. Attributes (from and to) of SequenceNumRange are not present or are present with invalid values.
InvalidMessageParameters	Used in any of the following cases: 1. The @groupExpiryTime is wrong or invalid. 2. The @groupMaxIdleDuration is wrong or invalid. 3. Both group parameters are present. 4. SequenceNum@last exists but is not one of the allowed {false|true} values.
InvalidReplyPattern	Used in either of the following cases: 1. The ReplyPattern format is wrong or invalid. 2. The ReplyTo element is missing for the Callback pattern.
InvalidExpiryTime	The ExpiryTime format is wrong or invalid.

Table 24 Invalid Message Format Fault Code Values

1061 **Note:**

Cases exist in which the Receiving RMP is unable to send RM Fault Indications for messages with invalid message headers, such as:

- The ReplyTo element is missing or invalid in the Callback and asynchronous Poll cases.
- The MessageId element is missing for the Request element.
- The RefToMessageIds is missing for the PollRequest element.

4.5.2 Message Processing Faults

The Receiving RMP publishes these faults when there is an error processing a valid Reliable Messaging message.

Local part name	**Description and Cause(s)**
FeatureNotSupported	The Receiving RMP receives a message with an RM feature that it does not support. An example is an RM message with a MessageOrder element sent to a Receiving RMP that doesn't support Guaranteed Message Ordering.
PermanentProcessingFailure	Permanent and fatal processing failures such as: 1. Persistence Storage failures. 2. Message Delivery failures. A PermanentProcessingFailure fault indicates that the failure is fatal and subsequent retries of the same message will also fail.
MessageProcessingFailure	Used in transient failure cases such as: 1. The number of buffered requests exceeded the maximum limit. 2. The number of threads reached the maximum limit, etc. 3. The Deliver operation fails. A transient fault, unlike a permanent fault, is temporary; the message may succeed after a subsequent retry.
GroupAborted	All processing for the group associated with the reliable message request has been aborted by the Receiving RMP. The Receiving RMP MUST NOT deliver subsequent messages within that group.

Table 25 Messaging Processing Failure Fault Code Values

4.5.3 RM Fault Examples

Example 8 RM Fault Indication for Reliable Messaging

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="1" to="1" fault="InvalidRequest" />
      </SequenceReplies>
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

If the PollRequest element in **Example 4** was missing the soap:mustUnderstand attribute, the InvalidPollRequest fault may be sent as follows.

Example 9 RM Fault Indication for PollRequest message

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="0" to="5" fault="InvalidPollRequest"/>
        <ReplyRange from="15" to="20" fault="InvalidPollRequest"/>
      </SequenceReplies>
      <NonSequenceReply groupId="mid://20040202.103811@wsr-sender.org"
       fault="InvalidPollRequest"/>
      <SequenceReplies groupId="mid://20040202.103807@wsr-sender.org/">
        <ReplyRange from="713" to="6150" fault="InvalidPollRequest"/>
      </SequenceReplies>
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

4.6 Extensibility Features of Schema

The core schema for this specification (associated in **Section** 1.3, **Table 2**,with the “wsrm” namespace prefix) specifies extension mechanisms for some schema elements.

The following elements (which have a complex sequence type) allow the presence of zero or more extension elements (of type xs:anyType; that is, any type not defined in this core namespace is allowed) at the beginning of the sequence, as well as zero or more extension attributes (with similar namespace restrictions):

- Request

- Response
- PollRequest
- NonSequenceReply
- SequenceReplies
- ReplyRange

The extensibility of the ReplyTo elements (**Sections** 4.2.3.2 and 4.3.1) is somewhat different; it is described in the appropriate sections above.

5 Operational Aspects and Semantics

5.1 Message Group Life Cycle

5.1.1 Group Termination

Being able to know when a group may be terminated and its persistent resources reclaimed is essential for keeping the resource footprint of reliability low. However, this section is not just about efficient management of resources: it describes normative behavioral rules for RMPs when handling group termination.

Termination of a group in the Sending RMP and in the Receiving RMP are two distinct events, not synchronized by any special message but instead occurring as the result of rules applying separately to the Sending and Receiving RMPs. As a consequence, the termination of a group may occur at quite different times on the Sending RMP and the Receiving RMP. However, the lack of synchronization allowed by these termination rules is not consequential.

Groups undergoing termination on the Sending RMP and the Receiving RMP pass through the following states:

Group complete:

- The Sending RMP considers a group complete when all of its messages have been sent and the last sent message has an ending marker (SequenceNum@last="true" or it has a sequence number with the maximum value). Note that completeness occurs even if not all of the group's messages have been either acknowledged or faulted (in case GuaranteedDelivery is enabled).

- The Receiving RMP considers a group complete when a message with an ending marker has been received and all previous messages for this group also have been received (no number missing in the sequence) although not necessarily delivered yet.

Group closed:

- When a group is closed in the Sending RMP, the RMP expects to send no new message in this group. However, the RMP MAY resend messages as needed if GuaranteedDelivery is enabled. If a new message is submitted for a closed group, the Sending RMP MUST notify the Producer that the group is closed and MUST NOT send the message.

- When a group is closed in the Receiving RMP, the RMP expects to receive no new message for this group. After a group is closed and before it is "removed" (see definition below), a Receiving RMP MUST NOT deliver messages received with this group identifier, whether or not they are duplicates of previous messages and regardless of whether they result from a resend of previously failed messages initiated before closing on the Sending RMP (in case GuaranteedDelivery is enabled).

Note:

Due to time-out, a group may be closed without being complete. Once complete, a group will close (see termination rules).

Group Removed:

Group removal occurs at the time the group is closed or afterward. Intuitively, a group is removed when a Receiving RMP does not need to remember anything about this group, i.e., when there is no need to check for duplicates of its messages in the future (for example, when all of its messages have expired).

- When a group is removed in the Sending RMP, the RMP is NOT REQUIRED to verify that future submitted messages are improperly associated with the removed group and MAY treat them as part of a new group. However, the Sending RMP is responsible for generating group identifiers, and it SHOULD generate values unique enough to avoid later reuse of the group identifier of a removed group (for example, generation mechanisms including a timestamp will make reuse impossible).

- When a group is removed in the Receiving RMP, the RMP is no longer supposed to remember anything about this group. In particular, the group identifier is discarded from the RMP state. When receiving a message with same group identifier as a removed group, a Receiving RMP is NOT REQUIRED to confirm whether or not this group identifier value has already been used; the RMP MAY treat such a message as part of a new group.

5.1.2 Group Termination Parameters

Two RM Agreement Items, GroupExpiryTime and GroupMaxIdleDuration, determine when a group can be terminated. These two items are considered Group Termination parameters that control the persistence of the group data. The corresponding message header attributes are @groupExpiryTime and @groupMaxIdleDuration respectively. The following requirements pertain to these header attributes:

a) The first message in a group (the one with Request/MessageId/SequenceNum@number=0) indicates which Group Termination (time-out) parameter is in use for the group. However, the Receiving RMP MUST use the first message received for this group to indicate which termination parameter is associated with this group.

- If the first message in the sequence of a group has neither group time-out parameter present, the group will be terminated according to condition T3, T4 or T5.

- If the first message has one of the two time-out parameters present (either @groupExpiryTime or @groupMaxIdleDuration), the group will be subject to termination rules T1 or T2 described below.

- The Receiving RMP MUST return an InvalidMessageParameters fault if both group persistence parameters are present in any request message.

- If @groupExpiryTime is in use, the Sending RMP MUST NOT send a message in that group with an ExpiryTime value greater than @groupExpiryTime.

b) The group termination parameter sent on the first message in the group SHALL be used on all subsequent messages in that group and SHALL be assigned a value.

c) If the Receiving RMP receives a message with a group termination parameter that is not consistent with the termination parameter used in previous messages for this group, the Receiving RMP MUST return an InvalidMessageParameters fault.

When the group is ordered, the fault SHALL be returned for the message with lowest sequence number that was found inconsistent in the group. If the group is not required to be ordered, the fault SHALL be returned for the first message received that was found inconsistent in the group.

d) The Sending RMP MAY modify either time-out parameter, sending a subsequent message with the new value. When applying termination rules, the Sending RMP MUST use the value in the message with the highest sequence number sent for the group. The Receiving RMP MUST use the value from the message with the highest sequence number received for the group.

e) @groupMaxIdleDuration can be either increased or decreased without restriction. The Sending RMP may increase or decrease @groupExpiryTime as long as it is never less than the max(ExpiryTime) of the messages sent for the group so far.

The Receiving RMP MUST publish an InvalidMessageParameters Fault for a message with a @groupExpiryTime value less than the max(ExpiryTime) of the messages previously received for the group.

5.1.3 Termination Rules

Termination is the process by which an RMP discontinues the use of a group, allowing the RMP to reclaim resources used by the group. Termination typically involves two steps that may occur at different times: closing and removal. Removal of a group may happen some time after it is closed, allowing an RMP to filter out potential duplicate messages. The general rule is that a group is removed once all of its messages have expired. If we define max(ExpiryTime) as the maximum date and time of all ExpiryTime values of the messages sent for a group (on the Sender side) or received for a group (on the Receiver side), a group will not be removed before max(ExpiryTime) occurs.

There are two general indicators an RMP will use to terminate a group:

a) Message Marker: Information within a message (either Request/MessageId/SequenceNum@last=”true” or the maximum sequence number) indicates the last message for the group. This is used by termination rules T3, T4.

b) Timing: Either the group's lifespan expired or its idle time exceeded a time-out. This is used by termination rules T1, T2. Or due to message expiration, a group with the ordering requirement cannot be delivered. This is used by termination rule T5.

These termination rules apply to both ordered and unordered groups. However, these rules do not apply to groups that contain a single message with no sequence number.

5.1.3.1 Termination by expiration (T1):

Context:

The group specified @groupExpiryTime.

Receiver side:

Triggering event: @groupExpiryTime is in the past.

The RMP MUST close and remove the group.

Sender side:

Triggering event: @groupExpiryTime is in the past (note: in this case, max(ExpiryTime) also is past).

The RMP MUST close and remove the group.

5.1.3.2 Termination by idle time-out (T2):

Context:

The group specified @groupMaxIdleDuration.

Receiver side:

Triggering event: The time since the last received message for the group is over @groupMaxIdleDuration.

The RMP MUST close the group. But unlike T1, some of its past messages may not have expired yet. In case Duplicate Elimination is required, the RMP MUST NOT remove the group until max (ExpiryTime) is reached in order to make sure all potential duplicates for the group will not be delivered.

Sender side:

Triggering event: The time since the last sent message for the group is over @groupMaxIdleDuration.

The RMP MUST close the group. If GuaranteedDelivery was required, the RMP MUST remove the group once it has received either acknowledgment or notification of delivery failure for all sent messages. If no GuaranteedDelivery was required, the RMP MUST remove the group immediately.

5.1.3.3 Termination by completeness (T3):

Context:

No specific context.

Receiver side:

Triggering event: The RMP receives a message marked last (Request/MessageId/SequenceNum@last=”true”). If all previous messages for the group have been received, the group is closed immediately. Alternately, the group is closed when the RMP receives the last missing message in the group.

The RMP MUST close the group. However, its removal is done according to T1 or T2 depending on which time-out parameter was specified for the group. If no time-out parameter was specified, the group is removed once all of its messages have expired, i.e., the date and time max (ExpiryTime) has passed.

Note:

In the case in which a message is received with an ending marker before all previous messages have been received, the group remains active. No termination process is initiated yet.

Sender side:

Triggering event: The RMP sends a message marked last.

All messages of the group have been sent. The RMP MUST close the group. If GuaranteedDelivery was required, the RMP MUST remove the group once it has received either acknowledgment or notification of delivery failure for all sent messages. If GuaranteedDelivery was not required, the RMP MUST remove the group immediately.

5.1.3.4 Termination by sequence exhaustion (T4):

Context:

No specific context.

Receiver side:

Triggering event: The RMP receives a message with a sequence number of the maximum value. If all previous messages for the group have been received, the group is closed immediately. Alternately, the group is closed when the RMP receives the last missing message in the group.

The group closing and removal follow the rules in T3, the message with the maximum sequence number acting as a message with the ending mark.

Note:

In case a message is received with the maximum sequence number before all previous messages have been received, the group remains active. No termination process is initiated yet.

Sender side:

Triggering event: The RMP sends a message with a sequence number with the maximum value.

The group closing and removal follow the rules in T3, the message with the maximum sequence number acting as a message with the ending mark.

5.1.3.5 Termination by ordering failure (T5):

Context:

The group requires the Guaranteed Message Ordering reliability feature.

Receiving side:

Triggering event: In an ordered group, a received message expires before delivery or faults with a fault code other than MessageProcessingFailure. If all previous messages for the group have been received, the group is closed immediately. Alternately, the group is closed when the RMP receives the last missing message in the group.

The RMP MUST close the group. The group is removed according to rule T3.

Sender Side:

Triggering event: In an ordered group, an unacknowledged message expires or the RMP receives an RM Fault for this Reliable Message with a fault code other than MessageProcessingFailure.

The RMP MUST close the group. The group is removed according to rule T3.

5.1.3.6 Summary of Group Termination Rules

Conditions for terminating a group in a Receiving RMP:

Group Closing	***Group Removal***
When @groupExpiryTime has passed.	(after closing) When @groupExpiryTime has passed.
When the @groupMaxIdleDuration time-out has expired.	(after closing) When Max(ExpiryTime) has passed.
When a group is complete.	(after closing) When Max(ExpiryTime) has passed.
When a group is ordered AND an undelivered message expires or faults.	(after closing) When Max(ExpiryTime) has passed.

Table 26 Conditions for terminating a group – Receiving RMP

Conditions for terminating a group in a Sending RMP:

Group Closing	***Group Removal***
When @groupExpiryTime has passed.	(after closing) When @groupExpiryTime has passed.
When the @groupMaxIdleDuration time-out has expired.	(after closing) In case GuaranteedDelivery is not required, remove the group immediately. Otherwise, remove it if all messages have been either acknowledged or faulted.
When a group is complete.	(after closing) In case GuaranteedDelivery is not required, remove the group immediately. Otherwise, remove it if all messages have been either acknowledged or faulted.
When a group is ordered AND an unacknowledged message expires or faults.	(after closing) Remove the group after all messages have been either acknowledged or faulted.

Table 27 Conditions for terminating a group – Sending RMP

5.2 Attachments

When an RMP implementing this specification uses the W3C Note “SOAP Messages with Attachments” specification [SOAP with Attachments], it MUST follow the following rules:

1) The Sending RMP MUST include the whole SOAP envelope containing the WS-Reliability header elements in the first MIME part.

2) It MUST set the charset parameter of the Content-Type header of the first MIME part to either UTF-8 or UTF-16.

3) It MAY include zero or more additional MIME parts in a Reliable Message.

4) The Receiving RMP MUST deliver all MIME parts in a Reliable Message to the Consumer.

6 HTTP Binding

This section specifies two normative bindings of WS-Reliability header elements to SOAP header blocks carried in messages using HTTP as a transport protocol:

- SOAP 1.1 over HTTP POST binding: An implementation of WS-Reliability MAY support mapping the WS-Reliability header elements as SOAP header blocks in accordance with the SOAP 1.1 HTTP Binding specified in Section 6 of [SOAP 1.1]. In that case, the SOAP Request-response MEP defined in this specification will map to an HTTP request-response. The SOAP One-way MEP, as defined in **Section** 2.3, maps to the request of an HTTP request-response.

- SOAP 1.2 over HTTP POST binding: An implementation of WS-Reliability MAY support mapping the WS-Reliability header elements as SOAP header blocks in accordance with the SOAP 1.2 HTTP binding for the Request-Response MEP specified in Section 7, “SOAP HTTP Binding”, of [SOAP 1.2 Part 2].

If a Reliable Message request is invoked using SOAP 1.1, all subsequent message exchanges pertaining to that Message Identifier MUST use the SOAP 1.1 protocol. In addition, when an HTTP binding is used, it is RECOMMENDED the RMP comply with WS-I BP 1.1 [WS-I BP 1.1]. When no WSDL describes the messages being exchanged, the previous WS-I conformance requirements should be understood as conformance to the subset of the profile requirements pertaining to the message artifact only.

In case a message encounters a failure in processing the RM headers, the requirements for Fault handling in **Section** 4.5 apply. When using SOAP 1.1, conformance to the WS-I Basic Profile 1.1 requires the following:

- For SOAP One-way HTTP binding: the HTTP response entity-body SHALL be empty. If the RM Fault is a Message Format fault, the HTTP status code SHOULD be “400 Bad Request" (see R1113 in [WS-I BP 1.1]); otherwise, the RM fault is a Message Processing fault and the status code SHOULD be "500 Internal Server Error".

- For SOAP Request-response HTTP binding: the HTTP response contains a SOAP Fault element and has the "500 Internal Server Error" HTTP status code (see R1126 in [WS-I BP 1.1]).

These two requirements for Fault handling apply to all message exchanges described in this section and its sub-sections.

If a ReplyTo element present in a Request element or Poll Request header element sent using the SOAP 1.1 protocol uses the wsrm:BareURI (the default, described in **Sections** 4.2.3.2.2 and 4.3.1.2) reference scheme and uses the 'http:' URL scheme, the Receiving RMP MUST send the WS-Reliability response using the HTTP binding specified in Section 6 of SOAP 1.1.

If a Reliable Message request is invoked using SOAP 1.2, all subsequent message exchanges pertaining to its Message Identifier MUST use the SOAP 1.2 protocol.

If a ReplyTo element present in a Request element or Poll Request header element sent using the SOAP 1.2 protocol uses the wsrm:BareURI reference scheme and uses the 'http:' URL scheme, the the Receiving RMP MUST send the WS-Reliability response using the HTTP binding for Request-Response MEP specified in SOAP 1.2.

The following subsections specify the mapping of WS-Reliability header elements to HTTP request and response messages for the three RM-Reply Patterns. The Poll RM-Reply Pattern has two variations: synchronous and asynchronous.

The value of the ReplyPattern/Value element identifies the specific RM-Reply Pattern in use (see **Section** 4.2.3.1 for details).

This specification requires the transport layer to deliver messages to the reliability layer without corruption. When a request message contains the AckRequested element, the Receiving RMP MUST send an RM-Reply (an Acknowledgment Indication or an RM Fault Indication) for that request. For the Callback and Poll RM-Reply Patterns, a Response element can contain multiple Acknowledgment and/or RM Fault Indications.

For simplicity, the detailed examples show only the use of SOAP 1.1. However, the figures that show the mapping of WS-Reliability elements to HTTP POST request messages and HTTP response messages apply to both the SOAP 1.1 over HTTP POST binding and the SOAP 1.2 over HTTP POST binding.

6.1 Reliable Messaging with Response RM-Reply Pattern

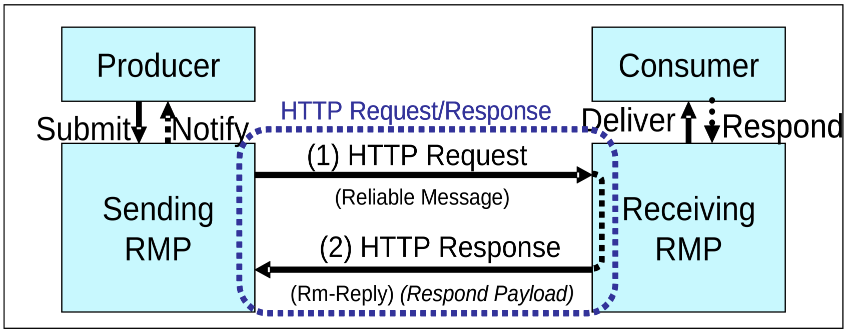

Figure 9 Response RM-Reply Pattern

As described in general for this RM-Reply Pattern (**Section** 2.4.1), the Receiving RMP MUST return the RM-Reply with the HTTP response on the same HTTP connection used by the Sending RMP to send the request. This is illustrated in **Figure 9.**

- In (1), the Sending RMP initiates an HTTP connection and sends a Message using the HTTP POST method, as in **Example 10**.

- In (2), using the same connection, the Receiving RMP sends back to the Sending RMP an HTTP response containing an RM-Reply; in **Example 11**, the RM-Reply is an Acknowledgment Indication.

Example 10 Request Message with Response RM-Reply Pattern

```
POST /abc/servlet/wsrEndpoint HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.100
SOAPAction: ""
Content-Length: 755

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <Request
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
     soap:mustUnderstand="1">
      <MessageId groupId="mid://20040202.103832@wsr-sender.org">
        <SequenceNum number="0"
         groupExpiryTime="2005-02-02T03:00:33-31:00" />
      </MessageId>
      <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
      <ReplyPattern>
        <Value>Response</Value>
      </ReplyPattern>
      <AckRequested/>
      <DuplicateElimination/>
      <MessageOrder/>
    </Request>
  </soap:Header>
  <soap:Body>
    <Request xmlns="http://example.org/wsr">Request Message</Request>
  </soap:Body>
</soap:Envelope>
```

Example 11 Acknowledgment Indication with Response RM-Reply Pattern

```
HTTP/1.0 200 OK
Server: WS-ReliabilityServer
Date: Mon, 02 Feb 2004 10:38:32 GMT
Content-Language: en
Content-Type: text/xml; charset=utf-8
Content-Length: 414

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="0" to="0"/>
      </SequenceReplies>
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

6.2 Reliable Messaging with Callback RM-Reply Pattern

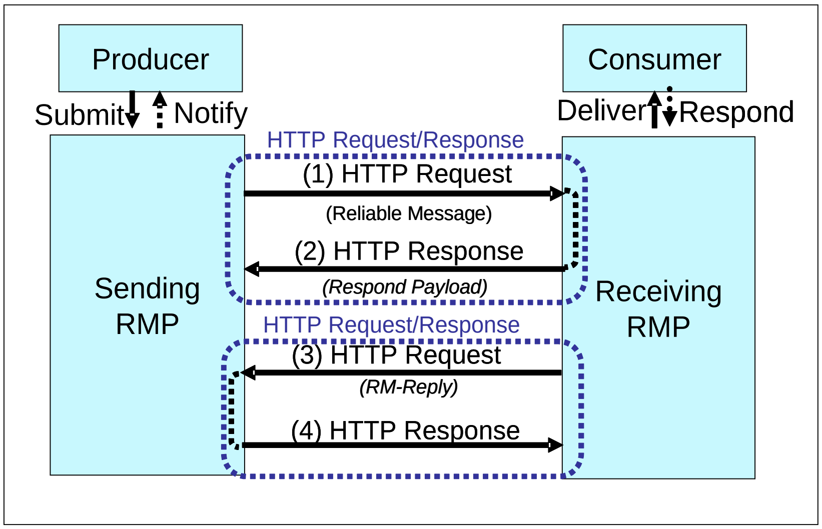

Figure 10 Callback RM-Reply Pattern

As described in general for this RM-Reply Pattern (**Section** 2.4.2) and as illustrated in **Figure 10**, two distinct HTTP request/response exchanges are involved.

- In (1), the Sending RMP initiates a new HTTP request and sends a Reliable Message with the Callback RM Reply Pattern. **Example 12** shows such an HTTP message.
- In (2), the HTTP response may have an empty entity-body (in case of a SOAP One-way MEP instance).
- In (3), the Receiving RMP MUST return the RM-Reply on an HTTP connection different from the one the Sending RMP used to send the message. The direction of the HTTP

connection used by the Receiving RMP is from the Receiving RMP to the Sending RMP. **Example 14** shows an Acknowledgment Indication as the RM-Reply.

- In (4), there is no HTTP entity-body unless the RM-Reply was bundled with a new Reliable Message on a SOAP Request-response MEP instance.

Example 12 Request Message with Callback RM-Reply Pattern

```
POST /abc/servlet/wsrEndpoint HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.100
SOAPAction: ""
Content-Length: 863

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <Request
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
     soap:mustUnderstand="1">
      <MessageId groupId="mid://20040202.103832@wsr-sender.org">
        <SequenceNum number="0"
         groupExpiryTime="2005-02-02T03:00:33-31:00" />
      </MessageId>
      <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
      <ReplyPattern>
        <Value>Callback</Value>
        <ReplyTo>
          <BareURI>http://wsr-sender.org/abc/wsrmListener</BareURI>
        </ReplyTo>
      </ReplyPattern>
      <AckRequested/>
      <DuplicateElimination/>
      <MessageOrder/>
    </Request>
  </soap:Header>
  <soap:Body>
    <Request xmlns="http://example.org/wsr">Request Message</Request>
  </soap:Body>
</soap:Envelope>
```

Example 13 HTTP response with no content

```
HTTP/1.0 200 OK
Server: WS-ReliabilityServer
Date: Mon, 02 Feb 2004 10:38:32 GMT
Content-Language: en
Content-Type: text/xml; charset=utf-8
Content-Length: 0
```

Example 14 Acknowledgment Indication with Callback RM-Reply Pattern

```
POST /abc/wsrmListener HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.200
SOAPAction: ""
Content-Length: 414

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="0" to="0"/>
      </SequenceReplies >
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

6.3 Reliable Messaging with Poll RM-Reply Pattern

The general rules for this RM-Reply Pattern are described in **Section** 2.4.3. When the Sending RMP issues a PollRequest, the Receiving RMP MAY return the RM-Reply on the HTTP connection used to send the PollRequest message (synchronous), or it MAY return the RM-Reply on a different HTTP connection (asynchronous). Whether the RM-Reply corresponding to the PollRequest is synchronous or asynchronous depends on the presence of a ReplyTo element in the PollRequest element.

6.3.1 Synchronous Poll RM-Reply Pattern

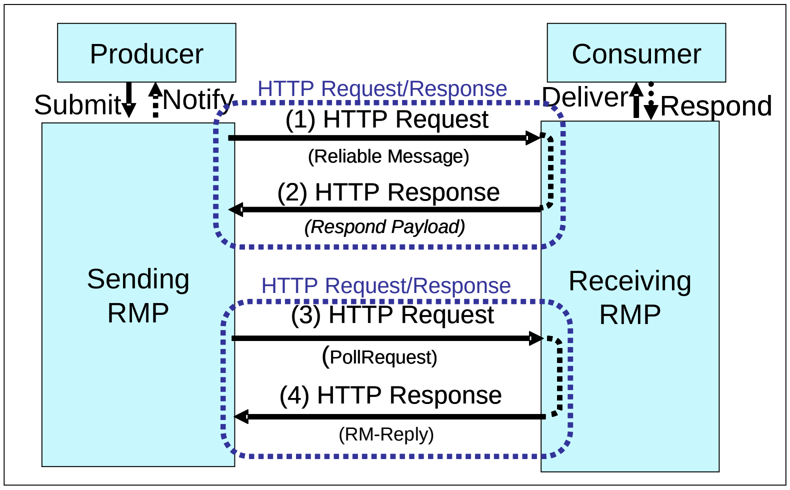

Figure 11 Synchronous Poll RM-Reply Pattern

Figure 11 illustrates the synchronous variant of the Poll RM Reply Pattern.

- In (1), the Sending RMP initiates a new HTTP Request and sends a Reliable Message with the Poll RM-Reply Pattern.

- In (2), the HTTP response may have an empty entity-body (in case of a SOAP One-way MEP instance).
- In (3), at a later time the Sending RMP initiates a different HTTP Request to send a PollRequest message. The PollRequest does not include the ReplyTo element (see **Example 15**).
- In (4), the Receiving RMP returns the RM-Reply in an HTTP response on the same HTTP connection used to send the PollRequest, as illustrated in **Figure 11**. The HTTP response (4) includes an RM-Reply (e.g., an Acknowledgment Indication as in **Example 16**).

Example 15 PollRequest message with Synchronous Poll RM-Reply Pattern

```
POST /abc/servlet/wsrmListener HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.100
SOAPAction: ""
Content-Length: 433

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <PollRequest
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
     soap:mustUnderstand="1">
      <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
        <SequenceNumRange from="0" to="20"/>
      </RefToMessageIds>
    </PollRequest>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

Example 16 Synchronous Acknowledgment Indication

```
HTTP/1.0 200 OK
Server: WS-ReliabilityServer
Date: Mon, 02 Feb 2004 10:38:32 GMT
Content-Language: en
Content-Type: text/xml; charset=utf-8
Content-Length: 456

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="0" to="14"/>
        <ReplyRange from="16" to="20"/>
      </SequenceReplies>
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

6.3.2 Asynchronous Poll RM-Reply Pattern

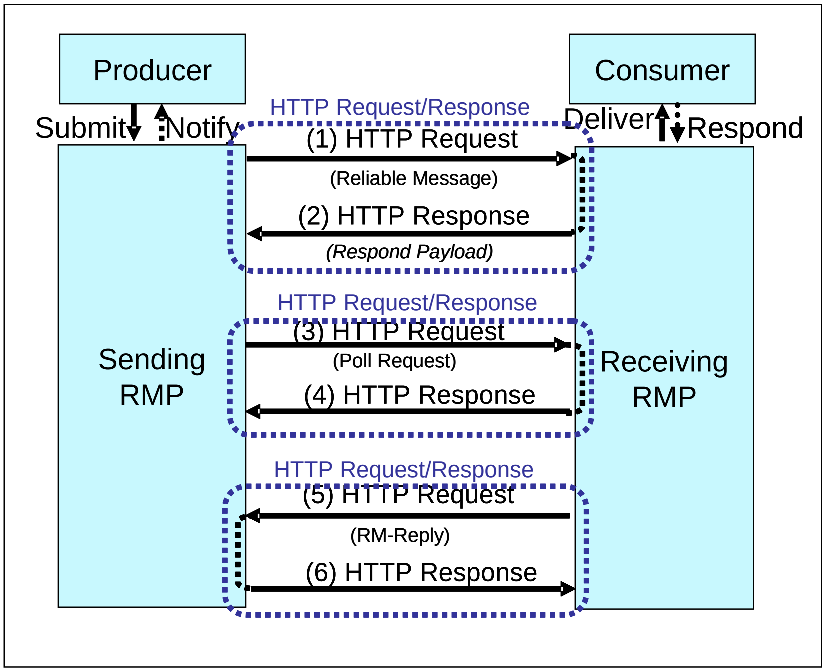

Figure 12 Asynchronous Poll RM-Reply Pattern

Figure 12 illustrates the asynchronous variant of the Poll RM Reply Pattern.

- In (1), the Sending RMP initiates a new HTTP Request and sends a Reliable Message with the Poll RM-Reply Pattern.

- In (2), the HTTP response may have an empty entity-body (in the case of a SOAP One-way MEP instance).
- In (3)**,** the Sending RMP initiates a new HTTP request and sends a PollRequest message. Note that in **Example 17**, the PollRequest element has a ReplyTo element.
- In (4), the HTTP response (4) has no HTTP entity-body (see **Example 13**).
- In (5), the Receiving RMP sends the RM-Reply in a different HTTP request to the listener identified by the ReplyTo element (see **Example 18**).
- In (6), the HTTP response has no HTTP entity-body (see **Example 13**).

Example 17 PollRequest message with Asynchronous Poll RM-Reply Pattern

```
POST /abc/servlet/wsrmListener HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.100
SOAPAction: ""
Content-Length: 553

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <PollRequest
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd"
     soap:mustUnderstand="1">
      <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
        <SequenceNumRange from="0" to="20"/>
      </RefToMessageIds>
      <ReplyTo>
        <BareURI>http://wsr-sender.org/xyz/servlet/wsrmListener
        </BareURI>
      </ReplyTo>
    </PollRequest>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

Example 18 Asynchronous Acknowledgment Indication

```
POST /xyz/servlet/wsrmListener HTTP/1.0
Content-Type: text/xml; charset=utf-8
Host: 192.168.183.200
SOAPAction: ""
Content-Length: 456

<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
  <soap:Header>
    <Response soap:mustUnderstand="1"
     xmlns="http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd">
      <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
        <ReplyRange from="0" to="14"/>
        <ReplyRange from="16" to="20"/>
      </SequenceReplies>
    </Response>
  </soap:Header>
  <soap:Body />
</soap:Envelope>
```

7 Conformance

In order to conform to this specification, an implementation must satisfy all of the following conditions:

- It has implemented all required syntax, features and behaviors.
- It complies with the following interpretation of the keywords OPTIONAL and MAY: as stated in [RFC2119], when these keywords apply to the behavior of the implementation, the implementation is free to support these behaviors or not.
- It MUST be capable of processing the prescribed failure mechanism for those optional features it has chosen to implement. If an RMP conforming to this requirement has implemented an optional feature, syntax or behavior defined in this specification, it can interoperate with another implementation that has not.
- It MUST be capable of generating the prescribed failure mechanism for those optional features it has not chosen to implement. If an RMP conforming to this requirement has not implemented an optional feature, syntax or behavior defined in this specification, it can interoperate with another implementation that has.

8 References

[ebMS] "Message Service Specification Version 2.0", OASIS ebXML Messaging Services Technical Committee, OASIS Standard, 1 April 2002. Available at

http://www.ebxml.org/specs/ebMS2.pdf

[RFC1738] "Uniform Resource Locators (URL)", T. Berners-Lee et al, RFC 1738, IESG and IETF, December 1994. Available at

http://www.ietf.org/rfc/rfc1738.txt

[RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, Bradner, S., IESG and IETF, March 1997. Available at

http://www.ietf.org/rfc/rfc2119.txt

[RFC2392] “Content-ID and Message-ID Uniform Resource Locators”, RFC2392, E. Levinson, IESG and IETF, August 1998. Available at

http://www.ietf.org/rfc/rfc2392.txt

[RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, Tim Berners-Lee et al, IESG and IETF, August 1998. Available at

http://www.ietf.org/rfc/rfc2396.txt

[RFC2616] "Hypertext Transfer Protocol – HTTP/1.1", RFC 2616, R. Fielding et al, IESG and IETF, June 1999. Available at

http://www.ietf.org/rfc/rfc2616.txt

[RFC2822] "Internet Message Format", RFC 2822, P. Resnick, Editor, IESG and IETF, April 2001. Available at

http://www.ietf.org/rfc/rfc2822.txt

[SOAP 1.1] "Simple Object Access Protocol (SOAP) 1.1", Don Box et al, W3C Note, 8 May, 2000. Available at

http://www.w3.org/TR/2000/NOTE-SOAP-20000508/

[SOAP 1.2 Part 1] "SOAP 1.2 Part 1: Messaging Framework", Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, eds., W3C Recommendation, 24 June 2003. Available at

http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[SOAP 1.2 Part 2] "SOAP 1.2 Part 1: Adjuncts", Martin Gudgin, Marc Hadley, Noah Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, eds., W3C Recommendation, 24 June 2003. Available at

http://www.w3.org/TR/2003/REC-soap12-part2-20030624/

[SOAP with Attachments] "SOAP Messages with Attachments", John J. Barton, Satish Thatte, Henrik Frystyk Nielsen, W3C Note, 11 December 2000, Available at

http://www.w3.org/TR/SOAP-attachments

[XML] "Extensible Markup Language (XML) 1.0 (Third Edition)", Tim Bray et al, eds., W3C Recommendation, first published 10 February 1998, revised 4 February 2004. Available at

http://www.w3.org/TR/2004/REC-xml-20040204

[XML Namespaces] "Namespaces in XML", Tim Bray et al., eds., W3C Recommendation, 14 January 1999. Available at

http://www.w3.org/TR/1999/REC-xml-names-19990114/

[XML Schema Part 1] "XML Schema Part 1: Structures", Henry S. Thompson, David Beech, Murray Maloney, Noah Mendelsohn, eds., W3C Recommendation, 2 May 2001. Available at

http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[XML Schema Part 2] "XML Schema Part 2: Datatypes", Paul V. Biron and Ashok Malhotra, eds. W3C Recommendation, 2 May 2001. Available at

http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

[XPath 1.0] "XML Path Language (XPath) Version 1.0", James Clark, Steve DeRose, eds., W3C Recommendation, 16 November 1999. Available at

http://www.w3.org/TR/1999/REC-xpath-19991116

[WSDL 1.1] "Web Services Description Language (WSDL) 1.1", Erik Christensen, Francisco Curbera, Greg Meredith, Sanjiva Weerawarana, eds., W3C Note, 15 March 2001. Available at

http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[WS-I BP 1.1] "Basic Profile Version 1.1", Keith Ballinger, David Ehnebuske, Christopher Ferris, Martin Gudgin, Mark Nottingham, Prasad Yendluri, eds., WS-I specification, 8 August 2004. Available at

http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-07-21.html

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

http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf

Appendix A.Schema (Normative)

The schemas for this specification have the following URLs and are located using the filenames shown in the table:

Schema Namespace URL	File name	Prefix
http://docs.oasis-open.org/wsrm/2004/06/ws-reliability-1.1.xsd	ws-reliability-1.1.xsd	wsrm
http://docs.oasis-open.org/wsrm/2004/06/reference-1.1.xsd	reference-1.1.xsd	ref
http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd	fnp-1.1.xsd	fnp
http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd	wsrmfp-1.1.xsd	wsrmfp

Table 28 WS-Reliability Schema Prefixes

RMPs MUST include the SOAP mustUnderstand attribute (defined in the same namespace used for the soap:Envelope element) in all Reliable Messaging specified header blocks and MUST observe the following restrictions:

- For SOAP 1.1, the mustUnderstand attribute value is restricted to "1".
- For SOAP 1.2, the mustUnderstand attribute value is restricted to "1" or "true".

Appendix B.WS-Reliability Features, Properties and Compositors (Normative and Optional)

B.1. Introduction

Users of a Web Service need to be aware of the reliability capabilities (RM capabilities) the service supports or requires. One practical location to advertise these capabilities is in the service description (WSDL document), which allows publishing both abstract service definitions and concrete protocol details (bindings). This allows clients (including other Web services) to easily obtain information about specific capabilities (such as guaranteed delivery, duplicate elimination, message ordering, and the supported reply patterns) of a specific Web service before calling the service. While bundling RM capabilities with the service description may not be desirable in all cases, this convenient approach often should be appropriate. The WSDL annotation mechanism described here adds such capability assertions in a flexible way.

WS-Reliability uses the WSDL 1.1 extensibility points to define an extensible framework consisting of features, properties and compositors. This framework addresses the needs of a reliable Web service to advertise its capabilities and the composability of those capabilities.

The following extensibility elements are relevant to RM capabilities:

- **feature** – see **Appendix** B.3.2.
- **property** – see **Appendix** B.3.3.
- **compositor** – see **Appendix** B.3.1.

An annotation composed with the above extensibility elements will specify the reliability features and properties associated with specific WSDL constructs. Features and properties represent RM capabilities; compositors specify how these capabilities are composed.

This would, for example, allow a Web service description to advertise that clients invoking the service must use duplicate elimination or message ordering.

B.2. Conformance

Implementations of WS-Reliability are expected (though not required) to understand the WSDL extensibility points defined in this section.

Understanding these extensibility points promotes interoperability: a service advertises its supported and required features when its WSDL document contains these extensibility points. Therefore it is RECOMMENDED that implementations recognize, understand and support these extensibility points.

It is also possible for services to advertise features through other channels (such as UDDI) in addition to these extensibility points.

B.3. WSDL Extensibility Elements

B.3.1. Compositor

The compositor semantics describe how features and properties are composed for the enclosing component (or WSDL 1.1 element). The compositor's semantics determine whether the usage of composed elements by a client to the service is required or optional. All of the RM capabilities represented by these elements must be supported by the service. A compositor element can occur as a child element of wsdl11:portType, wsdl11:operation (which itself may be a child of wsdl11:portType or wsdl11:binding), wsdl11:binding, wsdl11:service and wsdl11:port. The compositor element uses the extensibility defined by WSDL 1.1. A compositor element specifies the semantics for combining its children elements. These children elements can be additional compositors, features, properties or extensibility elements.

A compositor element is expressed by the following pseudo-syntax:

```
<fnp:compositor uri="..." name="NCName"?>
[fnp:feature/> | <fnp:property/> | <fnp:compositor/> |
    <extensibility-element/>]+
</fnp:compositor>
```

The uri attribute of the compositor specifies its semantics. Four different compositors (URIs) and their capability-related semantics are described below. It is possible to provide additional compositors by using other URIs. The possibility of additional compositors and the existence of extensibility points (represented by "<extensibility-element>") make the framework extensible. The optional @name identifies the compositor. An element built with such compositors represents an RM capability.

- **all:** this compositor specifies that a service invocation MUST comply with all of the children elements representing RM capability assertions. This compositor is identified by the URI:

  http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositors/all

- **choice:** this compositor specifies that a service invocation MUST comply with exactly one of the possibly many children elements representing RM capability assertions. This compositor is identified by the URI:

  http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositors/choice

- **one-or-more:** this compositor specifies that a service invocation MUST comply with at least one of the possibly many children elements representing RM capability assertions. This compositor is identified by the URI:

  http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositors/one-or-more

- **zero-or-more:** this compositor specifies that a service invocation MAY comply with one or more of the children elements representing RM capability assertions. This compositor is identified by the URI:

  http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositors/zero-or-more

Examples for each compositor are provided in **Appendix** B.7 below.

Compositors specified at different WSDL components are implicitly aggregated using the 'all' compositor at the dependent WSDL component. Consider the example below:

```
<wsdl11:definitions>
  ...
  <wsdl11:portType name="myPortType">
    <fnp:compositor uri="..." name="A">
      ...
    </fnp:compositor>
    ...
  </wsdl11:portType>
  <wsdl11:binding name="myBinding" type="myPortType">
    <fnp:compositor uri="..." name="B">
      ...
    </fnp:compositor>
    ...
  <wsdl11:binding>
  <wsdl11:service name="myService">
    <wsdl11:port name="myPort" binding="myBinding>
      ...
    </wsdl11:port>
  </wsdl11:service>
<wsdl11:definitions>
```

The compositor specified at the wsdl11:portType "myPortType" and the compositor specified at wsdl11:binding "myBinding" are aggregated at the dependent wsdl11:port "myPort" using the 'all' compositor. The equivalent compositor at "myPort" is

```
<fnp:compositor
  uri="http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/all">
  <fnp:compositor uri="..." name="A">
  </fnp:compositor>
  <fnp:compositor uri="..." name="B">
    ...
  </fnp:compositor>
</fnp:compositor>
```

B.3.2. Feature

A feature describes an abstract RM capability or assertion associated with a WSDL element. A feature can occur only as a child of a compositor.

The enclosing compositor(s) define(s) whether or not the usage of a feature is required. A feature is identified by a URI. Recognizing the URI of a feature implies understanding the feature identified by that URI.

A feature element is expressed by the following pseudo-syntax:

```
<fnp:feature uri="...">
   [<fnp:compositor/> | <extensibility-element/>]*
</fnp:feature>
```

B.3.3. Property

A property is identified by a QName. A property is an assertion or constraint on a specific RM capability and its value(s). A property can occur only as a child of a compositor.

Typically, properties are (but are not required to be) associated with a feature and are described in a feature specification. The QName identifier of a property uniquely identifies the property. Recognizing the property QName identifier implies understanding the semantics associated with that property. The property QName identifier typically points to a global XML Schema element declaration. A property specification typically specifies the schema containing this global element declaration. There may be a constraint on the set of values a property can have; such a constraint is specified by a QName identifying an XML Schema type.

```
<fnp:property name="xs:QName">
   [<fnp:value>xs:anyType</fnp:value> |
       <fnp:constraint>xs:QName</fnp:constraint>]
   [<extensibility-element/>]*
</fnp:property>
```

B.4. WS-Reliability Feature

The WS-Reliability feature is identified by the URI

http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd

This feature URI identifies the WS-Reliability specification. Understanding this URI implies understanding the WS-Reliability specification.

B.5. WS-Reliability Properties

This section identifies properties for the WS-Reliability specification. Typically these properties are scoped within the feature identified by the URI

http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd

B.5.1. Guaranteed Delivery Property

This property is identified by the QName "wsrmfp:GuaranteedDelivery" and corresponds to the semantics specified by the WS-Reliability guaranteed delivery semantics. The type of this property is "xs:boolean".

B.5.2. Duplicate Elimination Property

This property is identified by the QName "wsrmfp:NoDuplicateDelivery" and corresponds to the semantics specified by the WS-Reliability duplicate elimination semantics. The type of this property is "xs:boolean".

B.5.3. Message Ordering Property

This property is identified by the QName "wsrmfp:OrderedDelivery" and corresponds to the semantics specified by the WS-Reliability message ordering semantics. The type of this property is "xs:boolean".

B.5.4. Reply Pattern Property

This property is identified by the QName "wsrmfp:ReplyPattern" and corresponds to the semantics specified by the WS-Reliability reply pattern options. The type of this property is "xs:string". (values: Response, Poll, Callback)

B.6. Compositor Examples

B.6.1. Example for the "all" compositor

```
<wsdl11:portType name="Example-1">
  <fnp:compositor
   uri="http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/all">
    <fnp:feature
     uri="http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd"
      <fnp:compositor uri=
        "http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/all">
        <fnp:property name="wsrmfp:NoDuplicateDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
        <fnp:property name="wsrmfp:OrderedDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
        <fnp:property name="wsrmfp:GuaranteedDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
      </fnp:compositor>
    </fnp:feature>
  </fnp:compositor>
  ...
</wsdl11:portType>
```

In the example above, the reliability feature identified by URI "http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd" is required by the portType. This feature consists of three properties, all of which are required because of the semantics of the 'all' compositor that composes the three properties.

B.6.2. Example for the "choice" compositor:

```
<wsdl11:binding name="Example-2">
  <fnp:compositor
   uri="http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/all">
    <fnp:feature
     uri="http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd"
      <fnp:compositor uri=
     "http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositors/choice">
        <fnp:property name="wsrmfp:ReplyPattern">
          <value>Response</value>
        </fnp:property>
        <fnp:property name="wsrmfp:ReplyPattern">
          <value>Callback</value>
        </fnp:property>
        <fnp:property name="wsrmfp:ReplyPattern">
          <value>Poll</value>
        </fnp:property>
      </fnp:compositor>
    </fnp:feature>
  </fnp:compositor>
  ...
</wsdl11:binding>
```

In the example above, the reliability feature identified by URI "http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd" is required by the portType. This feature consists of three properties composed by the 'choice' compositor; the client must choose one.

B.6.3. Example for the "one-or-more" compositor:

```
<wsdl11:portType name="Example-3">
  <fnp:compositor
   uri="http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/all">
    <fnp:feature
     uri="http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd" >
      <fnp:compositor uri=
  "http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/one-or-more">
        <fnp:property name="wsrmfp:NoDuplicateDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
        <fnp:property name="wsrmfp:OrderedDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
        <fnp:property name="wsrmfp:GuaranteedDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
      </fnp:compositor>
    </fnp:feature>
  </fnp:compositor>
  ...
</wsdl11:portType>
```

B.6.4. Example for the "zero-or-more" compositor:

```
<wsdl11:portType name="Example-4">
  <fnp:compositor
   uri="http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/all">
    <fnp:feature
     uri="http://docs.oasis-open.org/wsrm/2004/06/wsrmfp-1.1.xsd"
      <fnp:compositor uri=
 "http://docs.oasis-open.org/wsrm/2004/06/fnp-1.1.xsd/compositor/zero-or-more">
        <fnp:property name="wsrmfp:NoDuplicateDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
        <fnp:property name="wsrmfp:OrderedDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
        <fnp:property name="wsrmfp:GuaranteedDelivery">
          <fnp:value>true</fnp:value>
        </fnp:property>
      </fnp:compositor>
    </fnp:feature>
  </fnp:compositor>
  ...
</wsdl11:portType>
```

Appendix C.Acknowledgments

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

- David Ingham, Arjuna Technologies Limited
- Joseph Chiusano, Booz Allen Hamilton
- Peter Furniss, Choreology Ltd
- Jeff Turpin, Cyclone Commerce
- Pramila Mullan, France Telecom
- Jacques Durand, Fujitsu
- Kazunori Iwasa (Secretary), Fujitsu
- Tom Rutt (Chair), Fujitsu
- Jishnu Mukerji, Hewlett-Packard
- Robert Freund, Hitachi
- Eisaku Nishiyama, Hitachi
- Nobuyuki Yamamoto, Hitachi
- Ben Bloch, Individual
- Mark Hansen, Individual
- Paolo Romano, Individual
- Dock Allen, Mitre Corporation
- Junichi Tatemura, NEC Corporation
- Alan Weissberger, NEC Corporation
- Magdolna Gerendai, Nokia
- Szabolcs Payrits, Nokia
- Mark Peel, Novell
- Sunil Kunisetty (Secretary), Oracle
- Anish Karmarkar, Oracle
- Jeff Mischkinsky, Oracle
- Marc Goodner (Secretary), SAP
- Pete Wenzel, SeeBeyond Technology Corporation
- Doug Bunting (Secretary), Sun Microsystems

- Tony Graham, Sun Microsystems
- Chi-Yuen Ng, University of Hong Kong
- Patrick Yee, University of Hong Kong
- Prasad Yendluri, webMethods, Inc.
- Scott Werden, WRQ, Inc.

And the following people made contributions to produce Ver 1.0 of this specification:

Colleen Evans, Sonic Software Corporation / Dave Chappell, Sonic Software Corporation / Doug Bunting, Sun Microsystems, Inc. / George Tharakan, Sun Microsystems, Inc. / Hisashi Shimamura, NEC Corporation / Jacques Durand, Fujitsu Software Corporation / Jeff Mischkinsky, Oracle Corporation / Katsutoshi Nihei, NEC Corporation / Kazunori Iwasa, Fujitsu Limited / Martin Chapman, Oracle Corporation / Masayoshi Shimamura, Fujitsu Limited / Nicholas Kassem, Sun Microsystems, Inc. / Nobuyuki Yamamoto, Hitachi Limited / Sunil Kunisetty, Oracle Corporation / Tetsuya Hashimoto, Hitachi Limited / Tom Rutt, Fujitsu Software Corporation / Yoshihide Nomura, Fujitsu Limited / Akira Ochi, Fujitsu Limited / Hirotaka Hara, Fujitsu Limited / Hiroyuki Tomisawa, Hitachi Limited / Katsuhisa Nakazato, Fujitsu Limited / Masahiko Narita, Fujitsu Limited / Nobuyuki Saji, NEC Corporation / Shuichi Imabayashi, Fujitsu Limited

Appendix D.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.

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.