



Web Services Reliable Messaging TC WS-Reliability 1.1

OASIS Standard, 15 November 2004

Document identifier:

wsmr-ws_reliability-v1.1-spec-os

Location:

<http://docs.oasis-open.org/wsmr/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 wsmr@lists.oasis-open.org list. Others should use the comment form at http://www.oasis-open.org/committees/comments/form.php?wg_abbrev=wsmr.

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/wsmr/>).

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

34 Table of Contents

35	1	Introduction.....	6
36	1.1	Purpose of WS-Reliability.....	6
37	1.2	Definition and Scope of Reliable Messaging.....	6
38	1.3	Notational Conventions.....	7
39	1.4	Relation to Other Specifications.....	8
40	1.5	Terminology.....	9
41	2	Messaging Model.....	11
42	2.1	Messaging Context.....	11
43	2.2	RMP Operations and Their Invocation.....	11
44	2.2.1	Binding between WSDL Operation Types and RMP Invocations.....	12
45	2.3	Assumed SOAP Message Exchange Patterns.....	12
46	2.4	Message Reply Patterns.....	13
47	2.4.1	Response RM-Reply Pattern.....	13
48	2.4.2	Callback RM-Reply Pattern.....	14
49	2.4.3	Poll RM-Reply Pattern.....	14
50	2.5	Message Identification and Grouping.....	15
51	3	Reliability Agreement and Features.....	16
52	3.1	RM Agreement.....	16
53	3.1.1	Definition.....	16
54	3.1.2	RM Agreement Items.....	16
55	3.1.3	Scope of an Agreement Item.....	17
56	3.1.4	Rules.....	18
57	3.1.5	Creation, Representation and Deployment of RM Agreements.....	18
58	3.1.6	RM Capability.....	18
59	3.2	Main Reliability Features.....	18
60	3.2.1	Guaranteed Delivery.....	19
61	3.2.2	Duplicate Elimination.....	19
62	3.2.3	Guaranteed Message Ordering.....	20
63	4	Message Format.....	21
64	4.1	Structure.....	21
65	4.2	Request Element.....	23
66	4.2.1	Element: Request/MessageId.....	24
67	4.2.2	Element: Request/ExpiryTime.....	26
68	4.2.3	Element: Request/ReplyPattern.....	27

69	4.2.4 Element: Request/AckRequested.....	28
70	4.2.5 Element: Request/DuplicateElimination.....	29
71	4.2.6 Element: Request/MessageOrder.....	29
72	4.2.7 Example.....	29
73	4.3 PollRequest Element.....	30
74	4.3.1 Element: PollRequest/ReplyTo.....	31
75	4.3.2 Element: PollRequest/RefToMessageIds.....	32
76	4.3.3 Example.....	33
77	4.4 Response Element.....	34
78	4.4.1 Element: Response/NonSequenceReply.....	35
79	4.4.2 Element: Response/SequenceReplies.....	36
80	4.4.3 Example.....	37
81	4.5 Fault Codes For Reliable Messaging Failures.....	37
82	4.5.1 Message Format Faults.....	38
83	4.5.2 Message Processing Faults.....	40
84	4.5.3 RM Fault Examples.....	41
85	4.6 Extensibility Features of Schema.....	41
86	5 Operational Aspects and Semantics.....	43
87	5.1 Message Group Life Cycle.....	43
88	5.1.1 Group Termination.....	43
89	5.1.2 Group Termination Parameters.....	44
90	5.1.3 Termination Rules.....	45
91	5.2 Attachments.....	48
92	6 HTTP Binding.....	49
93	6.1 Reliable Messaging with Response RM-Reply Pattern.....	50
94	6.2 Reliable Messaging with Callback RM-Reply Pattern.....	52
95	6.3 Reliable Messaging with Poll RM-Reply Pattern.....	54
96	6.3.1 Synchronous Poll RM-Reply Pattern.....	54
97	6.3.2 Asynchronous Poll RM-Reply Pattern.....	56
98	7 Conformance.....	59
99	8 References.....	60
100	Appendix A. Schema (Normative).....	62
101	Appendix B. WS-Reliability Features, Properties and Compositors (Normative and Optional)....	63
102	B.1. Introduction.....	63
103	B.2. Conformance.....	63
104	B.3. WSDL Extensibility Elements.....	64
105	B.3.1. Compositor.....	64

106	B.3.2. Feature	65
107	B.3.3. Property	66
108	B.4. WS-Reliability Feature	66
109	B.5. WS-Reliability Properties	66
110	B.5.1. Guaranteed Delivery Property	66
111	B.5.2. Duplicate Elimination Property	66
112	B.5.3. Message Ordering Property	66
113	B.5.4. Reply Pattern Property	67
114	B.6. Compositor Examples	67
115	B.6.1. Example for the "all" compositor	67
116	B.6.2. Example for the "choice" compositor	68
117	B.6.3. Example for the "one-or-more" compositor	69
118	B.6.4. Example for the "zero-or-more" compositor	69
119	Appendix C. Acknowledgments	70
120	Appendix D. Notices	72

121 List of Tables

122	Table 1 Labels.....	8
123	Table 2 Prefixes.....	8
124	Table 3 RM Agreement Items.....	17
125	Table 4 Request Element.....	23
126	Table 5 MessageId Element.....	24
127	Table 6 SequenceNum Element.....	25
128	Table 7 ExpiryTime Element.....	26
129	Table 8 ReplyPattern Element.....	27
130	Table 9 Value Element.....	27
131	Table 10 ReplyTo Element.....	28
132	Table 11 BareURI Element.....	28
133	Table 12 AckRequested Element.....	29
134	Table 13 DuplicateElimination Element.....	29
135	Table 14 MessageOrder Element.....	29
136	Table 15 PollRequest Element.....	31
137	Table 16 ReplyTo Element.....	31
138	Table 17 BareURI Element.....	32
139	Table 18 RefToMessageIds Element.....	32
140	Table 19 SequenceNumRange Element.....	33
141	Table 20 Response Element.....	35
142	Table 21 NonSequenceReply Element.....	35
143	Table 22 SequenceReplies Element.....	36
144	Table 23 ReplyRange Element.....	36
145	Table 24 Invalid Message Format Fault Code Values.....	39
146	Table 25 Messaging Processing Failure Fault Code Values.....	40
147	Table 26 Conditions for terminating a group – Receiving RMP.....	48
148	Table 27 Conditions for terminating a group – Sending RMP.....	48
149	Table 28 WS-Reliability Schema Prefixes.....	62

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.

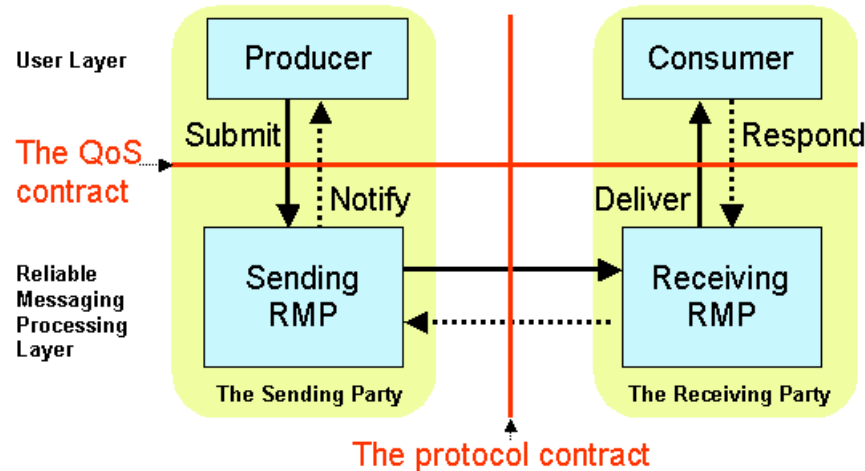


Figure 1 Reliable Messaging Contracts

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)

- 220 • appendices not explicitly identified as normative

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

<i>Label</i>	<i>Meaning</i>
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

223 This specification uses the following namespace prefixes:

<i>Prefix</i>	<i>Namespace</i>
soap	http://schemas.xmlsoap.org/soap/envelope/
soap12	http://www.w3.org/2003/05/soap-envelope
wstrm	http://docs.oasis-open.org/wstrm/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/wstrm/2004/06/fnp-1.1.xsd
wstrmfp	http://docs.oasis-open.org/wstrm/2004/06/wstrmfp-1.1.xsd
ref	http://docs.oasis-open.org/wstrm/2004/06/reference-1.1.xsd

Table 2 Prefixes

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

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

226 **1.4 Relation to Other Specifications**

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

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

243 1.5 Terminology

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

246 **Reliable Messaging (RM):**

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

250 **Reliable Messaging Processor (RMP):**

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

255 **Reliable Message:**

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

257 **Payload:**

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

260 **Producer (or Payload Producer)**

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

263 **Consumer (or Payload Consumer)**

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

267 **Deliver:**

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

269 **Submit:**

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

272 **Respond:**

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

275 **Notify:**

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

279 **RMP Operations:**

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

284 **Message Identifier:**

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

287 **Duplicate Message:**

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

289 **Message Delivery:**

290 Completion of the Deliver operation for a Reliable Message.

291 **Acknowledgment Indication:**

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

296 **Reliable Messaging Fault Indication (RM Fault):**

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

300 **Reliable Messaging Reply (RM-Reply):**

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

303 **Response, Callback and Poll RM-Reply Patterns:**

304 See [Section 2.5](#).

305 **PollRequest Message:**

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

308 **Intermediary:**

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

310 **Publish (an RM-Reply):**

311 The set of mechanisms that make an RM-Reply available to the Sending RMP. The particular
312 mechanism used for a given Publish operation depends on the RM-Reply Pattern ([Section 2.5](#))
313 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].

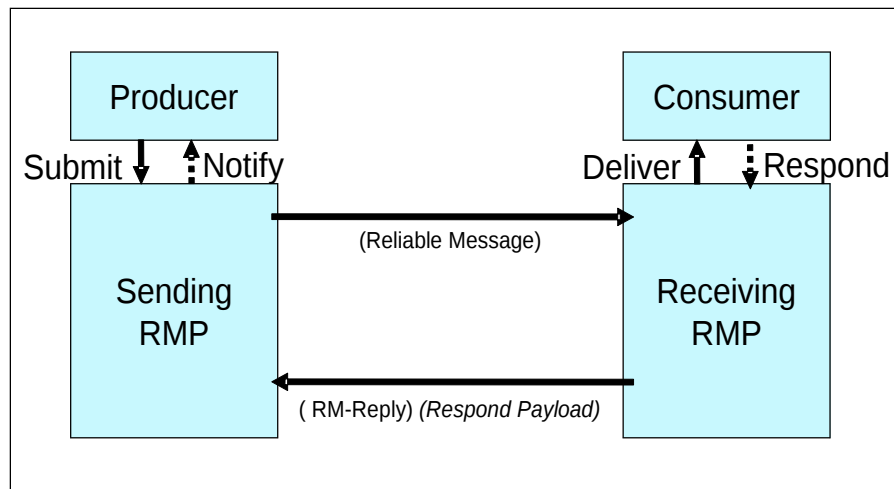


Figure 2 Messaging Model

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.

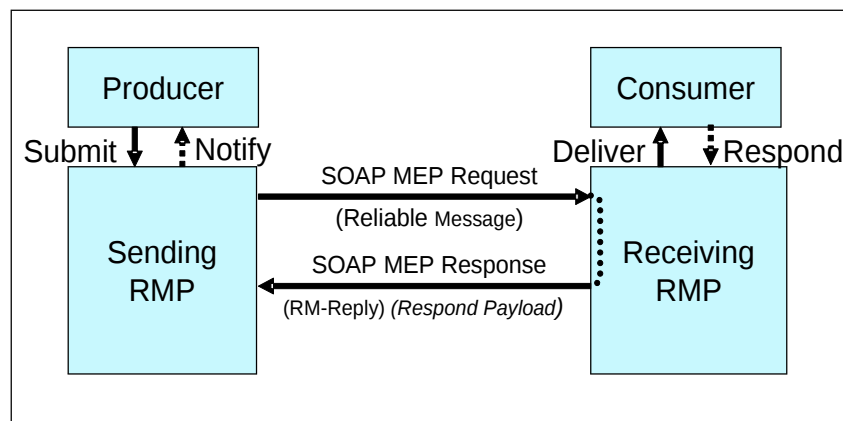


Figure 3 Response RM-Reply Pattern

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.

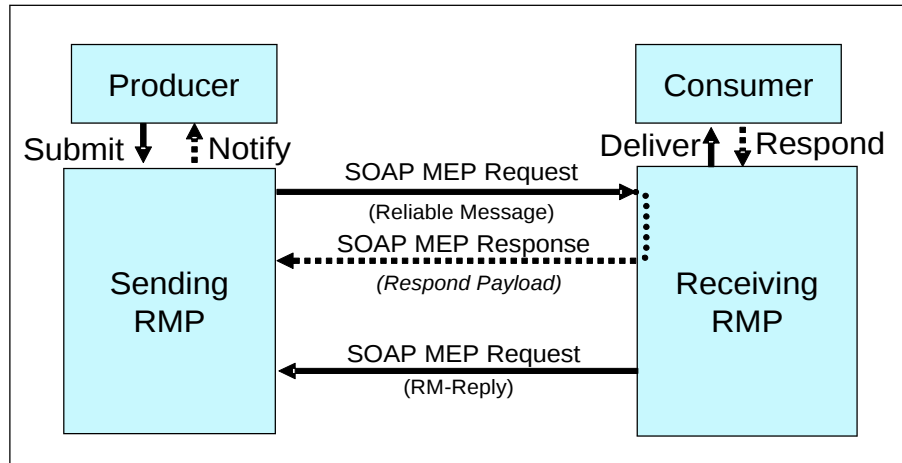


Figure 4 Callback RM-Reply Pattern

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.

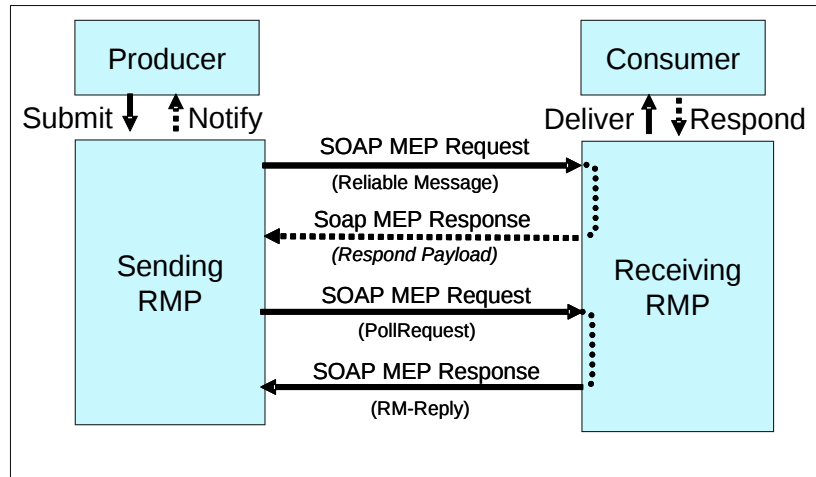


Figure 5 Synchronous Poll RM-Reply Pattern

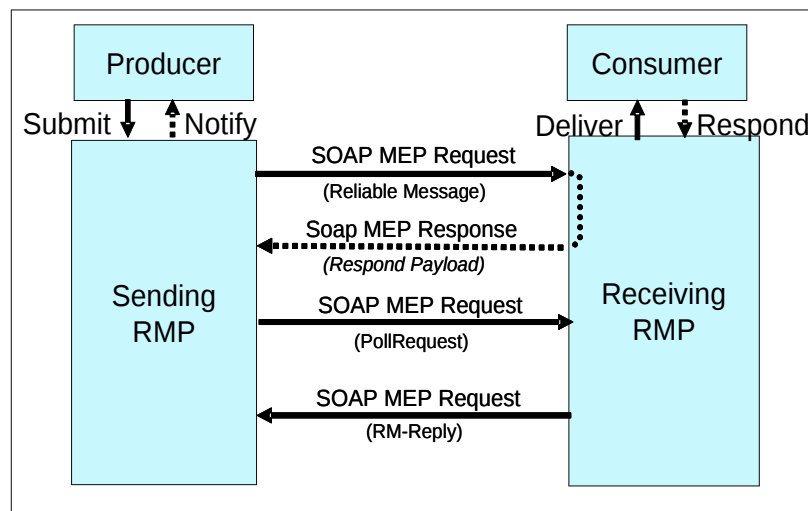


Figure 6 Asynchronous Poll RM-Reply Pattern

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.

574 • When resending a message, the message contents must not change.

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

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

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

584 • a SOAP server Fault (in the SOAP Body) in addition to the Acknowledgment Indication
585 (in the SOAP Header).

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

588 **3.2.3 Guaranteed Message Ordering**

589 Quality of Service requirements:

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

594 **Note:**

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

599 Protocol requirements:

600 Ordering is supported only over messages of the same group.

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

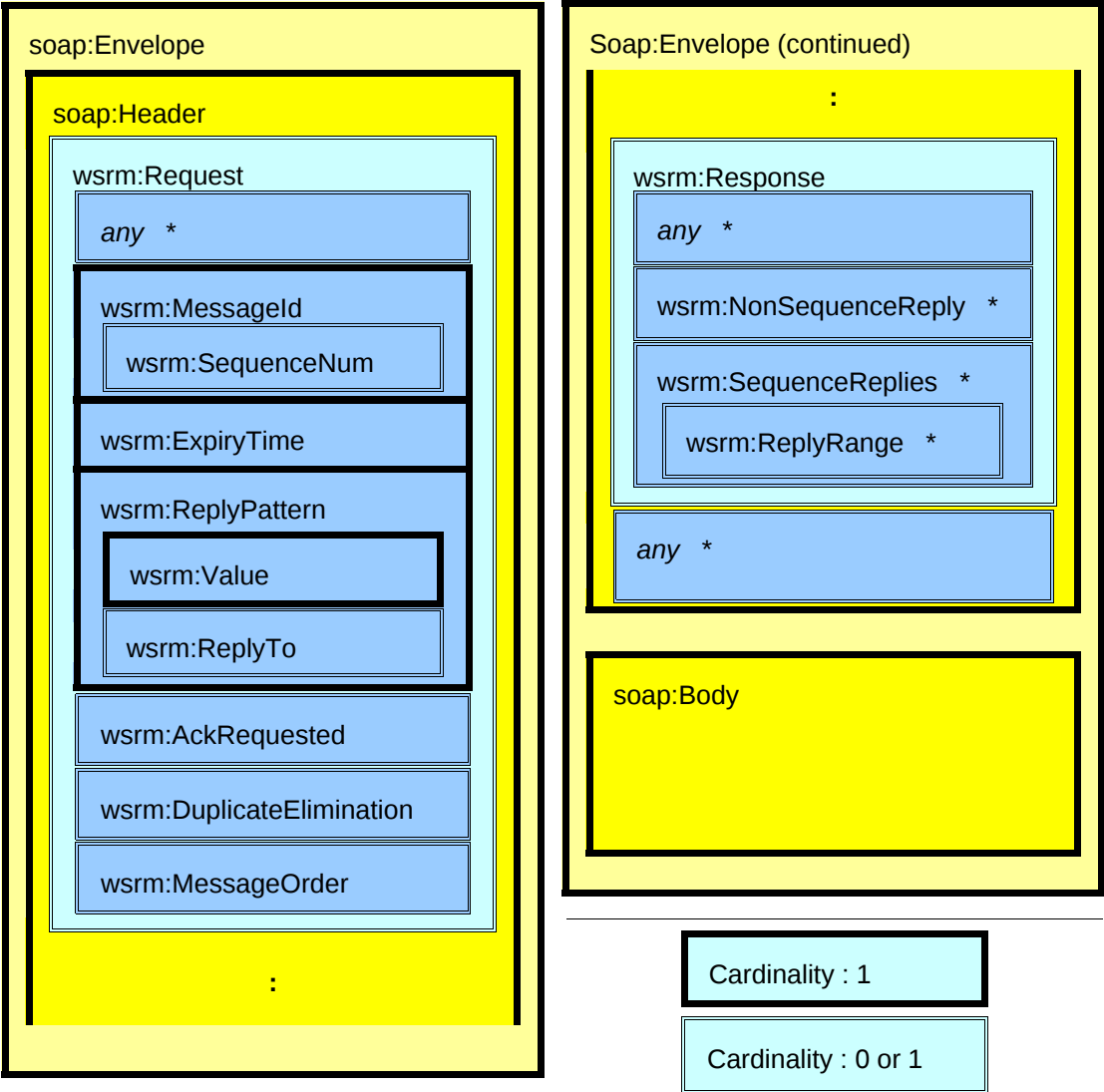
- 603 • The Sending RMP MUST reflect the order of the Submit invocations on this RMP in the
604 sequence numbers of the corresponding messages sent.
- 605 • The Receiving RMP MUST deliver the messages received according to the order
606 expressed by their sequence numbers, which is the same as the submission order.

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

4 Message Format

4.1 Structure

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 `wsm:Request` element. On the right side a response to a Reliable Message contains a `wsm:Response` element. Both `wsm:Request` and `wsm:Response` elements may be found in the same message.



* : An element with this mark may appear more than one time

Figure 7 Structure of WS-Reliability elements

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

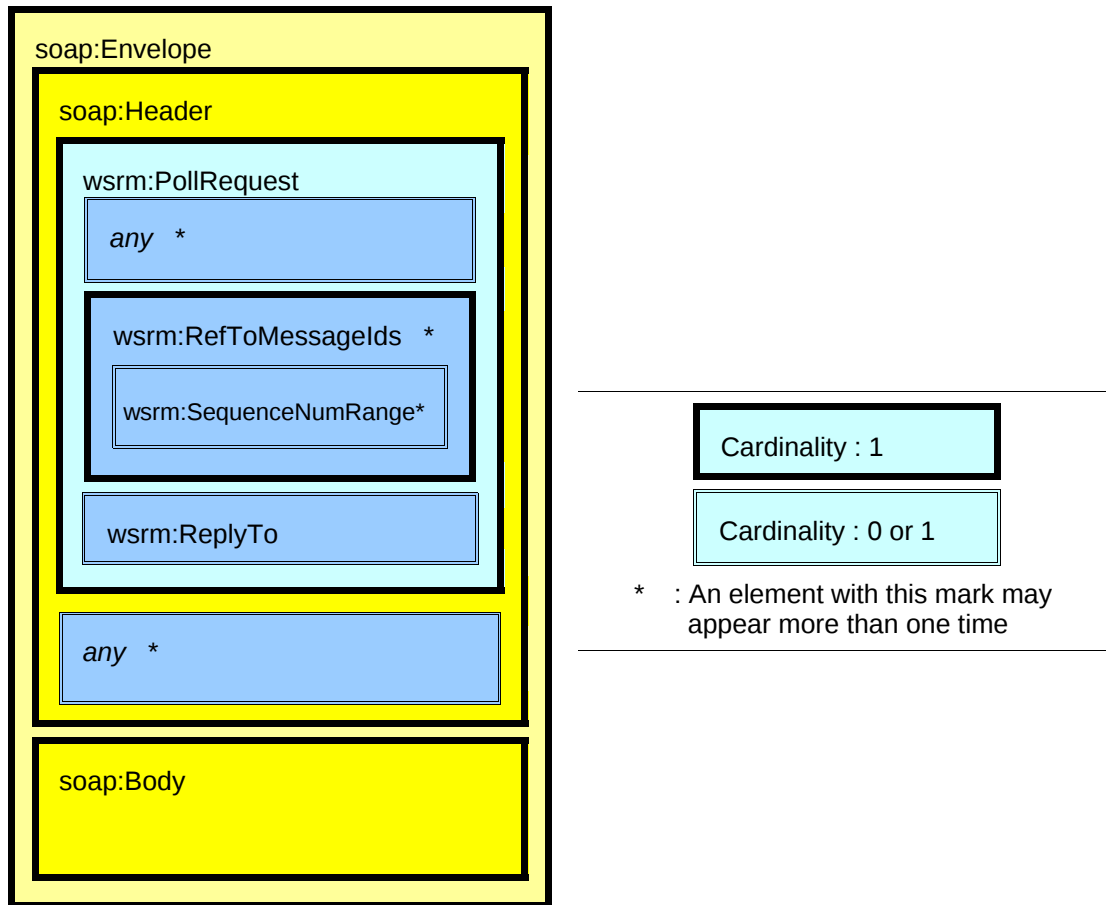


Figure 8 Structure of PollRequest message elements

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

618 <http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd>

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

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

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

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

- 627 • **Request** element
- 628 • **PollRequest** element
- 629 • **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/wsrn/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:

- 674 • a **groupExpiryTime** attribute
- 675 • a **groupMaxIdleDuration** attribute
- 676 • a **number** attribute
- 677 • a **last** attribute

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

680 **Example 2** illustrates the SequenceNum element with some message fragments:

Example 2 SequenceNum Element

681 1) First message

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

686 2) Second message

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

691 3) The last message for the group

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

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

Table 6 SequenceNum Element

696 **4.2.1.2.1 Attribute: Request/MessageId/SequenceNum@groupExpiryTime**

697 This attribute represents the GroupExpiryTime agreement item (**Section 3.1.2, Table 3**). It
 698 specifies the the date and time at which the sender wishes the group to terminate. The
 699 @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.

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

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

Table 10 ReplyTo Element

758 **4.2.3.2.1 Attribute: Request/ReplyPattern/ReplyTo@reference-scheme**

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

762 **4.2.3.2.2 Element: Request/ReplyPattern/ReplyTo/BareURI**

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

768 **Section 6** provides additional information about the specific case for which the content of a
769 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

770 **4.2.4 Element: Request/AckRequested**

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

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

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

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

Table 12 AckRequested Element

784 4.2.5 Element: Request/DuplicateElimination

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

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

Table 13 DuplicateElimination Element

789 4.2.6 Element: Request/MessageOrder

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

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

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

Table 14 MessageOrder Element

795 4.2.7 Example

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

Example 3 Reliable Message with Request header

```
801 POST /abc/servlet/wsrEndpoint HTTP/1.0
802 Content-Type: text/xml; charset=utf-8
803 Host: 192.168.183.100
804 SOAPAction: ""
805 Content-Length: 736
806
807 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
808   <soap:Header>
809     <Request
810       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
811       soap:mustUnderstand="1">
812       <MessageId groupId="mid://20040202.103832@wsr-sender.org">
813         <SequenceNum number="0"
814           groupExpiryTime="2005-02-02T03:00:33-31:00" />
815       </MessageId>
816       <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
817       <ReplyPattern>
818         <Value>Poll</Value>
819       </ReplyPattern>
820       <AckRequested/>
821       <DuplicateElimination/>
822       <MessageOrder/>
823     </Request>
824   </soap:Header>
825   <soap:Body>
826     <Request xmlns="http://example.org/wsr">Request Message</Request>
827   </soap:Body>
828 </soap:Envelope>
```

4.3 PollRequest Element

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

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

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

840 This element includes the following attribute and child elements:

- 841 • SOAP **mustUnderstand** attribute (see [Appendix A](#) for details)
- 842 • a **ReplyTo** element

843 • 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

```
844 <PollRequest
845   xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd"
846   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
847   soap:mustUnderstand="1">
848   <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
849     <SequenceNumRange from="0" to="5"/>
850     <SequenceNumRange from="15" to="20"/>
851   </RefToMessageIds>
852   <RefToMessageIds groupId="mid://20040202.103811@wsr-sender.org" />
853   <RefToMessageIds groupId="mid://20040202.103807@wsr-sender.org">
854     <SequenceNumRange from="713" to="6150"/>
855   </RefToMessageIds>
856 </PollRequest>
```

857 4.3.1 Element: PollRequest/ReplyTo

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

861 **Section 4.2.3.2** provides additional information about the very similar
862 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

863 4.3.1.1 Attribute: PollRequest/ReplyTo@reference-scheme

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

866 **4.3.1.2 Element: PollRequest/ReplyTo/BareURI**

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

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

Table 17 BareURI Element

869 **4.3.2 Element: PollRequest/RefToMessageIds**

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

- 873 • a **groupId** attribute
- 874 • zero or more **SequenceNumRange** elements

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

Table 18 RefToMessageIds Element

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

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

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

885 **4.3.2.1 Attribute: PollRequest/RefToMessageIds@groupId**

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

888 **4.3.2.2 Element: PollRequest/RefToMessagelds/SequenceNumRange**

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

- 892 • a **from** attribute
893 • a **to** attribute

894 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

895 **4.3.2.2.1 Attribute:**
896 **PollRequest/RefToMessagelds/SequenceNumRange@from**

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

899 **4.3.2.2.2 Attribute: PollRequest/RefToMessagelds/SequenceNumRange@to**

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

902 **4.3.3 Example**

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

Example 5 PollRequest Message embedded in HTTP Request

```
907 POST /abc/servlet/wsrEndpoint HTTP/1.0
908 Content-Type: text/xml; charset=utf-8
909 Host: 192.168.183.100
910 SOAPAction: ""
911 Content-Length: 432
912
913 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
914   <soap:Header>
915     <PollRequest
916       xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd"
917       soap:mustUnderstand="1">
918       <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
919         <SequenceNumRange from="0" to="20"/>
920       </RefToMessageIds>
921     </PollRequest>
922   </soap:Header>
923   <soap:Body />
924 </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

946 **Example 6** shows an instance of the Response element.

Example 6 Response Element

```

947 <Response
948   xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd"
949   xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
950   soap:mustUnderstand="1">
951   <NonSequenceReply groupId="mid://20040202.103832@wsr-sender.org" />
952   <SequenceReplies groupId="mid://20040202.103807@wsr-sender.org">
953     <ReplyRange from="1" to="4" />
954     <ReplyRange from="5" to="5" fault="wsrm:InvalidRequest" />
955     <ReplyRange from="6" to="42" />
956   </SequenceReplies>
957   <NonSequenceReply groupId="mid://20040202.103811@wsr-sender.org"
958     fault="wsrm:PermanentProcessingFailure" />
959 </Response>

```

960 **4.4.1 Element: Response/NonSequenceReply**

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

- 963 • a **groupId** attribute
- 964 • a **fault** attribute

965 The @fault indicates a particular fault for the identified message. Without this attribute, the
966 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

967 **4.4.1.1 Attribute: Response/NonSequenceReply@groupId**

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

971 **4.4.1.2 Attribute: Response/NonSequenceReply@fault**

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

974 **4.4.2 Element: Response/SequenceReplies**

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

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

Table 22 SequenceReplies Element

978 **4.4.2.1 Attribute: Response/SequenceReplies@groupId**

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

982 **4.4.2.2 Element: Response/SequenceReplies/ReplyRange**

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

1024 • The Message Format fault set, which includes all faults generated because of a
1025 malformed Reliable Message header.

1026 • The Message Processing fault set, which includes all faults generated while processing
1027 the message.

1028 They are explained in detail in the following sections. The Receiving RMP returns these protocol-
1029 specific fault codes within the Response header element. Reliable Message Faults are carried in
1030 the SOAP Header and do not rely exclusively on the SOAP Fault model for the following reasons:

1031 • The SOAP Fault model does not allow batching of several faults in the same message.

1032 • RM Faults may be carried along with business messages that are unrelated to these
1033 faults; they should not affect the processing of the SOAP body in such messages.

1034 The rules for processing faults are:

1035 • The Receiving RMP MUST NOT deliver a message for which an RM Fault is published.
1036 Therefore, the Receiving RMP MUST NOT send an Acknowledgment Indication for such
1037 a message.

1038 • If a Reliable Message sent over a SOAP Request-response MEP cannot be delivered to
1039 the Consumer, the response of the SOAP MEP instance SHALL contain a SOAP Fault
1040 (in the SOAP Body) in addition to the appropriate RM Fault (in the SOAP Header). If the
1041 specific RM Fault encountered was due to a problem with the request header element,
1042 the Receiving RMP MUST set the value of the soap:Fault@faultcode attribute to
1043 "soap:Client" (for SOAP 1.1 messages) or the soap12:Fault/Code/Value element to
1044 "soap12:Sender" (for SOAP 1.2 messages). If the specific RM Fault encountered was
1045 due to a problem with processing by the Receiving RMP, the Receiving RMP MUST set
1046 the value of the soap:Fault@faultcode attribute to "soap:Server" (for SOAP 1.1
1047 messages) or the soap12:Fault/Code/Value element to "soap12:Receiver" (for SOAP
1048 1.2 messages). The Sending RMP and Producer expect either a complete response or
1049 a SOAP Fault when using the SOAP Request-response MEP; this requirement satisfies
1050 those expectations. More details are given in **Section 3.2** and in the HTTP Binding
1051 section (**Section 6**).

1052 • When a Reliable Message sent over a SOAP One-way MEP cannot be delivered to the
1053 Consumer due to a failure in processing the RM headers, a SOAP Fault SHALL NOT be
1054 returned. The HTTP binding section (**Section 6**) gives more details on the
1055 recommended behavior in such case.

1056 The Fault codes described in **Sections 4.5.1** and **4.5.2** are allowed values for @fault in a
1057 Response element.

1058 **4.5.1 Message Format Faults**

1059 The Receiving RMP publishes these faults when the message format of the Reliable Messaging
1060 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:**

- 1062 Cases exist in which the Receiving RMP is unable to send RM Fault Indications for messages
 1063 with invalid message headers, such as:
- 1064 • The ReplyTo element is missing or invalid in the Callback and asynchronous Poll cases.
 - 1065 • The MessageId element is missing for the Request element.
 - 1066 • The RefToMessageIds is missing for the PollRequest element.

1067 4.5.2 Message Processing Faults

1068 The Receiving RMP publishes these faults when there is an error processing a valid Reliable
 1069 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/wsrn/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/wsrn/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

- 1109 • Response
- 1110 • PollRequest
- 1111 • NonSequenceReply
- 1112 • SequenceReplies
- 1113 • ReplyRange
- 1114 The extensibility of the ReplyTo elements (**Sections** [4.2.3.2](#) and [4.3.1](#)) is somewhat different; it is
- 1115 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.

1200 d) The Sending RMP MAY modify either time-out parameter, sending a subsequent
1201 message with the new value. When applying termination rules, the Sending RMP MUST
1202 use the value in the message with the highest sequence number sent for the group. The
1203 Receiving RMP MUST use the value from the message with the highest sequence number
1204 received for the group.

1205 e) @groupMaxIdleDuration can be either increased or decreased without restriction. The
1206 Sending RMP may increase or decrease @groupExpiryTime as long as it is never less than
1207 the max(ExpiryTime) of the messages sent for the group so far.

1208 The Receiving RMP MUST publish an InvalidMessageParameters Fault for a message with
1209 a @groupExpiryTime value less than the max(ExpiryTime) of the messages previously
1210 received for the group.

1211 5.1.3 Termination Rules

1212 Termination is the process by which an RMP discontinues the use of a group, allowing the RMP
1213 to reclaim resources used by the group. Termination typically involves two steps that may occur
1214 at different times: closing and removal. Removal of a group may happen some time after it is
1215 closed, allowing an RMP to filter out potential duplicate messages. The general rule is that a
1216 group is removed once all of its messages have expired. If we define max(ExpiryTime) as the
1217 maximum date and time of all ExpiryTime values of the messages sent for a group (on the
1218 Sender side) or received for a group (on the Receiver side), a group will not be removed before
1219 max(ExpiryTime) occurs.

1220 There are two general indicators an RMP will use to terminate a group:

- 1221 a) Message Marker: Information within a message (either
1222 Request/MessageId/SequenceNum@last="true" or the maximum sequence number)
1223 indicates the last message for the group. This is used by termination rules T3, T4.
- 1224 b) Timing: Either the group's lifespan expired or its idle time exceeded a time-out. This is
1225 used by termination rules T1, T2. Or due to message expiration, a group with the ordering
1226 requirement cannot be delivered. This is used by termination rule T5.

1227 These termination rules apply to both ordered and unordered groups. However, these rules do
1228 not apply to groups that contain a single message with no sequence number.

1229 5.1.3.1 Termination by expiration (T1):

1230 Context:

1231 The group specified @groupExpiryTime.

1232 Receiver side:

1233 Triggering event: @groupExpiryTime is in the past.

1234 The RMP MUST close and remove the group.

1235 Sender side:

1236 Triggering event: @groupExpiryTime is in the past (note: in this case, max(ExpiryTime) also is
1237 past).

1238 The RMP MUST close and remove the group.

1239 **5.1.3.2 Termination by idle time-out (T2):**

1240 Context:

1241 The group specified @groupMaxIdleDuration.

1242 Receiver side:

1243 Triggering event: The time since the last received message for the group is over
1244 @groupMaxIdleDuration.

1245 The RMP MUST close the group. But unlike T1, some of its past messages may not have expired
1246 yet. In case Duplicate Elimination is required, the RMP MUST NOT remove the group until max
1247 (ExpiryTime) is reached in order to make sure all potential duplicates for the group will not be
1248 delivered.

1249 Sender side:

1250 Triggering event: The time since the last sent message for the group is over
1251 @groupMaxIdleDuration.

1252 The RMP MUST close the group. If GuaranteedDelivery was required, the RMP MUST remove
1253 the group once it has received either acknowledgment or notification of delivery failure for all sent
1254 messages. If no GuaranteedDelivery was required, the RMP MUST remove the group
1255 immediately.

1256 **5.1.3.3 Termination by completeness (T3):**

1257 Context:

1258 No specific context.

1259 Receiver side:

1260 Triggering event: The RMP receives a message marked last
1261 (Request/MessageId/SequenceNum@last="true"). If all previous messages for the group have
1262 been received, the group is closed immediately. Alternately, the group is closed when the RMP
1263 receives the last missing message in the group.

1264 The RMP MUST close the group. However, its removal is done according to T1 or T2 depending
1265 on which time-out parameter was specified for the group. If no time-out parameter was specified,
1266 the group is removed once all of its messages have expired, i.e., the date and time max
1267 (ExpiryTime) has passed.

1268 **Note:**

1269 In the case in which a message is received with an ending marker before all previous messages
1270 have been received, the group remains active. No termination process is initiated yet.

1271 Sender side:

1272 Triggering event: The RMP sends a message marked last.

1273 All messages of the group have been sent. The RMP MUST close the group. If
1274 GuaranteedDelivery was required, the RMP MUST remove the group once it has received either
1275 acknowledgment or notification of delivery failure for all sent messages. If GuaranteedDelivery
1276 was not required, the RMP MUST remove the group immediately.

1277 **5.1.3.4 Termination by sequence exhaustion (T4):**

1278 Context:

1279 No specific context.

1280 Receiver side:

1281 Triggering event: The RMP receives a message with a sequence number of the maximum value.
1282 If all previous messages for the group have been received, the group is closed immediately.
1283 Alternately, the group is closed when the RMP receives the last missing message in the group.

1284 The group closing and removal follow the rules in T3, the message with the maximum sequence
1285 number acting as a message with the ending mark.

1286 **Note:**

1287 In case a message is received with the maximum sequence number before all previous
1288 messages have been received, the group remains active. No termination process is initiated yet.

1289 Sender side:

1290 Triggering event: The RMP sends a message with a sequence number with the maximum value.

1291 The group closing and removal follow the rules in T3, the message with the maximum sequence
1292 number acting as a message with the ending mark.

1293 **5.1.3.5 Termination by ordering failure (T5):**

1294 Context:

1295 The group requires the Guaranteed Message Ordering reliability feature.

1296 Receiving side:

1297 Triggering event: In an ordered group, a received message expires before delivery or faults with
1298 a fault code other than MessageProcessingFailure. If all previous messages for the group have
1299 been received, the group is closed immediately. Alternately, the group is closed when the RMP
1300 receives the last missing message in the group.

1301 The RMP MUST close the group. The group is removed according to rule T3.

1302 Sender Side:

1303 Triggering event: In an ordered group, an unacknowledged message expires or the RMP
1304 receives an RM Fault for this Reliable Message with a fault code other than
1305 MessageProcessingFailure.

1306 The RMP MUST close the group. The group is removed according to rule T3.

1307 **5.1.3.6 Summary of Group Termination Rules**

1308 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

1309 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

1310 5.2 Attachments

1311 When an RMP implementing this specification uses the W3C Note “SOAP Messages with
1312 Attachments” specification [SOAP with Attachments], it MUST follow the following rules:

- 1313 1) The Sending RMP MUST include the whole SOAP envelope containing the WS-
1314 Reliability header elements in the first MIME part.
- 1315 2) It MUST set the charset parameter of the Content-Type header of the first MIME part to
1316 either UTF-8 or UTF-16.
- 1317 3) It MAY include zero or more additional MIME parts in a Reliable Message.
- 1318 4) The Receiving RMP MUST deliver all MIME parts in a Reliable Message to the
1319 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 wsrn: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 wsrn: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

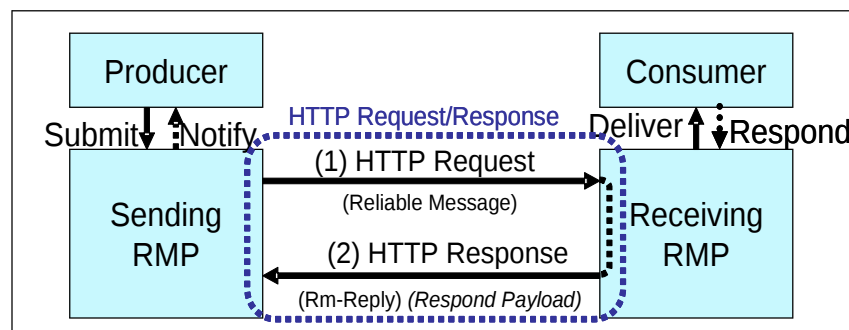


Figure 9 Response RM-Reply Pattern

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

```
1384 POST /abc/servlet/wsrEndpoint HTTP/1.0
1385 Content-Type: text/xml; charset=utf-8
1386 Host: 192.168.183.100
1387 SOAPAction: ""
1388 Content-Length: 755
1389
1390 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1391   <soap:Header>
1392     <Request
1393       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
1394       soap:mustUnderstand="1">
1395       <MessageId groupId="mid://20040202.103832@wsr-sender.org">
1396         <SequenceNum number="0"
1397           groupExpiryTime="2005-02-02T03:00:33-31:00" />
1398       </MessageId>
1399       <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
1400       <ReplyPattern>
1401         <Value>Response</Value>
1402       </ReplyPattern>
1403       <AckRequested/>
1404       <DuplicateElimination/>
1405       <MessageOrder/>
1406     </Request>
1407   </soap:Header>
1408   <soap:Body>
1409     <Request xmlns="http://example.org/wsr">Request Message</Request>
1410   </soap:Body>
1411 </soap:Envelope>
```

Example 11 Acknowledgment Indication with Response RM-Reply Pattern

```
1412 HTTP/1.0 200 OK
1413 Server: WS-ReliabilityServer
1414 Date: Mon, 02 Feb 2004 10:38:32 GMT
1415 Content-Language: en
1416 Content-Type: text/xml; charset=utf-8
1417 Content-Length: 414
1418
1419 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1420   <soap:Header>
1421     <Response soap:mustUnderstand="1"
1422       xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd">
1423       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1424         <ReplyRange from="0" to="0"/>
1425       </SequenceReplies>
1426     </Response>
1427   </soap:Header>
1428   <soap:Body />
1429 </soap:Envelope>
```

6.2 Reliable Messaging with Callback RM-Reply Pattern

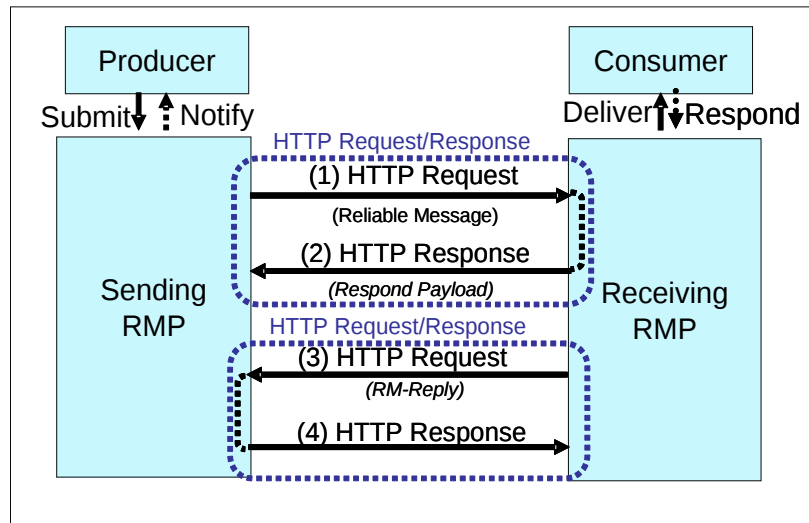


Figure 10 Callback RM-Reply Pattern

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

1439 connection used by the Receiving RMP is from the Receiving RMP to the Sending RMP.
1440 **Example 14** shows an Acknowledgment Indication as the RM-Reply.

- 1441 • In (4), there is no HTTP entity-body unless the RM-Reply was bundled with a new
1442 Reliable Message on a SOAP Request-response MEP instance.

Example 12 Request Message with Callback RM-Reply Pattern

```
1443 POST /abc/servlet/wsrEndpoint HTTP/1.0
1444 Content-Type: text/xml; charset=utf-8
1445 Host: 192.168.183.100
1446 SOAPAction: ""
1447 Content-Length: 863
1448
1449 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1450   <soap:Header>
1451     <Request
1452       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
1453       soap:mustUnderstand="1">
1454       <MessageId groupId="mid://20040202.103832@wsr-sender.org">
1455         <SequenceNum number="0"
1456           groupExpiryTime="2005-02-02T03:00:33-31:00" />
1457       </MessageId>
1458       <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
1459       <ReplyPattern>
1460         <Value>Callback</Value>
1461         <ReplyTo>
1462           <BareURI>http://wsr-sender.org/abc/wsrmlistener</BareURI>
1463         </ReplyTo>
1464       </ReplyPattern>
1465       <AckRequested/>
1466       <DuplicateElimination/>
1467       <MessageOrder/>
1468     </Request>
1469   </soap:Header>
1470   <soap:Body>
1471     <Request xmlns="http://example.org/wsr">Request Message</Request>
1472   </soap:Body>
1473 </soap:Envelope>
```

Example 13 HTTP response with no content

```
1474 HTTP/1.0 200 OK
1475 Server: WS-ReliabilityServer
1476 Date: Mon, 02 Feb 2004 10:38:32 GMT
1477 Content-Language: en
1478 Content-Type: text/xml; charset=utf-8
1479 Content-Length: 0
```

Example 14 Acknowledgment Indication with Callback RM-Reply Pattern

```
1480 POST /abc/wsrmlistener HTTP/1.0
1481 Content-Type: text/xml; charset=utf-8
1482 Host: 192.168.183.200
1483 SOAPAction: ""
1484 Content-Length: 414
1485
1486 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
1487   <soap:Header>
1488     <Response soap:mustUnderstand="1"
1489       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd">
1490       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1491         <ReplyRange from="0" to="0"/>
1492       </SequenceReplies >
1493     </Response>
1494   </soap:Header>
1495   <soap:Body />
1496 </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

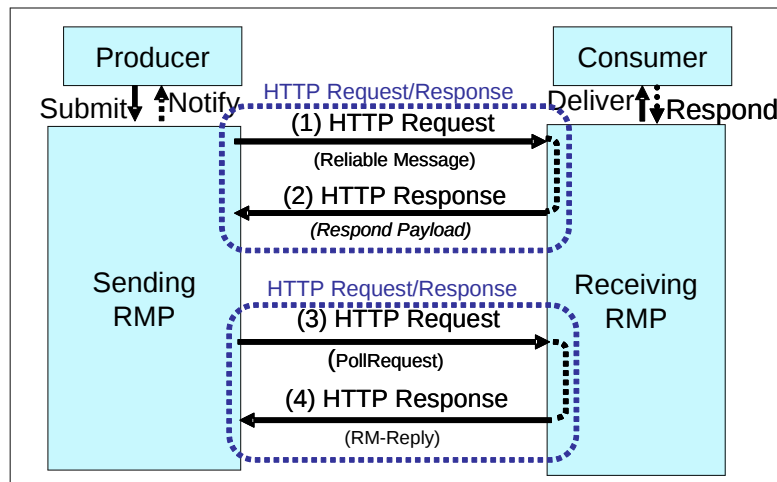


Figure 11 Synchronous Poll RM-Reply Pattern

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.

- 1508 • In (2), the HTTP response may have an empty entity-body (in case of a SOAP One-way
1509 MEP instance).
- 1510 • In (3), at a later time the Sending RMP initiates a different HTTP Request to send a
1511 PollRequest message. The PollRequest does not include the ReplyTo element (see
1512 **Example 15**).
- 1513 • In (4), the Receiving RMP returns the RM-Reply in an HTTP response on the same
1514 HTTP connection used to send the PollRequest, as illustrated in **Figure 11**. The HTTP
1515 response (4) includes an RM-Reply (e.g., an Acknowledgment Indication as in **Example**
1516 **16**).

Example 15 PollRequest message with Synchronous Poll RM-Reply Pattern

```
1517 POST /abc/servlet/wsrmlistener HTTP/1.0
1518 Content-Type: text/xml; charset=utf-8
1519 Host: 192.168.183.100
1520 SOAPAction: ""
1521 Content-Length: 433
1522
1523 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1524   <soap:Header>
1525     <PollRequest
1526       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
1527       soap:mustUnderstand="1">
1528       <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
1529         <SequenceNumRange from="0" to="20"/>
1530       </RefToMessageIds>
1531     </PollRequest>
1532   </soap:Header>
1533   <soap:Body />
1534 </soap:Envelope>
```

Example 16 Synchronous Acknowledgment Indication

```
1535 HTTP/1.0 200 OK
1536 Server: WS-ReliabilityServer
1537 Date: Mon, 02 Feb 2004 10:38:32 GMT
1538 Content-Language: en
1539 Content-Type: text/xml; charset=utf-8
1540 Content-Length: 456
1541
1542 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1543   <soap:Header>
1544     <Response soap:mustUnderstand="1"
1545       xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd">
1546       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1547         <ReplyRange from="0" to="14"/>
1548         <ReplyRange from="16" to="20"/>
1549       </SequenceReplies>
1550     </Response>
1551   </soap:Header>
1552   <soap:Body />
1553 </soap:Envelope>
```

6.3.2 Asynchronous Poll RM-Reply Pattern

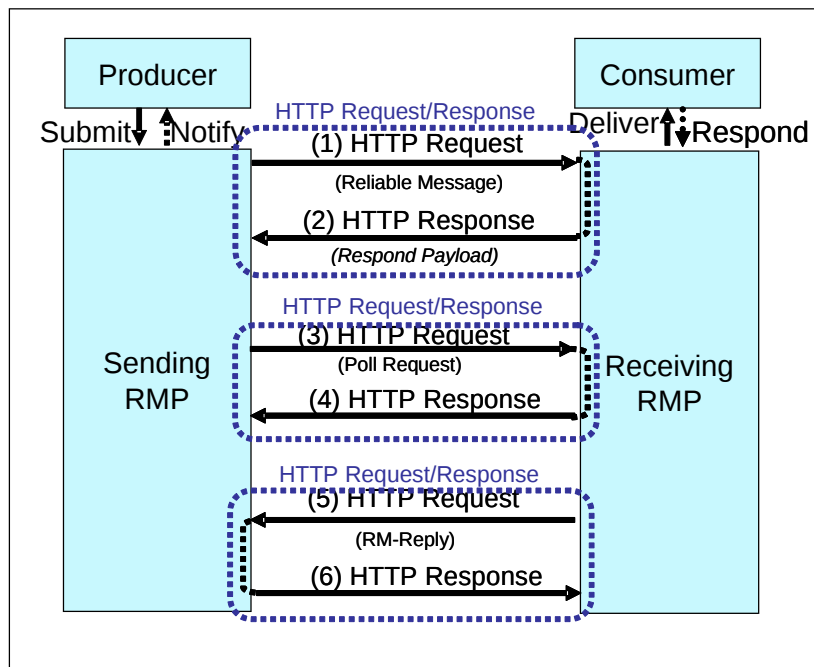


Figure 12 Asynchronous Poll RM-Reply Pattern

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.

- 1558 • In (2), the HTTP response may have an empty entity-body (in the case of a SOAP One-
1559 way MEP instance).
- 1560 • In (3), the Sending RMP initiates a new HTTP request and sends a PollRequest
1561 message. Note that in **Example 17**, the PollRequest element has a ReplyTo element.
- 1562 • In (4), the HTTP response (4) has no HTTP entity-body (see **Example 13**).
- 1563 • In (5), the Receiving RMP sends the RM-Reply in a different HTTP request to the
1564 listener identified by the ReplyTo element (see **Example 18**).
- 1565 • In (6), the HTTP response has no HTTP entity-body (see **Example 13**).

Example 17 PollRequest message with Asynchronous Poll RM-Reply Pattern

```
1566 POST /abc/servlet/wsrmlListener HTTP/1.0
1567 Content-Type: text/xml; charset=utf-8
1568 Host: 192.168.183.100
1569 SOAPAction: ""
1570 Content-Length: 553
1571
1572 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1573   <soap:Header>
1574     <PollRequest
1575       xmlns="http://docs.oasis-open.org/wsrml/2004/06/ws-reliability-1.1.xsd"
1576       soap:mustUnderstand="1">
1577       <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
1578         <SequenceNumRange from="0" to="20"/>
1579       </RefToMessageIds>
1580       <ReplyTo>
1581         <BareURI>http://wsr-sender.org/xyz/servlet/wsrmlListener
1582       </BareURI>
1583       </ReplyTo>
1584     </PollRequest>
1585   </soap:Header>
1586   <soap:Body />
1587 </soap:Envelope>
```

Example 18 Asynchronous Acknowledgment Indication

```
1588 POST /xyz/servlet/wsrmlistener HTTP/1.0
1589 Content-Type: text/xml; charset=utf-8
1590 Host: 192.168.183.200
1591 SOAPAction: ""
1592 Content-Length: 456
1593
1594 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1595   <soap:Header>
1596     <Response soap:mustUnderstand="1"
1597       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd">
1598       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1599         <ReplyRange from="0" to="14"/>
1600         <ReplyRange from="16" to="20"/>
1601       </SequenceReplies>
1602     </Response>
1603   </soap:Header>
1604   <soap:Body />
1605 </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

- 1621
- 1622 [ebMS] "Message Service Specification Version 2.0", OASIS ebXML Messaging Services
1623 Technical Committee, OASIS Standard, 1 April 2002. Available at
1624 <http://www.ebxml.org/specs/ebMS2.pdf>
- 1625 [RFC1738] "Uniform Resource Locators (URL)", T. Berners-Lee et al, RFC 1738, IESG and IETF,
1626 December 1994. Available at
1627 <http://www.ietf.org/rfc/rfc1738.txt>
- 1628 [RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119,
1629 Bradner, S., IESG and IETF, March 1997. Available at
1630 <http://www.ietf.org/rfc/rfc2119.txt>
- 1631 [RFC2392] "Content-ID and Message-ID Uniform Resource Locators", RFC2392, E. Levinson,
1632 IESG and IETF, August 1998. Available at
1633 <http://www.ietf.org/rfc/rfc2392.txt>
- 1634 [RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, Tim Berners-Lee et
1635 al, IESG and IETF, August 1998. Available at
1636 <http://www.ietf.org/rfc/rfc2396.txt>
- 1637 [RFC2616] "Hypertext Transfer Protocol – HTTP/1.1", RFC 2616, R. Fielding et al, IESG and
1638 IETF, June 1999. Available at
1639 <http://www.ietf.org/rfc/rfc2616.txt>
- 1640 [RFC2822] "Internet Message Format", RFC 2822, P. Resnick, Editor, IESG and IETF, April
1641 2001. Available at
1642 <http://www.ietf.org/rfc/rfc2822.txt>
- 1643 [SOAP 1.1] "Simple Object Access Protocol (SOAP) 1.1", Don Box et al, W3C Note, 8 May, 2000.
1644 Available at
1645 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- 1646 [SOAP 1.2 Part 1] "SOAP 1.2 Part 1: Messaging Framework", Martin Gudgin, Marc Hadley, Noah
1647 Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, eds., W3C Recommendation, 24
1648 June 2003. Available at
1649 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

1650 [SOAP 1.2 Part 2] "SOAP 1.2 Part 1: Adjuncts", Martin Gudgin, Marc Hadley, Noah Mendelsohn,
 1651 Jean-Jacques Moreau, Henrik Frystyk Nielsen, eds., W3C Recommendation, 24 June 2003.
 1652 Available at
 1653 <http://www.w3.org/TR/2003/REC-soap12-part2-20030624/>

1654 [SOAP with Attachments] "SOAP Messages with Attachments", John J. Barton, Satish Thatte,
 1655 Henrik Frystyk Nielsen, W3C Note, 11 December 2000, Available at
 1656 <http://www.w3.org/TR/SOAP-attachments>

1657 [XML] "Extensible Markup Language (XML) 1.0 (Third Edition)", Tim Bray et al, eds., W3C
 1658 Recommendation, first published 10 February 1998, revised 4 February 2004. Available at
 1659 <http://www.w3.org/TR/2004/REC-xml-20040204>

1660 [XML Namespaces] "Namespaces in XML", Tim Bray et al., eds., W3C Recommendation, 14
 1661 January 1999. Available at
 1662 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>

1663 [XML Schema Part 1] "XML Schema Part 1: Structures", Henry S. Thompson, David Beech,
 1664 Murray Maloney, Noah Mendelsohn, eds., W3C Recommendation, 2 May 2001. Available at
 1665 <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>

1666 [XML Schema Part 2] "XML Schema Part 2: Datatypes", Paul V. Biron and Ashok Malhotra, eds.
 1667 W3C Recommendation, 2 May 2001. Available at
 1668 <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>

1669 [XPath 1.0] "XML Path Language (XPath) Version 1.0", James Clark, Steve DeRose, eds., W3C
 1670 Recommendation, 16 November 1999. Available at
 1671 <http://www.w3.org/TR/1999/REC-xpath-19991116>

1672 [WSDL 1.1] "Web Services Description Language (WSDL) 1.1", Erik Christensen, Francisco
 1673 Curbera, Greg Meredith, Sanjiva Weerawarana, eds., W3C Note, 15 March 2001. Available at
 1674 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>

1675 [WS-I BP 1.1] "Basic Profile Version 1.1", Keith Ballinger, David Ehnebuske, Christopher Ferris,
 1676 Martin Gudgin, Mark Nottingham, Prasad Yendluri, eds., WS-I specification, 8 August 2004.
 1677 Available at
 1678 <http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-07-21.html>

1679 [WSS] "OASIS Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)", Chris
 1680 Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds, OASIS Standard 200401, March 2004.
 1681 Available at
 1682 <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/wsrn/2004/06/ws-reliability-1.1.xsd	ws-reliability-1.1.xsd	wsrm
http://docs.oasis-open.org/wsrn/2004/06/reference-1.1.xsd	reference-1.1.xsd	ref
http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd	fnp-1.1.xsd	fnp
http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd	wsrmp-1.1.xsd	wsrmp

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/wsrn/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/wsrn/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/wsrn/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/wsrn/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:

```

1765 <wsdl11:definitions>
1766   ...
1767   <wsdl11:portType name="myPortType">
1768     <fnp:compositor uri="..." name="A">
1769       ...
1770     </fnp:compositor>
1771     ...
1772   </wsdl11:portType>
1773   <wsdl11:binding name="myBinding" type="myPortType">
1774     <fnp:compositor uri="..." name="B">
1775       ...
1776     </fnp:compositor>
1777     ...
1778   <wsdl11:binding>
1779     <wsdl11:service name="myService">
1780       <wsdl11:port name="myPort" binding="myBinding">
1781         ...
1782       </wsdl11:port>
1783     </wsdl11:service>
1784   </wsdl11:definitions>

```

1785 The compositor specified at the wsdl11:portType "myPortType" and the compositor specified at
 1786 wsdl11:binding "myBinding" are aggregated at the dependent wsdl11:port "myPort" using the 'all'
 1787 compositor. The equivalent compositor at "myPort" is

```

1788 <fnp:compositor
1789   uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1790   <fnp:compositor uri="..." name="A">
1791   </fnp:compositor>
1792   <fnp:compositor uri="..." name="B">
1793     ...
1794   </fnp:compositor>
1795 </fnp:compositor>

```

1796 B.3.2. Feature

1797 A feature describes an abstract RM capability or assertion associated with a WSDL element. A
 1798 feature can occur only as a child of a compositor.

1799 The enclosing compositor(s) define(s) whether or not the usage of a feature is required. A feature
 1800 is identified by a URI. Recognizing the URI of a feature implies understanding the feature
 1801 identified by that URI.

1802 A feature element is expressed by the following pseudo-syntax:

```

1803 <fnp:feature uri="...">
1804   [<fnp:compositor/> | <extensibility-element/>]*
1805 </fnp:feature>

```

1806 **B.3.3. Property**

1807 A property is identified by a QName. A property is an assertion or constraint on a specific RM
1808 capability and its value(s). A property can occur only as a child of a compositor.

1809 Typically, properties are (but are not required to be) associated with a feature and are described
1810 in a feature specification. The QName identifier of a property uniquely identifies the property.
1811 Recognizing the property QName identifier implies understanding the semantics associated with
1812 that property. The property QName identifier typically points to a global XML Schema element
1813 declaration. A property specification typically specifies the schema containing this global element
1814 declaration. There may be a constraint on the set of values a property can have; such a
1815 constraint is specified by a QName identifying an XML Schema type.

```
1816 <fnp:property name="xs:QName">  
1817   [<fnp:value>xs:anyType</fnp:value> |  
1818     <fnp:constraint>xs:QName</fnp:constraint>]  
1819   [<extensibility-element/>]*  
1820 </fnp:property>
```

1821 **B.4. WS-Reliability Feature**

1822 The WS-Reliability feature is identified by the URI

1823 <http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd>

1824 This feature URI identifies the WS-Reliability specification. Understanding this URI implies
1825 understanding the WS-Reliability specification.

1826 **B.5. WS-Reliability Properties**

1827 This section identifies properties for the WS-Reliability specification. Typically these properties
1828 are scoped within the feature identified by the URI

1829 <http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd>

1830 **B.5.1. Guaranteed Delivery Property**

1831 This property is identified by the QName "wsrmfp:GuaranteedDelivery" and corresponds to the
1832 semantics specified by the WS-Reliability guaranteed delivery semantics. The type of this
1833 property is "xs:boolean".

1834 **B.5.2. Duplicate Elimination Property**

1835 This property is identified by the QName "wsrmfp:NoDuplicateDelivery" and corresponds to the
1836 semantics specified by the WS-Reliability duplicate elimination semantics. The type of this
1837 property is "xs:boolean".

1838 **B.5.3. Message Ordering Property**

1839 This property is identified by the QName "wsrmfp:OrderedDelivery" and corresponds to the
1840 semantics specified by the WS-Reliability message ordering semantics. The type of this property
1841 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/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
    <fnp:feature
      uri="http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd"
      <fnp:compositor uri=
        "http://docs.oasis-open.org/wsrn/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/wsrn/2004/06/wsrnfp-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:

```
1874 <wsdl11:binding name="Example-2">
1875   <fnp:compositor
1876     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1877     <fnp:feature
1878       uri="http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd"
1879       <fnp:compositor uri=
1880         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositors/choice">
1881         <fnp:property name="wsrmp:ReplyPattern">
1882           <value>Response</value>
1883         </fnp:property>
1884         <fnp:property name="wsrmp:ReplyPattern">
1885           <value>Callback</value>
1886         </fnp:property>
1887         <fnp:property name="wsrmp:ReplyPattern">
1888           <value>Poll</value>
1889         </fnp:property>
1890       </fnp:compositor>
1891     </fnp:feature>
1892   </fnp:compositor>
1893   ...
1894 </wsdl11:binding>
```

1895 In the example above, the reliability feature identified by URI "[http://docs.oasis-](http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd)
1896 [open.org/wsrn/2004/06/wsrmp-1.1.xsd](http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd)" is required by the portType. This feature consists of
1897 three properties composed by the 'choice' compositor; the client must choose one.

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

```
1899 <wsdl11:portType name="Example-3">
1900   <fnp:compositor
1901     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1902     <fnp:feature
1903       uri="http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd" >
1904       <fnp:compositor uri=
1905         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/one-or-more">
1906         <fnp:property name="wsrnfp:NoDuplicateDelivery">
1907           <fnp:value>true</fnp:value>
1908         </fnp:property>
1909         <fnp:property name="wsrnfp:OrderedDelivery">
1910           <fnp:value>true</fnp:value>
1911         </fnp:property>
1912         <fnp:property name="wsrnfp:GuaranteedDelivery">
1913           <fnp:value>true</fnp:value>
1914         </fnp:property>
1915       </fnp:compositor>
1916     </fnp:feature>
1917   </fnp:compositor>
1918   ...
1919 </wsdl11:portType>
```

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

```
1921 <wsdl11:portType name="Example-4">
1922   <fnp:compositor
1923     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1924     <fnp:feature
1925       uri="http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd"
1926       <fnp:compositor uri=
1927         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/zero-or-more">
1928         <fnp:property name="wsrnfp:NoDuplicateDelivery">
1929           <fnp:value>true</fnp:value>
1930         </fnp:property>
1931         <fnp:property name="wsrnfp:OrderedDelivery">
1932           <fnp:value>true</fnp:value>
1933         </fnp:property>
1934         <fnp:property name="wsrnfp:GuaranteedDelivery">
1935           <fnp:value>true</fnp:value>
1936         </fnp:property>
1937       </fnp:compositor>
1938     </fnp:feature>
1939   </fnp:compositor>
1940   ...
1941 </wsdl11:portType>
```

Appendix C.Acknowledgments

1942

1943 The following individuals were members of the committee during the development of this
1944 specification:

- 1945 □ David Ingham, Arjuna Technologies Limited
- 1946 □ Joseph Chiusano, Booz Allen Hamilton
- 1947 □ Peter Furniss, Choreology Ltd
- 1948 □ Jeff Turpin, Cyclone Commerce
- 1949 □ Pramila Mullan, France Telecom
- 1950 □ Jacques Durand, Fujitsu
- 1951 □ Kazunori Iwasa (Secretary), Fujitsu
- 1952 □ Tom Rutt (Chair), Fujitsu
- 1953 □ Jishnu Mukerji, Hewlett-Packard
- 1954 □ Robert Freund, Hitachi
- 1955 □ Eisaku Nishiyama, Hitachi
- 1956 □ Nobuyuki Yamamoto, Hitachi
- 1957 □ Ben Bloch, Individual
- 1958 □ Mark Hansen, Individual
- 1959 □ Paolo Romano, Individual
- 1960 □ Dock Allen, Mitre Corporation
- 1961 □ Junichi Tatemura, NEC Corporation
- 1962 □ Alan Weissberger, NEC Corporation
- 1963 □ Magdolna Gerendai, Nokia
- 1964 □ Szabolcs Payrits, Nokia
- 1965 □ Mark Peel, Novell
- 1966 □ Sunil Kunisetty (Secretary), Oracle
- 1967 □ Anish Karmarkar, Oracle
- 1968 □ Jeff Mischkinsky, Oracle
- 1969 □ Marc Goodner (Secretary), SAP
- 1970 □ Pete Wenzel, SeeBeyond Technology Corporation
- 1971 □ Doug Bunting (Secretary), Sun Microsystems

- 1972 □ Tony Graham, Sun Microsystems
- 1973 □ Chi-Yuen Ng, University of Hong Kong
- 1974 □ Patrick Yee, University of Hong Kong
- 1975 □ Prasad Yendluri, webMethods, Inc.
- 1976 □ Scott Werden, WRQ, Inc.

1977 And the following people made contributions to produce Ver 1.0 of this specification:

- 1978 Colleen Evans, Sonic Software Corporation / Dave Chappell, Sonic Software Corporation / Doug
- 1979 Bunting, Sun Microsystems, Inc. / George Tharakan, Sun Microsystems, Inc. / Hisashi
- 1980 Shimamura, NEC Corporation / Jacques Durand, Fujitsu Software Corporation / Jeff Mischkinsky,
- 1981 Oracle Corporation / Katsutoshi Nihei, NEC Corporation / Kazunori Iwasa, Fujitsu Limited / Martin
- 1982 Chapman, Oracle Corporation / Masayoshi Shimamura, Fujitsu Limited / Nicholas Kassem, Sun
- 1983 Microsystems, Inc. / Nobuyuki Yamamoto, Hitachi Limited / Sunil Kunisetty, Oracle Corporation /
- 1984 Tetsuya Hashimoto, Hitachi Limited / Tom Rutt, Fujitsu Software Corporation / Yoshihide
- 1985 Nomura, Fujitsu Limited / Akira Ochi, Fujitsu Limited / Hirotaka Hara, Fujitsu Limited / Hiroyuki
- 1986 Tomisawa, Hitachi Limited / Katsuhisa Nakazato, Fujitsu Limited / Masahiko Narita, Fujitsu
- 1987 Limited / Nobuyuki Saji, NEC Corporation / Shuichi Imabayashi, Fujitsu Limited

Appendix D. Notices

1988

1989 OASIS takes no position regarding the validity or scope of any intellectual property or other rights
1990 that might be claimed to pertain to the implementation or use of the technology described in this
1991 document or the extent to which any license under such rights might or might not be available;
1992 neither does it represent that it has made any effort to identify any such rights. Information on
1993 OASIS's procedures with respect to rights in OASIS specifications can be found at the OASIS
1994 website. Copies of claims of rights made available for publication and any assurances of licenses
1995 to be made available, or the result of an attempt made to obtain a general license or permission
1996 for the use of such proprietary rights by implementors or users of this specification, can be
1997 obtained from the OASIS Executive Director.

1998 OASIS invites any interested party to bring to its attention any copyrights, patents or patent
1999 applications, or other proprietary rights which may cover technology that may be required to
2000 implement this specification. Please address the information to the OASIS Executive Director.

2001 **Copyright © OASIS Open 2003-2004. All Rights Reserved.**

2002 This document and translations of it may be copied and furnished to others, and derivative works
2003 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,
2004 published and distributed, in whole or in part, without restriction of any kind, provided that the
2005 above copyright notice and this paragraph are included on all such copies and derivative works.
2006 However, this document itself does not be modified in any way, such as by removing the
2007 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS
2008 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual
2009 Property Rights document must be followed, or as required to translate it into languages other
2010 than English.

2011 The limited permissions granted above are perpetual and will not be revoked by OASIS or its
2012 successors or assigns.

2013 This document and the information contained herein is provided on an "AS IS" basis and OASIS
2014 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO
2015 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE
2016 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A
2017 PARTICULAR PURPOSE.