



# 2 Web Services Reliable Messaging TC 3 WS-Reliability v1.1

4 OASIS Standard, 15 November 2004

5 **Document identifier:**

6 wsrn-ws\_reliability-v1.1-spec-os

7 **Location:**

8 <http://docs.oasis-open.org/wsrn/ws-reliability/v1.1>

9 **Editors:**

10 Principal editor:

11 Kazunori Iwasa, Fujitsu Limited <[kiwasa@jp.fujitsu.com](mailto:kiwasa@jp.fujitsu.com)>

12 Assisting editors:

13 Jacques Durand, Fujitsu Software Corporation <[jdurand@us.fujitsu.com](mailto:jdurand@us.fujitsu.com)>

14 Tom Rutt, Fujitsu Software Corporation <[trutt@us.fujitsu.com](mailto:trutt@us.fujitsu.com)>

15 Mark Peel, Novell, Inc. <[mpeel@novell.com](mailto:mpeel@novell.com)>

16 Sunil Kunistetty, Oracle Corporation <[Sunil.Kunistetty@oracle.com](mailto:Sunil.Kunistetty@oracle.com)>

17 Doug Bunting, Sun Microsystems <[Doug.Bunting@sun.com](mailto:Doug.Bunting@sun.com)>

18 **Abstract:**

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

23 **Status:**

24 This document is an OASIS Standard.

25 Committee members should send comments on this specification to the  
26 [wsrm@lists.oasis-open.org](mailto:wsrm@lists.oasis-open.org) list. Others should use the comment form at  
27 [http://www.oasis-open.org/committees/comments/form.php?wg\\_abbrev=wsrm](http://www.oasis-open.org/committees/comments/form.php?wg_abbrev=wsrm).

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

32 If necessary, the errata page for this version of of the specification will be located at  
33 <http://www.oasis-open.org/committees/wsrn/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

---

# 150 1 Introduction

## 151 2 Purpose of WS-Reliability

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

## 159 3 Definition and Scope of Reliable Messaging

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

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

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

170 Reliable messaging requires the definition and enforcement of contracts between:

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

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

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

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

### 185 **Note:**

186 This specification does not make any assumption about the implementation of a messaging  
187 service user component (Producer or Consumer components in **Figure 1**): such a component  
188 could be an application, a queuing or logging system, a database, a SOAP node, or the next  
189 handler in the message processing chain. The QoS contracts concern only the conditions of

190 invocation of the “Deliver”, “Submit”, “Respond” and “Notify” operations. The interpretation of  
191 these operations is a matter of implementation.

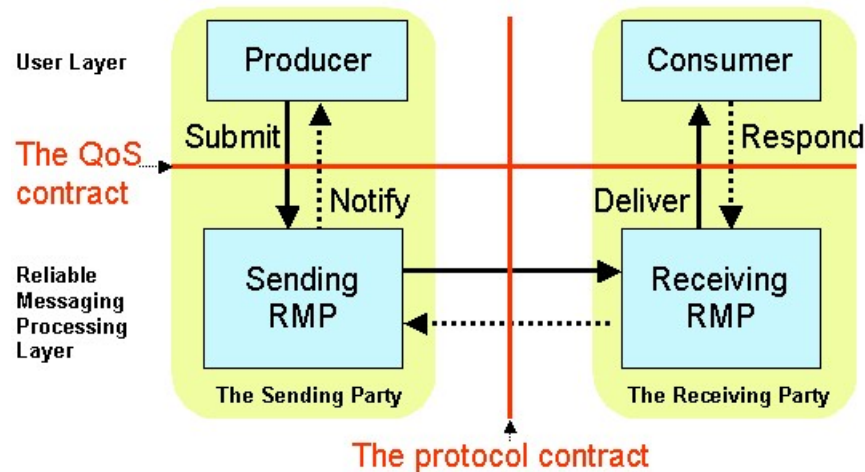


Figure 1 Reliable Messaging Contracts

192 The current specification defines the following reliability features:

- 193 • Guaranteed message delivery, or At-Least-Once delivery semantics.
- 194 • Guaranteed message duplicate elimination, or At-Most-Once delivery semantics.
- 195 • Guaranteed message delivery and duplicate elimination, or Exactly-Once delivery  
196 semantics.
- 197 • Guaranteed message ordering for delivery within a group of messages.

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

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

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

## 211 4 Notational Conventions

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

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

- 218 • examples
- 219 • 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  
 222 in 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	<a href="http://schemas.xmlsoap.org/soap/envelope/">http://schemas.xmlsoap.org/soap/envelope/</a>
soap12	<a href="http://www.w3.org/2003/05/soap-envelope">http://www.w3.org/2003/05/soap-envelope</a>
wsm	<a href="http://docs.oasis-open.org/wsm/2004/06/ws-reliability-1.1.xsd">http://docs.oasis-open.org/wsm/2004/06/ws-reliability-1.1.xsd</a>
xs	<a href="http://www.w3.org/2001/XMLSchema/">http://www.w3.org/2001/XMLSchema/</a>
wsdl11	<a href="http://schemas.xmlsoap.org/wsdl/">http://schemas.xmlsoap.org/wsdl/</a>
fnp	<a href="http://docs.oasis-open.org/wsm/2004/06/fnp-1.1.xsd">http://docs.oasis-open.org/wsm/2004/06/fnp-1.1.xsd</a>
wsmfp	<a href="http://docs.oasis-open.org/wsm/2004/06/wsmfp-1.1.xsd">http://docs.oasis-open.org/wsm/2004/06/wsmfp-1.1.xsd</a>
ref	<a href="http://docs.oasis-open.org/wsm/2004/06/reference-1.1.xsd">http://docs.oasis-open.org/wsm/2004/06/reference-1.1.xsd</a>

**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 **5 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    **6 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  
281 operations control the transfer of payload data (and, in one case, failure information) between  
282 the RMP and a user component (Producer or Consumer). An RMP operation is not necessarily  
283 implemented by an RMP, but it must be either supported in some way by an RMP or invoked by  
284 the RMP.

285 **Message Identifier:**

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

288 **Duplicate Message:**

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

290 **Message Delivery:**

291 Completion of the Deliver operation for a Reliable Message.

292 **Acknowledgment Indication:**

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

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

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

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

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

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

305 See [Section 2.5](#).

306 **PollRequest Message:**

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

309 **Intermediary:**

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

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

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

## 315 7 Messaging Model

## 316 8 Messaging Context

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

- 319 • **Intermediary transparency.** SOAP Intermediaries do not play any active role in the  
320 reliability mechanisms. They can be abstracted from the communication between  
321 Sending RMP and Receiving RMP: the RMPs are the only parties involved in  
322 implementing the RM protocol (e.g., for handling RM-Replies). There is no role for an  
323 RMP other than Receiving RMP or Sending RMP. **Figure 2** illustrates this model.
- 324 • **Message integrity.** For the reliability mechanisms described here to fulfill the reliability  
325 contract, this specification strongly RECOMMENDS that message header integrity be  
326 guaranteed end-to-end by using adequate security options such as those described in  
327 Web Services Security: SOAP Message Security 1.0 [WSS].

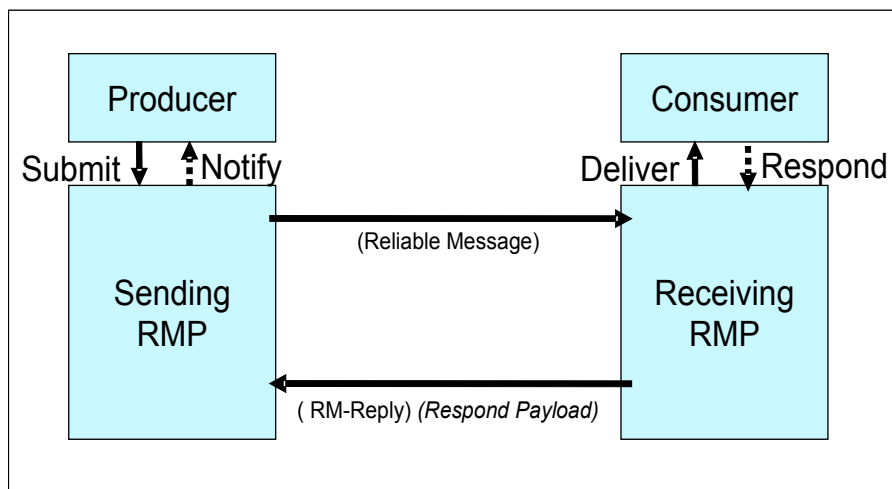


Figure 2 Messaging Model

## 328 9 RMP Operations and Their Invocation

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

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

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

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

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

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

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

## 361 **10 Binding between WSDL Operation Types and RMP Invocations**

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

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

## 374 **11 Assumed SOAP Message Exchange Patterns**

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

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

### 387 SOAP One-way MEP:

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

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

### 394 SOAP Request-response MEP:

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

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

## 400 12 Message Reply Patterns

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

## 402 13 Response RM-Reply Pattern

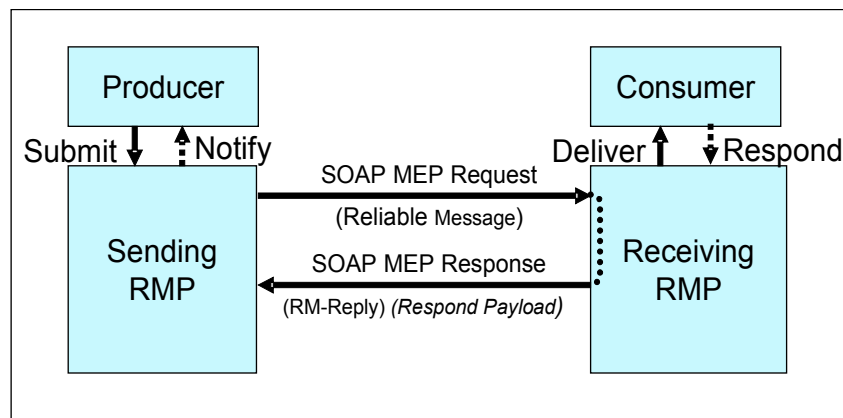
403 When the Response RM-Reply Pattern is in use, the following sequence of exchanges MUST  
404 occur:

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

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

409 **Figure 3** shows this reply pattern.

410



**Figure 3 Response RM-Reply Pattern**

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

## 413 14 Callback RM-Reply Pattern

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

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

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

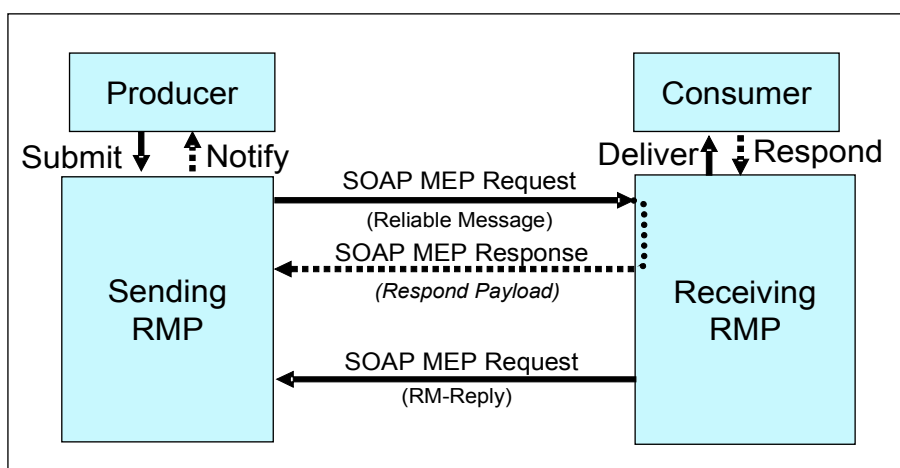


Figure 4 Callback RM-Reply Pattern

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

## 424 15 Poll RM-Reply Pattern

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

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

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

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

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



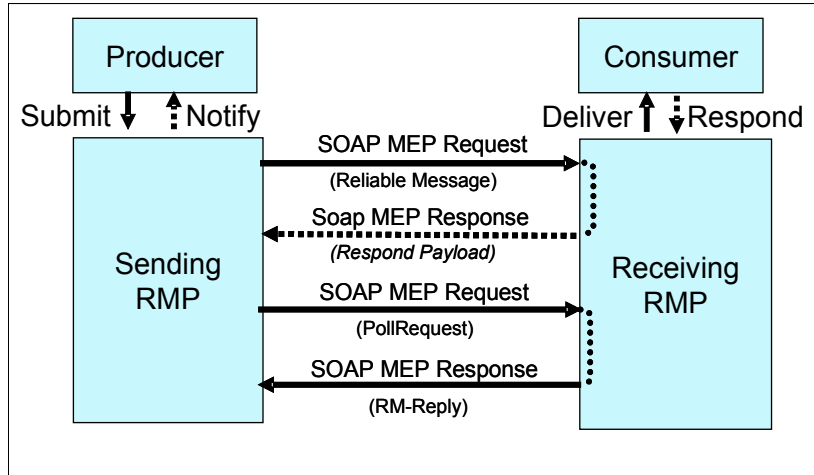


Figure 5 Synchronous Poll RM-Reply Pattern

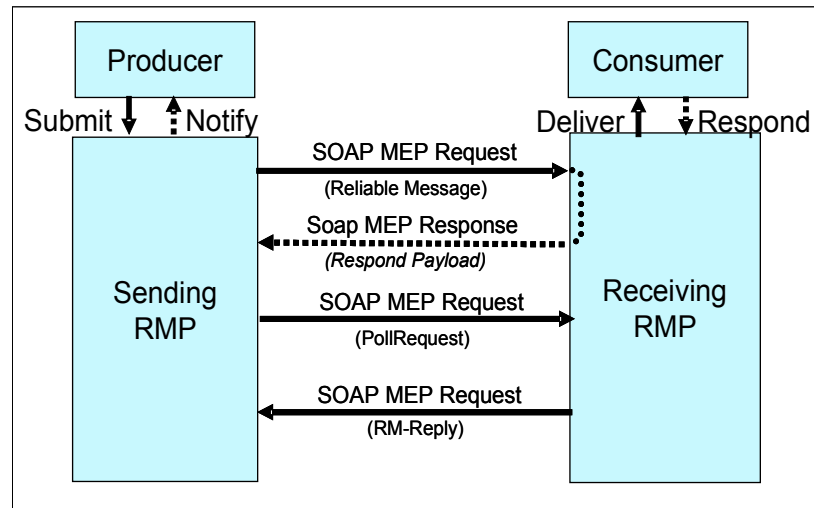


Figure 6 Asynchronous Poll RM-Reply Pattern

## 443 16 Message Identification and Grouping

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

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

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

---

## 455 **17 Reliability Agreement and Features**

### 456 **18 RM Agreement**

### 457 **19 Definition**

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

### 464 **20 RM Agreement Items**

465 An RM Agreement is a list of Agreement Items.

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

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

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

<b>Name</b>	<b>Value</b>	<b>Definition</b>
GuaranteedDelivery	enabled/disabled	For setting Guaranteed Delivery (see <b>Section 3.2.1</b> for details).
NoDuplicateDelivery	enabled/disabled	For setting message delivery without duplicates or Duplicate Elimination (see <b>Section 3.2.2</b> for details).
OrderedDelivery	enabled/disabled	For setting Guaranteed Message Ordering (see <b>Section 3.2.3</b> 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 <b>MUST NOT</b> 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**

## 475 **21 Scope of an Agreement Item**

476 There are two scopes to consider:

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

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

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

485     Message scope:

- 486     • ExpiryTime
- 487     • ReplyPattern

488     Group scope:

- 489     • OrderedDelivery
- 490     • GuaranteedDelivery
- 491     • NoDuplicateDelivery
- 492     • GroupExpiryTime

493           • GroupMaxIdleDuration

494 An RMP MUST NOT allow most Agreement items applicable at Group scope to vary between  
495 messages of a group. For example, a Sending RMP MUST NOT use different guaranteed  
496 delivery modes for different messages of a group. However, it is allowed to dynamically change  
497 the value of GroupExpiryTime or GroupMaxIdleDuration pertaining to a group (See **Section**  
498 **5.1.2**).

## 499 **22 Rules**

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

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

## 506 **23 Creation, Representation and Deployment of RM Agreements**

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

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

## 523 **24 RM Capability**

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

## 530 **25 Main Reliability Features**

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

## 534 **26 Guaranteed Delivery**

### 535 Quality of Service requirements:

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

### 540 **Notes:**

- 541 • This QoS feature guarantees only that the sender will always be notified of a delivery  
542 failure when a message is not delivered. It is, however, impossible to guarantee this  
543 while at the same time guaranteeing that (1) and (2) will never occur together for the  
544 same message. A proper usage by an implementation of the protocol options described  
545 in this specification will, however, greatly reduce situations where both (1) and (2)  
546 occur.
- 547 • The GuaranteedDelivery agreement is defined for messages resulting from invocations  
548 of the Submit operation. An extension of this agreement to messages resulting from  
549 invocations of the Respond operation is out of scope for this specification.

### 550 Protocol requirements:

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

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

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

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

## 565 **27 Duplicate Elimination**

### 566 Quality of Service requirements:

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

### 569 **Note:**

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

### 573 Protocol requirements:

574 An implementation of this specification must ensure the following invariants:

575 • Message instances resulting from separate invocations of Submit MUST NOT share the  
576 same Message Identifier.

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

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

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

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

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

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

## 591 **28 Guaranteed Message Ordering**

592 Quality of Service requirements:

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

597 **Note:**

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

602 Protocol requirements:

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

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

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

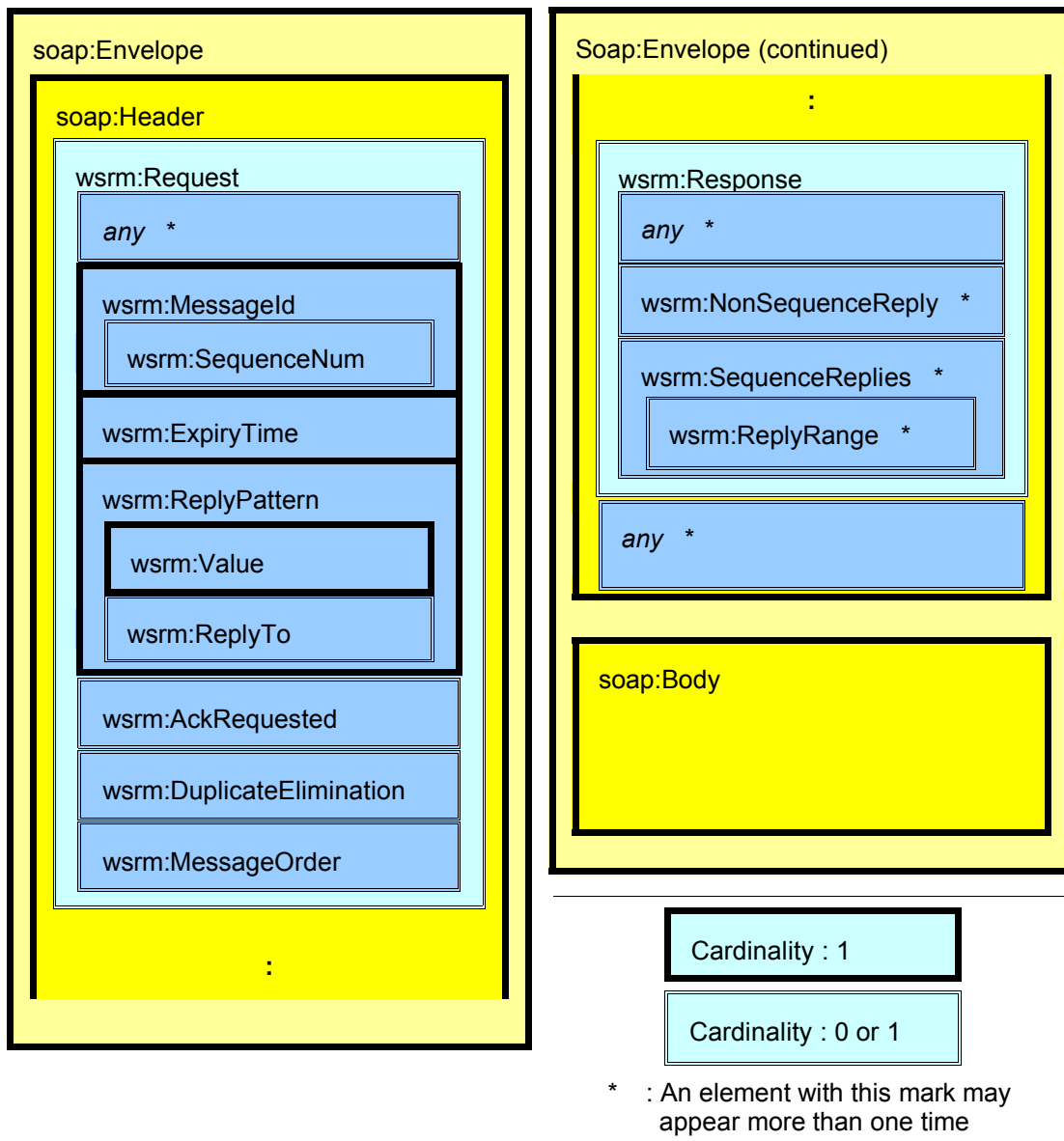
608 • The Receiving RMP MUST deliver the messages received according to the order  
609 expressed by their sequence numbers, which is the same as the submission order.

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

612 **29 Message Format**

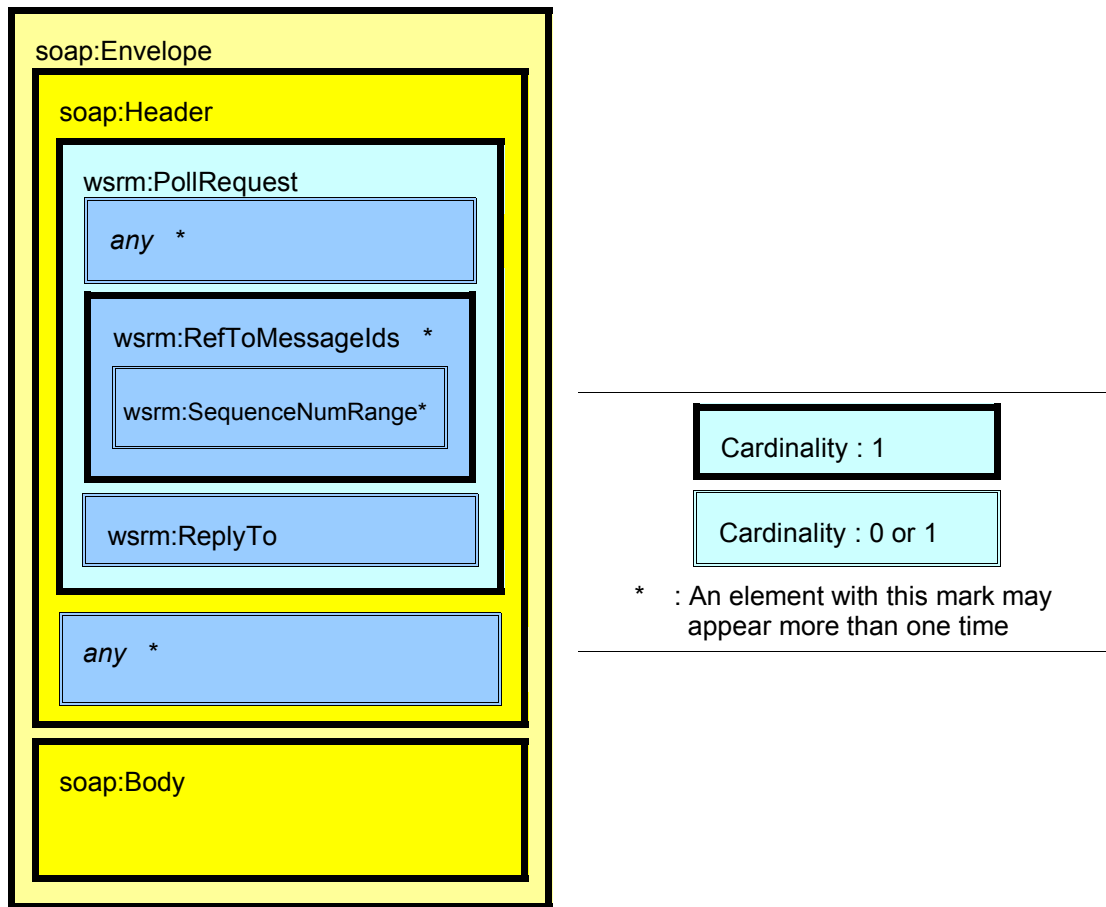
613 **30 Structure**

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



**Figure 7 Structure of WS-Reliability elements**

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



**Figure 8 Structure of PollRequest message elements**

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

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

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

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

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

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

- 630 • **Request** element
- 631 • **PollRequest** element
- 632 • **Response** element



## 633 31 Request Element

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

- 637 • SOAP **mustUnderstand** attribute (see **Appendix A** for details)
- 638 • **MessageId** element
- 639 • **ExpiryTime** element
- 640 • **ReplyPattern** element
- 641 • **AckRequested** element
- 642 • **DuplicateElimination** element
- 643 • **MessageOrder** element

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

**Table 4 Request Element**

644 **Example 1** shows an instance of a Request element.

## Example 1 Request Element

```
645 <Request
646   xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd"
647   xmlns:soap12="http://www.w3.org/2003/05/soap-envelope"
648   soap12:mustUnderstand="1">
649   <MessageId groupId="mid://20040202.103832@wsr-sender.org">
650     <SequenceNum number="0"
651       groupExpiryTime="2005-02-02T03:00:33-31:00" />
652   </MessageId>
653   <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
654   <ReplyPattern>
655     <Value>Response</Value>
656   </ReplyPattern>
657   <AckRequested/>
658   <DuplicateElimination/>
659   <MessageOrder/>
660 </Request>
```

### 661 32 Element: Request/MessageId

662 This element includes the following attribute:

- 663 • a **groupId** attribute

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

Table 5 MessageId Element

### 664 33 Attribute: Request/MessageId@groupId

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

### 670 34 Element: Request/MessageId/SequenceNum

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

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

676 This element includes the following attributes:

- 677 • a **groupExpiryTime** attribute
- 678 • a **groupMaxIdleDuration** attribute
- 679 • a **number** attribute
- 680 • a **last** attribute

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

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

**Example 2 SequenceNum Element**

684 1) First message

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

689 2) Second message

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

694 3) The last message for the group

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

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

**Table 6 SequenceNum Element**

699 **35 Attribute: Request/MessageId/SequenceNum@groupExpiryTime**

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

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

### 705 **36 Attribute: Request/MessageId/SequenceNum@groupMaxIdleDuration**

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

### 710 **37 Attribute: Request/MessageId/SequenceNum@number**

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

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

### 718 **38 Attribute: Request/MessageId/SequenceNum@last**

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

- 722 • **false**: Indicates the message is not the last message of the group or is not known to be  
723 the last message of the group.
- 724 • **true**: Indicates the message is known to be the last message sent within a group of  
725 messages.

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

### 727 **39 Element: Request/ExpiryTime**

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

Cardinality	1
Value	xs:dateTime
Attributes	None
Child elements	None

**Table 7 ExpiryTime Element**

735 **40 Element: Request/ReplyPattern**

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

- 738 • a **Value** element
- 739 • a **ReplyTo** element

Cardinality	1
Value	None
Attributes	None
Child elements	Value ReplyTo

**Table 8 ReplyPattern Element**

740 **41 Element: Request/ReplyPattern/Value**

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

- 746 • **Response**
- 747 • **Callback**
- 748 • **Poll**

749 These values respectively indicate which of the RM-Reply Patterns – Response, Callback or Poll  
750 – 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**

751 **42 Element: Request/ReplyPattern/ReplyTo**

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

757 If present, the reference-scheme attribute specifies the format of the single child element of the  
 758 ReplyTo element. If the attribute is omitted, the default content of the ReplyTo element is  
 759 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**

760 **43 Attribute: Request/ReplyPattern/ReplyTo@reference-scheme**

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

764 **44 Element: Request/ReplyPattern/ReplyTo/BareURI**

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

770 **Section 6** provides additional information about the specific case for which the content of a  
 771 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**

772 **45 Element: Request/AckRequested**

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

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

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

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

**Table 12 AckRequested Element**

## 786 **46 Element: Request/DuplicateElimination**

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

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

**Table 13 DuplicateElimination Element**

## 791 **47 Element: Request/MessageOrder**

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

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

Cardinality	0 or 1
Value	None
Attributes	None
Child elements	None

**Table 14 MessageOrder Element**

## 797 **48 Example**

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

### Example 3 Reliable Message with Request header

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

## 831 49 PollRequest Element

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

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

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

842 This element includes the following attribute and child elements:

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



845 • 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**

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

## 859 **50 Element: PollRequest/ReplyTo**

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

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

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

**Table 16 ReplyTo Element**

## 865 **51 Attribute: PollRequest/ReplyTo@reference-scheme**

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

868 **52 Element: PollRequest/ReplyTo/BareURI**

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

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

**Table 17 BareURI Element**

871 **53 Element: PollRequest/RefToMessagelds**

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

- 875 • a **groupId** attribute
- 876 • zero or more **SequenceNumRange** elements

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

**Table 18 RefToMessagelds Element**

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

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

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

887 **54 Attribute: PollRequest/RefToMessagelds@groupId**

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

890 **55 Element: PollRequest/RefToMessageIds/SequenceNumRange**

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

- 894 • a **from** attribute
- 895 • a **to** attribute

896 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**

897 **56 Attribute: PollRequest/RefToMessageIds/SequenceNumRange@from**

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

900 **57 Attribute: PollRequest/RefToMessageIds/SequenceNumRange@to**

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

904 **58 Example**

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

## Example 5 PollRequest Message embedded in HTTP Request

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

## 927 59 Response Element

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

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

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

- 933 • **NonSequenceReply** element(s)
- 934 • **SequenceReplies** element(s)

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

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

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

**Table 20 Response Element**

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

**Example 6 Response Element**

```

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

```

**60 Element: Response/NonSequenceReply**

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

- 965 • a **groupId** attribute
- 966 • a **fault** attribute

967 The @fault indicates a particular fault for the identified message. Without this attribute, the  
968 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**

969 **61 Attribute: Response/NonSequenceReply@groupid**

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

973 **62 Attribute: Response/NonSequenceReply@fault**

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

976 **63 Element: Response/SequenceReplies**

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

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

**Table 22 SequenceReplies Element**

980 **64 Attribute: Response/SequenceReplies@groupid**

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

984 **65 Element: Response/SequenceReplies/ReplyRange**

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

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

**Table 23 ReplyRange Element**

## 989 **66 Attribute: Response/SequenceReplies/ReplyRange@from**

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

## 991 **67 Attribute: Response/SequenceReplies/ReplyRange@to**

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

## 993 **68 Attribute: Response/SequenceReplies/ReplyRange@fault**

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

## 996 **69 Example**

997 The message below uses the Response reliability element, which in this case is carrying the  
998 response of a previous PollRequest element. The response acknowledges a message specified  
999 by the group identifier "mid://20040202.103811@wsr-sender.org" and messages for a group  
1000 specified by the group identifier "mid://20040202.103832@wsr-sender.org" within the ranges of  
1001 sequence numbers 0 to 14 and 16 to 20. The response also reports an RM Fault for a message  
1002 with sequence number 15 for the group.

### Example 7 RM-Reply message embedded in HTTP Response

```
1003 HTTP/1.0 200 OK
1004 Server: WS-ReliabilityServer
1005 Date: Mon, 02 Feb 2004 10:38:32 GMT
1006 Content-Language: en
1007 Content-Type: text/xml; charset=utf-8
1008 Content-Length: 593
1009
1010 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
1011   <soap:Header>
1012     <Response soap:mustUnderstand="1"
1013       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd">
1014       <NonSequenceReply groupId="mid://20040202.103811@wsr-sender.org"/>
1015       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1016         <ReplyRange from="0" to="14"/>
1017         <ReplyRange from="15" to="15" fault="InvalidRequest"/>
1018         <ReplyRange from="16" to="20"/>
1019       </SequenceReplies>
1020     </Response>
1021   </soap:Header>
1022   <soap:Body />
1023 </soap:Envelope>
```

## 1024 **70 Fault Codes For Reliable Messaging Failures**

1025 The protocol defines two fault categories:

1026 • The Message Format fault set, which includes all faults generated because of a  
1027 malformed Reliable Message header.

1028 • The Message Processing fault set, which includes all faults generated while processing  
1029 the message.

1030 They are explained in detail in the following sections. The Receiving RMP returns these  
1031 protocol-specific fault codes within the Response header element. Reliable Message Faults are  
1032 carried in the SOAP Header and do not rely exclusively on the SOAP Fault model for the  
1033 following reasons:

1034 • The SOAP Fault model does not allow batching of several faults in the same message.

1035 • RM Faults may be carried along with business messages that are unrelated to these  
1036 faults; they should not affect the processing of the SOAP body in such messages.

1037 The rules for processing faults are:

1038 • The Receiving RMP MUST NOT deliver a message for which an RM Fault is published.  
1039 Therefore, the Receiving RMP MUST NOT send an Acknowledgment Indication for  
1040 such a message.

1041 • If a Reliable Message sent over a SOAP Request-response MEP cannot be delivered to  
1042 the Consumer, the response of the SOAP MEP instance SHALL contain a SOAP Fault  
1043 (in the SOAP Body) in addition to the appropriate RM Fault (in the SOAP Header). If the  
1044 specific RM Fault encountered was due to a problem with the request header element,  
1045 the Receiving RMP MUST set the value of the soap:Fault@faultcode attribute to  
1046 "soap:Client" (for SOAP 1.1 messages) or the soap12:Fault/Code/Value element to  
1047 "soap12:Sender" (for SOAP 1.2 messages). If the specific RM Fault encountered was  
1048 due to a problem with processing by the Receiving RMP, the Receiving RMP MUST set  
1049 the value of the soap:Fault@faultcode attribute to "soap:Server" (for SOAP 1.1  
1050 messages) or the soap12:Fault/Code/Value element to "soap12:Receiver" (for SOAP  
1051 1.2 messages). The Sending RMP and Producer expect either a complete response or  
1052 a SOAP Fault when using the SOAP Request-response MEP; this requirement satisfies  
1053 those expectations. More details are given in **Section 3.2** and in the HTTP Binding  
1054 section (**Section 6**).

1055 • When a Reliable Message sent over a SOAP One-way MEP cannot be delivered to the  
1056 Consumer due to a failure in processing the RM headers, a SOAP Fault SHALL NOT be  
1057 returned. The HTTP binding section (**Section 6**) gives more details on the  
1058 recommended behavior in such case.

1059 The Fault codes described in **Sections 4.5.1** and **4.5.2** are allowed values for @fault in a  
1060 Response element.

## 1061 **71 Message Format Faults**

1062 The Receiving RMP publishes these faults when the message format of the Reliable Messaging  
1063 Headers is either invalid or wrong.



Local part name	Description and Cause(s)
InvalidRequest	<p>The Request element is wrong or invalid. Examples are:</p> <ol style="list-style-type: none"> <li>1. Any of the mandatory elements such as MessageId, ExpiryTime or ReplyPattern are missing.</li> <li>2. AckRequested, DuplicateElimination or MessageOrder elements appear twice.</li> <li>3. The soap:mustUnderstand attribute is missing.</li> </ol>
InvalidPollRequest	<p>The PollRequest element is wrong or invalid. Examples are:</p> <ol style="list-style-type: none"> <li>1. The soap:mustUnderstand attribute is missing.</li> <li>2. The RefToMessageIds element is missing.</li> </ol>
InvalidMessageId	<p>Used in any of the following cases:</p> <ol style="list-style-type: none"> <li>1. @groupId (for MessageId or RefToMessageIds) is not present or is present with an invalid value.</li> <li>2. @number in SequenceNum element is not present or is present with an invalid value.</li> <li>3. Attributes (from and to) of SequenceNumRange are not present or are present with invalid values.</li> </ol>
InvalidMessageParameters	<p>Used in any of the following cases:</p> <ol style="list-style-type: none"> <li>1. The @groupExpiryTime is wrong or invalid.</li> <li>2. The @groupMaxIdleDuration is wrong or invalid.</li> <li>3. Both group parameters are present.</li> <li>4. SequenceNum@last exists but is not one of the allowed {false true} values.</li> </ol>
InvalidReplyPattern	<p>Used in either of the following cases:</p> <ol style="list-style-type: none"> <li>1. The ReplyPattern format is wrong or invalid.</li> <li>2. The ReplyTo element is missing for the Callback pattern.</li> </ol>
InvalidExpiryTime	<p>The ExpiryTime format is wrong or invalid.</p>

**Table 24 Invalid Message Format Fault Code Values**

1064 **Note:**

- 1065 Cases exist in which the Receiving RMP is unable to send RM Fault Indications for messages  
 1066 with invalid message headers, such as:
- 1067 • The ReplyTo element is missing or invalid in the Callback and asynchronous Poll  
 1068 cases.
  - 1069 • The MessageId element is missing for the Request element.
  - 1070 • The RefToMessageIds is missing for the PollRequest element.

## 1071 72 Message Processing Faults

1072 The Receiving RMP publishes these faults when there is an error processing a valid Reliable  
 1073 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: <ol style="list-style-type: none"> <li>1. Persistence Storage failures.</li> <li>2. Message Delivery failures.</li> </ol> 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: <ol style="list-style-type: none"> <li>1. The number of buffered requests exceeded the maximum limit.</li> <li>2. The number of threads reached the maximum limit, etc.</li> <li>3. The Deliver operation fails.</li> </ol> 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**

## 1074 73 RM Fault Examples

### Example 8 RM Fault Indication for Reliable Messaging

```
1075 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
1076   <soap:Header>
1077     <Response soap:mustUnderstand="1"
1078       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd">
1079       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1080         <ReplyRange from="1" to="1" fault="InvalidRequest" />
1081       </SequenceReplies>
1082     </Response>
1083   </soap:Header>
1084   <soap:Body />
1085 </soap:Envelope>
```

1086 If the PollRequest element in **Example 4** was missing the soap:mustUnderstand attribute, the  
1087 InvalidPollRequest fault may be sent as follows.

### Example 9 RM Fault Indication for PollRequest message

```
1088 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
1089   <soap:Header>
1090     <Response soap:mustUnderstand="1"
1091       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd">
1092       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1093         <ReplyRange from="0" to="5" fault="InvalidPollRequest"/>
1094         <ReplyRange from="15" to="20" fault="InvalidPollRequest"/>
1095       </SequenceReplies>
1096       <NonSequenceReply groupId="mid://20040202.103811@wsr-sender.org"
1097         fault="InvalidPollRequest"/>
1098       <SequenceReplies groupId="mid://20040202.103807@wsr-sender.org/">
1099         <ReplyRange from="713" to="6150" fault="InvalidPollRequest"/>
1100       </SequenceReplies>
1101     </Response>
1102   </soap:Header>
1103   <soap:Body />
1104 </soap:Envelope>
```

## 1105 74 Extensibility Features of Schema

1106 The core schema for this specification (associated in **Section 1.3, Table 2**, with the “wsrm”  
1107 namespace prefix) specifies extension mechanisms for some schema elements.

1108 The following elements (which have a complex sequence type) allow the presence of zero or  
1109 more extension elements (of type xs:anyType; that is, any type not defined in this core  
1110 namespace is allowed) at the beginning of the sequence, as well as zero or more extension  
1111 attributes (with similar namespace restrictions):

- 1112 • Request

- 1113       • Response
- 1114       • PollRequest
- 1115       • NonSequenceReply
- 1116       • SequenceReplies
- 1117       • ReplyRange

1118   The extensibility of the ReplyTo elements (**Sections 4.2.3.2** and **4.3.1**) is somewhat different; it  
1119   is described in the appropriate sections above.

---

## 1120 75 Operational Aspects and Semantics

### 1121 76 Message Group Life Cycle

### 1122 77 Group Termination

1123 Being able to know when a group may be terminated and its persistent resources reclaimed is  
1124 essential for keeping the resource footprint of reliability low. However, this section is not just  
1125 about efficient management of resources: it describes normative behavioral rules for RMPs when  
1126 handling group termination.

1127 Termination of a group in the Sending RMP and in the Receiving RMP are two distinct events,  
1128 not synchronized by any special message but instead occurring as the result of rules applying  
1129 separately to the Sending and Receiving RMPs. As a consequence, the termination of a group  
1130 may occur at quite different times on the Sending RMP and the Receiving RMP. However, the  
1131 lack of synchronization allowed by these termination rules is not consequential.

1132 Groups undergoing termination on the Sending RMP and the Receiving RMP pass through the  
1133 following states:

#### 1134 **Group complete:**

1135       • The Sending RMP considers a group complete when all of its messages have been  
1136       sent and the last sent message has an ending marker (SequenceNum@last="true" or it  
1137       has a sequence number with the maximum value). Note that completeness occurs even  
1138       if not all of the group's messages have been either acknowledged or faulted (in case  
1139       GuaranteedDelivery is enabled).

1140       • The Receiving RMP considers a group complete when a message with an ending  
1141       marker has been received and all previous messages for this group also have been  
1142       received (no number missing in the sequence) although not necessarily delivered yet.

#### 1143 **Group closed:**

1144       • When a group is closed in the Sending RMP, the RMP expects to send no new  
1145       message in this group. However, the RMP MAY resend messages as needed if  
1146       GuaranteedDelivery is enabled. If a new message is submitted for a closed group, the  
1147       Sending RMP MUST notify the Producer that the group is closed and MUST NOT send  
1148       the message.

1149       • When a group is closed in the Receiving RMP, the RMP expects to receive no new  
1150       message for this group. After a group is closed and before it is "removed" (see  
1151       definition below), a Receiving RMP MUST NOT deliver messages received with this  
1152       group identifier, whether or not they are duplicates of previous messages and  
1153       regardless of whether they result from a resend of previously failed messages initiated  
1154       before closing on the Sending RMP (in case GuaranteedDelivery is enabled).

#### 1155 **Note:**

1156 Due to time-out, a group may be closed without being complete. Once complete, a group will  
1157 close (see termination rules).

## 1158 **Group Removed:**

1159 Group removal occurs at the time the group is closed or afterward. Intuitively, a group is removed  
1160 when a Receiving RMP does not need to remember anything about this group, i.e., when there is  
1161 no need to check for duplicates of its messages in the future (for example, when all of its  
1162 messages have expired).

1163 • When a group is removed in the Sending RMP, the RMP is NOT REQUIRED to verify  
1164 that future submitted messages are improperly associated with the removed group and  
1165 MAY treat them as part of a new group. However, the Sending RMP is responsible for  
1166 generating group identifiers, and it SHOULD generate values unique enough to avoid  
1167 later reuse of the group identifier of a removed group (for example, generation  
1168 mechanisms including a timestamp will make reuse impossible).

1169 • When a group is removed in the Receiving RMP, the RMP is no longer supposed to  
1170 remember anything about this group. In particular, the group identifier is discarded from  
1171 the RMP state. When receiving a message with same group identifier as a removed  
1172 group, a Receiving RMP is NOT REQUIRED to confirm whether or not this group  
1173 identifier value has already been used; the RMP MAY treat such a message as part of a  
1174 new group.

## 1175 **78 Group Termination Parameters**

1176 Two RM Agreement Items, GroupExpiryTime and GroupMaxIdleDuration, determine when a  
1177 group can be terminated. These two items are considered Group Termination parameters that  
1178 control the persistence of the group data. The corresponding message header attributes are  
1179 @groupExpiryTime and @groupMaxIdleDuration respectively. The following requirements  
1180 pertain to these header attributes:

1181 a) The first message in a group (the one with  
1182 Request/MessageId/SequenceNum@number=0) indicates which Group Termination (time-  
1183 out) parameter is in use for the group. However, the Receiving RMP MUST use the first  
1184 message received for this group to indicate which termination parameter is associated with  
1185 this group.

1186 • If the first message in the sequence of a group has neither group time-out parameter  
1187 present, the group will be terminated according to condition T3, T4 or T5.

1188 • If the first message has one of the two time-out parameters present (either  
1189 @groupExpiryTime or @groupMaxIdleDuration), the group will be subject to  
1190 termination rules T1 or T2 described below.

1191 • The Receiving RMP MUST return an InvalidMessageParameters fault if both group  
1192 persistence parameters are present in any request message.

1193 • If @groupExpiryTime is in use, the Sending RMP MUST NOT send a message in that  
1194 group with an ExpiryTime value greater than @groupExpiryTime.

1195 b) The group termination parameter sent on the first message in the group SHALL be used  
1196 on all subsequent messages in that group and SHALL be assigned a value.

1197 c) If the Receiving RMP receives a message with a group termination parameter that is not  
1198 consistent with the termination parameter used in previous messages for this group, the  
1199 Receiving RMP MUST return an InvalidMessageParameters fault.

1200 When the group is ordered, the fault SHALL be returned for the message with lowest  
1201 sequence number that was found inconsistent in the group. If the group is not required to

1202 be ordered, the fault SHALL be returned for the first message received that was found  
1203 inconsistent in the group.

1204 d) The Sending RMP MAY modify either time-out parameter, sending a subsequent  
1205 message with the new value. When applying termination rules, the Sending RMP MUST  
1206 use the value in the message with the highest sequence number sent for the group. The  
1207 Receiving RMP MUST use the value from the message with the highest sequence number  
1208 received for the group.

1209 e) @groupMaxIdleDuration can be either increased or decreased without restriction. The  
1210 Sending RMP may increase or decrease @groupExpiryTime as long as it is never less  
1211 than the max(ExpiryTime) of the messages sent for the group so far.

1212 The Receiving RMP MUST publish an InvalidMessageParameters Fault for a message with  
1213 a @groupExpiryTime value less than the max(ExpiryTime) of the messages previously  
1214 received for the group.

## 1215 **79 Termination Rules**

1216 Termination is the process by which an RMP discontinues the use of a group, allowing the RMP  
1217 to reclaim resources used by the group. Termination typically involves two steps that may occur  
1218 at different times: closing and removal. Removal of a group may happen some time after it is  
1219 closed, allowing an RMP to filter out potential duplicate messages. The general rule is that a  
1220 group is removed once all of its messages have expired. If we define max(ExpiryTime) as the  
1221 maximum date and time of all ExpiryTime values of the messages sent for a group (on the  
1222 Sender side) or received for a group (on the Receiver side), a group will not be removed before  
1223 max(ExpiryTime) occurs.

1224 There are two general indicators an RMP will use to terminate a group:

- 1225 a) Message Marker: Information within a message (either  
1226 Request/MessageId/SequenceNum@last="true" or the maximum sequence number)  
1227 indicates the last message for the group. This is used by termination rules T3, T4.
- 1228 b) Timing: Either the group's lifespan expired or its idle time exceeded a time-out. This is  
1229 used by termination rules T1, T2. Or due to message expiration, a group with the ordering  
1230 requirement cannot be delivered. This is used by termination rule T5.

1231 These termination rules apply to both ordered and unordered groups. However, these rules do  
1232 not apply to groups that contain a single message with no sequence number.

## 1233 **80 Termination by expiration (T1):**

1234 Context:

1235 The group specified @groupExpiryTime.

1236 Receiver side:

1237 Triggering event: @groupExpiryTime is in the past.

1238 The RMP MUST close and remove the group.

1239 Sender side:

1240 Triggering event: @groupExpiryTime is in the past (note: in this case, max(ExpiryTime) also is  
1241 past).

1242 The RMP MUST close and remove the group.

## 1243 **81 Termination by idle time-out (T2):**

### 1244 Context:

1245 The group specified @groupMaxIdleDuration.

### 1246 Receiver side:

1247 Triggering event: The time since the last received message for the group is over  
1248 @groupMaxIdleDuration.

1249 The RMP MUST close the group. But unlike T1, some of its past messages may not have  
1250 expired yet. In case Duplicate Elimination is required, the RMP MUST NOT remove the group  
1251 until max(ExpiryTime) is reached in order to make sure all potential duplicates for the group will  
1252 not be delivered.

### 1253 Sender side:

1254 Triggering event: The time since the last sent message for the group is over  
1255 @groupMaxIdleDuration.

1256 The RMP MUST close the group. If GuaranteedDelivery was required, the RMP MUST remove  
1257 the group once it has received either acknowledgment or notification of delivery failure for all  
1258 sent messages. If no GuaranteedDelivery was required, the RMP MUST remove the group  
1259 immediately.

## 1260 **82 Termination by completeness (T3):**

### 1261 Context:

1262 No specific context.

### 1263 Receiver side:

1264 Triggering event: The RMP receives a message marked last  
1265 (Request/MessageId/SequenceNum@last="true"). If all previous messages for the group have  
1266 been received, the group is closed immediately. Alternately, the group is closed when the RMP  
1267 receives the last missing message in the group.

1268 The RMP MUST close the group. However, its removal is done according to T1 or T2 depending  
1269 on which time-out parameter was specified for the group. If no time-out parameter was specified,  
1270 the group is removed once all of its messages have expired, i.e., the date and time  
1271 max(ExpiryTime) has passed.

### 1272 **Note:**

1273 In the case in which a message is received with an ending marker before all previous messages  
1274 have been received, the group remains active. No termination process is initiated yet.

### 1275 Sender side:

1276 Triggering event: The RMP sends a message marked last.

1277 All messages of the group have been sent. The RMP MUST close the group. If  
1278 GuaranteedDelivery was required, the RMP MUST remove the group once it has received either  
1279 acknowledgment or notification of delivery failure for all sent messages. If GuaranteedDelivery  
1280 was not required, the RMP MUST remove the group immediately.



### 1281 **83 Termination by sequence exhaustion (T4):**

1282 Context:

1283 No specific context.

1284 Receiver side:

1285 Triggering event: The RMP receives a message with a sequence number of the maximum value.  
1286 If all previous messages for the group have been received, the group is closed immediately.  
1287 Alternately, the group is closed when the RMP receives the last missing message in the group.

1288 The group closing and removal follow the rules in T3, the message with the maximum sequence  
1289 number acting as a message with the ending mark.

1290 **Note:**

1291 In case a message is received with the maximum sequence number before all previous  
1292 messages have been received, the group remains active. No termination process is initiated yet.

1293 Sender side:

1294 Triggering event: The RMP sends a message with a sequence number with the maximum value.

1295 The group closing and removal follow the rules in T3, the message with the maximum sequence  
1296 number acting as a message with the ending mark.

### 1297 **84 Termination by ordering failure (T5):**

1298 Context:

1299 The group requires the Guaranteed Message Ordering reliability feature.

1300 Receiving side:

1301 Triggering event: In an ordered group, a received message expires before delivery or faults with  
1302 a fault code other than MessageProcessingFailure. If all previous messages for the group have  
1303 been received, the group is closed immediately. Alternately, the group is closed when the RMP  
1304 receives the last missing message in the group.

1305 The RMP MUST close the group. The group is removed according to rule T3.

1306 Sender Side:

1307 Triggering event: In an ordered group, an unacknowledged message expires or the RMP  
1308 receives an RM Fault for this Reliable Message with a fault code other than  
1309 MessageProcessingFailure.

1310 The RMP MUST close the group. The group is removed according to rule T3.

### 1311 **85 Summary of Group Termination Rules**

1312 Conditions for terminating a group in a Receiving RMP:

<b>Group Closing</b>	<b>Group Removal</b>
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**

1313 Conditions for terminating a group in a Sending RMP:

<b>Group Closing</b>	<b>Group Removal</b>
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**

## 1314 **86 Attachments**

- 1315 When an RMP implementing this specification uses the W3C Note “SOAP Messages with  
1316 Attachments” specification [SOAP with Attachments], it MUST follow the following rules:
- 1317 1) The Sending RMP MUST include the whole SOAP envelope containing the WS-  
1318 Reliability header elements in the first MIME part.
  - 1319 2) It MUST set the charset parameter of the Content-Type header of the first MIME part to  
1320 either UTF-8 or UTF-16.
  - 1321 3) It MAY include zero or more additional MIME parts in a Reliable Message.
  - 1322 4) The Receiving RMP MUST deliver all MIME parts in a Reliable Message to the  
1323 Consumer.

---

## 1324 87 HTTP Binding

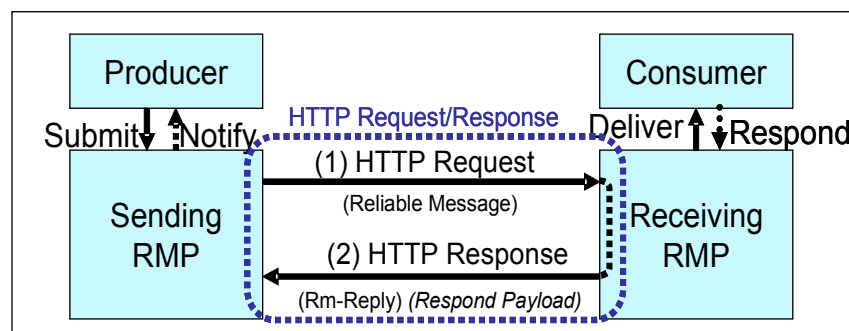
- 1325 This section specifies two normative bindings of WS-Reliability header elements to SOAP  
1326 header blocks carried in messages using HTTP as a transport protocol:
- 1327 • SOAP 1.1 over HTTP POST binding: An implementation of WS-Reliability MAY support  
1328 mapping the WS-Reliability header elements as SOAP header blocks in accordance  
1329 with the SOAP 1.1 HTTP Binding specified in Section 6 of [SOAP 1.1]. In that case, the  
1330 SOAP Request-response MEP defined in this specification will map to an HTTP  
1331 request-response. The SOAP One-way MEP, as defined in **Section 2.3**, maps to the  
1332 request of an HTTP request-response.
  - 1333 • SOAP 1.2 over HTTP POST binding: An implementation of WS-Reliability MAY support  
1334 mapping the WS-Reliability header elements as SOAP header blocks in accordance  
1335 with the SOAP 1.2 HTTP binding for the Request-Response MEP specified in Section 7,  
1336 "SOAP HTTP Binding", of [SOAP 1.2 Part 2].
- 1337 If a Reliable Message request is invoked using SOAP 1.1, all subsequent message exchanges  
1338 pertaining to that Message Identifier MUST use the SOAP 1.1 protocol. In addition, when an  
1339 HTTP binding is used, it is RECOMMENDED the RMP comply with WS-I BP 1.1 [WS-I BP 1.1].  
1340 When no WSDL describes the messages being exchanged, the previous WS-I conformance  
1341 requirements should be understood as conformance to the subset of the profile requirements  
1342 pertaining to the message artifact only.
- 1343 In case a message encounters a failure in processing the RM headers, the requirements for  
1344 Fault handling in **Section 4.5** apply. When using SOAP 1.1, conformance to the WS-I Basic  
1345 Profile 1.1 requires the following:
- 1346 • For SOAP One-way HTTP binding: the HTTP response entity-body SHALL be empty. If  
1347 the RM Fault is a Message Format fault, the HTTP status code SHOULD be "400 Bad  
1348 Request" (see R1113 in [WS-I BP 1.1]); otherwise, the RM fault is a Message  
1349 Processing fault and the status code SHOULD be "500 Internal Server Error".
  - 1350 • For SOAP Request-response HTTP binding: the HTTP response contains a SOAP  
1351 Fault element and has the "500 Internal Server Error" HTTP status code (see R1126 in  
1352 [WS-I BP 1.1]).
- 1353 These two requirements for Fault handling apply to all message exchanges described in this  
1354 section and its sub-sections.
- 1355 If a ReplyTo element present in a Request element or Poll Request header element sent using  
1356 the SOAP 1.1 protocol uses the wsrn:BareURI (the default, described in **Sections 4.2.3.2.2** and  
1357 **4.3.1.2**) reference scheme and uses the 'http:' URL scheme, the Receiving RMP MUST send the  
1358 WS-Reliability response using the HTTP binding specified in Section 6 of SOAP 1.1.
- 1359 If a Reliable Message request is invoked using SOAP 1.2, all subsequent message exchanges  
1360 pertaining to its Message Identifier MUST use the SOAP 1.2 protocol.
- 1361 If a ReplyTo element present in a Request element or Poll Request header element sent using  
1362 the SOAP 1.2 protocol uses the wsrn:BareURI reference scheme and uses the 'http:' URL  
1363 scheme, the the Receiving RMP MUST send the WS-Reliability response using the HTTP  
1364 binding for Request-Response MEP specified in SOAP 1.2.
- 1365 The following subsections specify the mapping of WS-Reliability header elements to HTTP  
1366 request and response messages for the three RM-Reply Patterns. The Poll RM-Reply Pattern  
1367 has two variations: synchronous and asynchronous.

1368 The value of the ReplyPattern/Value element identifies the specific RM-Reply Pattern in use (see  
1369 **Section 4.2.3.1** for details).

1370 This specification requires the transport layer to deliver messages to the reliability layer without  
1371 corruption. When a request message contains the AckRequested element, the Receiving RMP  
1372 MUST send an RM-Reply (an Acknowledgment Indication or an RM Fault Indication) for that  
1373 request. For the Callback and Poll RM-Reply Patterns, a Response element can contain multiple  
1374 Acknowledgment and/or RM Fault Indications.

1375 For simplicity, the detailed examples show only the use of SOAP 1.1. However, the figures that  
1376 show the mapping of WS-Reliability elements to HTTP POST request messages and HTTP  
1377 response messages apply to both the SOAP 1.1 over HTTP POST binding and the SOAP 1.2  
1378 over HTTP POST binding.

## 1379 **88 Reliable Messaging with Response RM-Reply Pattern**



**Figure 9 Response RM-Reply Pattern**

1380 As described in general for this RM-Reply Pattern (**Section 2.4.1**), the Receiving RMP MUST  
1381 return the RM-Reply with the HTTP response on the same HTTP connection used by the  
1382 Sending RMP to send the request. This is illustrated in **Figure 9**.

- 1383
- In (1), the Sending RMP initiates an HTTP connection and sends a Message using the  
1384 HTTP POST method, as in **Example 10**.
- 1385
- In (2), using the same connection, the Receiving RMP sends back to the Sending RMP  
1386 an HTTP response containing an RM-Reply; in **Example 11**, the RM-Reply is an  
1387 Acknowledgment Indication.

### Example 10 Request Message with Response RM-Reply Pattern

```
1388 POST /abc/servlet/wsrEndpoint HTTP/1.0
1389 Content-Type: text/xml; charset=utf-8
1390 Host: 192.168.183.100
1391 SOAPAction: ""
1392 Content-Length: 755
1393
1394 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1395   <soap:Header>
1396     <Request
1397       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
1398       soap:mustUnderstand="1">
1399       <MessageId groupId="mid://20040202.103832@wsr-sender.org">
1400         <SequenceNum number="0"
1401           groupExpiryTime="2005-02-02T03:00:33-31:00" />
1402       </MessageId>
1403       <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
1404       <ReplyPattern>
1405         <Value>Response</Value>
1406       </ReplyPattern>
1407       <AckRequested/>
1408       <DuplicateElimination/>
1409       <MessageOrder/>
1410     </Request>
1411   </soap:Header>
1412   <soap:Body>
1413     <Request xmlns="http://example.org/wsr">Request Message</Request>
1414   </soap:Body>
1415 </soap:Envelope>
```

### Example 11 Acknowledgment Indication with Response RM-Reply Pattern

```
1416 HTTP/1.0 200 OK
1417 Server: WS-ReliabilityServer
1418 Date: Mon, 02 Feb 2004 10:38:32 GMT
1419 Content-Language: en
1420 Content-Type: text/xml; charset=utf-8
1421 Content-Length: 414
1422
1423 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1424   <soap:Header>
1425     <Response soap:mustUnderstand="1"
1426       xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd">
1427       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1428         <ReplyRange from="0" to="0"/>
1429       </SequenceReplies>
1430     </Response>
1431   </soap:Header>
1432   <soap:Body />
1433 </soap:Envelope>
```

### 1434 89 Reliable Messaging with Callback RM-Reply Pattern

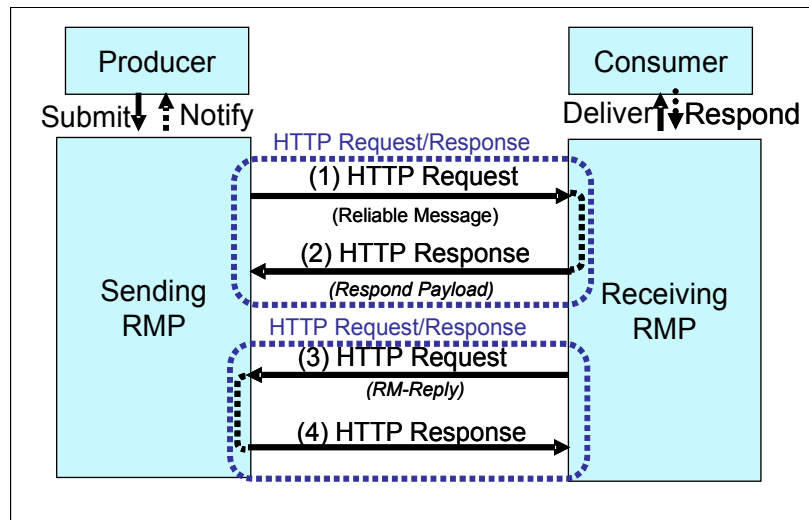


Figure 10 Callback RM-Reply Pattern

1435 As described in general for this RM-Reply Pattern (Section 2.4.2) and as illustrated in Figure  
1436 10, two distinct HTTP request/response exchanges are involved.

- 1437 • In (1), the Sending RMP initiates a new HTTP request and sends a Reliable Message  
1438 with the Callback RM Reply Pattern. Example 12 shows such an HTTP message.
- 1439 • In (2), the HTTP response may have an empty entity-body (in case of a SOAP One-way  
1440 MEP instance).
- 1441 • In (3), the Receiving RMP MUST return the RM-Reply on an HTTP connection different  
1442 from the one the Sending RMP used to send the message. The direction of the HTTP

- 1443 connection used by the Receiving RMP is from the Receiving RMP to the Sending  
1444 RMP. **Example 14** shows an Acknowledgment Indication as the RM-Reply.
- 1445 • In (4), there is no HTTP entity-body unless the RM-Reply was bundled with a new  
1446 Reliable Message on a SOAP Request-response MEP instance.

#### Example 12 Request Message with Callback RM-Reply Pattern

```
1447 POST /abc/servlet/wsrEndpoint HTTP/1.0
1448 Content-Type: text/xml; charset=utf-8
1449 Host: 192.168.183.100
1450 SOAPAction: ""
1451 Content-Length: 863
1452
1453 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1454   <soap:Header>
1455     <Request
1456       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
1457       soap:mustUnderstand="1">
1458       <MessageId groupId="mid://20040202.103832@wsr-sender.org">
1459         <SequenceNum number="0"
1460           groupExpiryTime="2005-02-02T03:00:33-31:00" />
1461       </MessageId>
1462       <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
1463       <ReplyPattern>
1464         <Value>Callback</Value>
1465         <ReplyTo>
1466           <BareURI>http://wsr-sender.org/abc/wsrmlistener</BareURI>
1467         </ReplyTo>
1468       </ReplyPattern>
1469       <AckRequested/>
1470       <DuplicateElimination/>
1471       <MessageOrder/>
1472     </Request>
1473   </soap:Header>
1474   <soap:Body>
1475     <Request xmlns="http://example.org/wsr">Request Message</Request>
1476   </soap:Body>
1477 </soap:Envelope>
```

#### Example 13 HTTP response with no content

```
1478 HTTP/1.0 200 OK
1479 Server: WS-ReliabilityServer
1480 Date: Mon, 02 Feb 2004 10:38:32 GMT
1481 Content-Language: en
1482 Content-Type: text/xml; charset=utf-8
1483 Content-Length: 0
```

### Example 14 Acknowledgment Indication with Callback RM-Reply Pattern

```
1484 POST /abc/wsrmlistener HTTP/1.0
1485 Content-Type: text/xml; charset=utf-8
1486 Host: 192.168.183.200
1487 SOAPAction: ""
1488 Content-Length: 414
1489
1490 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
1491   <soap:Header>
1492     <Response soap:mustUnderstand="1"
1493       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd">
1494       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1495         <ReplyRange from="0" to="0"/>
1496       </SequenceReplies >
1497     </Response>
1498   </soap:Header>
1499   <soap:Body />
1500 </soap:Envelope>
```

## 1501 90 Reliable Messaging with Poll RM-Reply Pattern

1502 The general rules for this RM-Reply Pattern are described in **Section 2.4.3**. When the Sending  
1503 RMP issues a PollRequest, the Receiving RMP MAY return the RM-Reply on the HTTP  
1504 connection used to send the PollRequest message (synchronous), or it MAY return the RM-  
1505 Reply on a different HTTP connection (asynchronous). Whether the RM-Reply corresponding to  
1506 the PollRequest is synchronous or asynchronous depends on the presence of a ReplyTo  
1507 element in the PollRequest element.

## 1508 91 Synchronous Poll RM-Reply Pattern

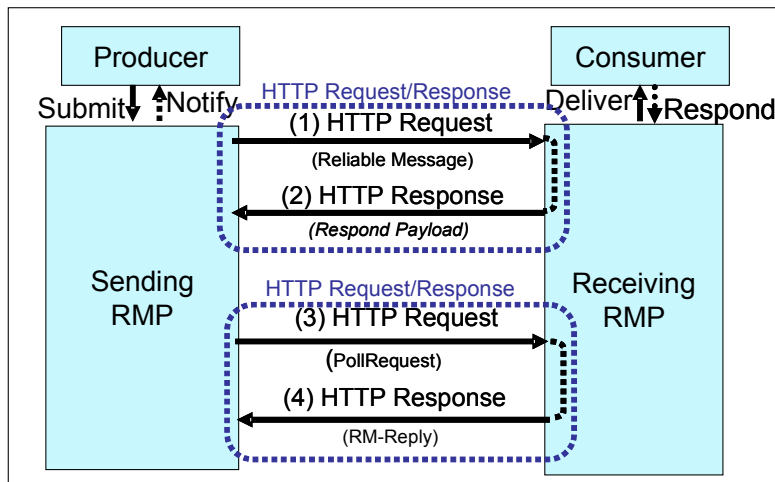


Figure 11 Synchronous Poll RM-Reply Pattern

1509 **Figure 11** illustrates the synchronous variant of the Poll RM Reply Pattern.

- 1510 • In (1), the Sending RMP initiates a new HTTP Request and sends a Reliable Message  
1511 with the Poll RM-Reply Pattern.



- 1512 • In (2), the HTTP response may have an empty entity-body (in case of a SOAP One-way  
1513 MEP instance).
- 1514 • In (3), at a later time the Sending RMP initiates a different HTTP Request to send a  
1515 PollRequest message. The PollRequest does not include the ReplyTo element (see  
1516 **Example 15**).
- 1517 • In (4), the Receiving RMP returns the RM-Reply in an HTTP response on the same  
1518 HTTP connection used to send the PollRequest, as illustrated in **Figure 11**. The HTTP  
1519 response (4) includes an RM-Reply (e.g., an Acknowledgment Indication as in **Example**  
1520 **16**).

#### Example 15 PollRequest message with Synchronous Poll RM-Reply Pattern

```
1521 POST /abc/servlet/wsrmlistener HTTP/1.0
1522 Content-Type: text/xml; charset=utf-8
1523 Host: 192.168.183.100
1524 SOAPAction: ""
1525 Content-Length: 433
1526
1527 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1528   <soap:Header>
1529     <PollRequest
1530       xmlns="http://docs.oasis-open.org/wsrmlistener/2004/06/ws-reliability-1.1.xsd"
1531       soap:mustUnderstand="1">
1532       <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
1533         <SequenceNumRange from="0" to="20"/>
1534       </RefToMessageIds>
1535     </PollRequest>
1536   </soap:Header>
1537   <soap:Body />
1538 </soap:Envelope>
```

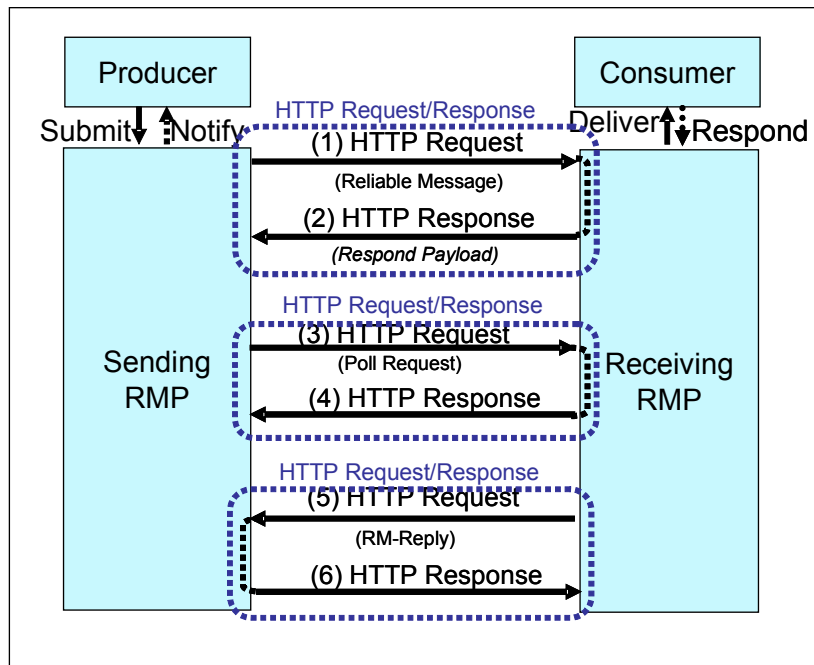
## Example 16 Synchronous Acknowledgment Indication

```

1539 HTTP/1.0 200 OK
1540 Server: WS-ReliabilityServer
1541 Date: Mon, 02 Feb 2004 10:38:32 GMT
1542 Content-Language: en
1543 Content-Type: text/xml; charset=utf-8
1544 Content-Length: 456
1545
1546 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1547   <soap:Header>
1548     <Response soap:mustUnderstand="1"
1549       xmlns="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd">
1550       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1551         <ReplyRange from="0" to="14"/>
1552         <ReplyRange from="16" to="20"/>
1553       </SequenceReplies>
1554     </Response>
1555   </soap:Header>
1556   <soap:Body />
1557 </soap:Envelope>

```

## 1558 92 Asynchronous Poll RM-Reply Pattern



**Figure 12 Asynchronous Poll RM-Reply Pattern**

1559 **Figure 12** illustrates the asynchronous variant of the Poll RM Reply Pattern.

- 1560 • In (1), the Sending RMP initiates a new HTTP Request and sends a Reliable Message
- 1561 with the Poll RM-Reply Pattern.

- 1562 • In (2), the HTTP response may have an empty entity-body (in the case of a SOAP One-  
1563 way MEP instance).
- 1564 • In (3), the Sending RMP initiates a new HTTP request and sends a PollRequest  
1565 message. Note that in **Example 17**, the PollRequest element has a ReplyTo element.
- 1566 • In (4), the HTTP response (4) has no HTTP entity-body (see **Example 13**).
- 1567 • In (5), the Receiving RMP sends the RM-Reply in a different HTTP request to the  
1568 listener identified by the ReplyTo element (see **Example 18**).
- 1569 • In (6), the HTTP response has no HTTP entity-body (see **Example 13**).

#### Example 17 PollRequest message with Asynchronous Poll RM-Reply Pattern

```

1570 POST /abc/servlet/wsrmlistener HTTP/1.0
1571 Content-Type: text/xml; charset=utf-8
1572 Host: 192.168.183.100
1573 SOAPAction: ""
1574 Content-Length: 553
1575
1576 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1577   <soap:Header>
1578     <PollRequest
1579       xmlns="http://docs.oasis-open.org/wsr/2004/06/ws-reliability-1.1.xsd"
1580       soap:mustUnderstand="1">
1581       <RefToMessageIds groupId="mid://20040202.103832@wsr-sender.org">
1582         <SequenceNumRange from="0" to="20"/>
1583       </RefToMessageIds>
1584       <ReplyTo>
1585         <BareURI>http://wsr-sender.org/xyz/servlet/wsrmlistener
1586         </BareURI>
1587       </ReplyTo>
1588     </PollRequest>
1589   </soap:Header>
1590   <soap:Body />
1591 </soap:Envelope>

```

## Example 18 Asynchronous Acknowledgment Indication

```
1592 POST /xyz/servlet/wsrmlistener HTTP/1.0
1593 Content-Type: text/xml; charset=utf-8
1594 Host: 192.168.183.200
1595 SOAPAction: ""
1596 Content-Length: 456
1597
1598 <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/" >
1599   <soap:Header>
1600     <Response soap:mustUnderstand="1"
1601       xmlns="http://docs.oasis-open.org/wsrml/2004/06/ws-reliability-1.1.xsd">
1602       <SequenceReplies groupId="mid://20040202.103832@wsr-sender.org">
1603         <ReplyRange from="0" to="14"/>
1604         <ReplyRange from="16" to="20"/>
1605       </SequenceReplies>
1606     </Response>
1607   </soap:Header>
1608   <soap:Body />
1609 </soap:Envelope>
```

---

## 1610 93 Conformance

- 1611 In order to conform to this specification, an implementation must satisfy all of the following  
1612 conditions:
- 1613 • It has implemented all required syntax, features and behaviors.
  - 1614 • It complies with the following interpretation of the keywords OPTIONAL and MAY: as  
1615 stated in [RFC2119], when these keywords apply to the behavior of the implementation,  
1616 the implementation is free to support these behaviors or not.
  - 1617 • It MUST be capable of processing the prescribed failure mechanism for those optional  
1618 features it has chosen to implement. If an RMP conforming to this requirement has  
1619 implemented an optional feature, syntax or behavior defined in this specification, it can  
1620 interoperate with another implementation that has not.
  - 1621 • It MUST be capable of generating the prescribed failure mechanism for those optional  
1622 features it has not chosen to implement. If an RMP conforming to this requirement has  
1623 not implemented an optional feature, syntax or behavior defined in this specification, it  
1624 can interoperate with another implementation that has.

---

## 1625 94References

- 1626 [ebMS] "Message Service Specification Version 2.0", OASIS ebXML Messaging Services  
1627 Technical Committee, OASIS Standard, 1 April 2002. Available at  
1628 <http://www.ebxml.org/specs/ebMS2.pdf>
- 1629 [RFC1738] "Uniform Resource Locators (URL)", T. Berners-Lee et al, RFC 1738, IESG and  
1630 IETF, December 1994. Available at  
1631 <http://www.ietf.org/rfc/rfc1738.txt>
- 1632 [RFC2119] "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119,  
1633 Bradner, S., IESG and IETF, March 1997. Available at  
1634 <http://www.ietf.org/rfc/rfc2119.txt>
- 1635 [RFC2392] "Content-ID and Message-ID Uniform Resource Locators", RFC2392, E. Levinson,  
1636 IESG and IETF, August 1998. Available at  
1637 <http://www.ietf.org/rfc/rfc2392.txt>
- 1638 [RFC2396] "Uniform Resource Identifiers (URI): Generic Syntax", RFC 2396, Tim Berners-Lee et  
1639 al, IESG and IETF, August 1998. Available at  
1640 <http://www.ietf.org/rfc/rfc2396.txt>
- 1641 [RFC2616] "Hypertext Transfer Protocol – HTTP/1.1", RFC 2616, R. Fielding et al, IESG and  
1642 IETF, June 1999. Available at  
1643 <http://www.ietf.org/rfc/rfc2616.txt>
- 1644 [RFC2822] "Internet Message Format", RFC 2822, P. Resnick, Editor, IESG and IETF, April  
1645 2001. Available at  
1646 <http://www.ietf.org/rfc/rfc2822.txt>
- 1647 [SOAP 1.1] "Simple Object Access Protocol (SOAP) 1.1", Don Box et al, W3C Note, 8 May,  
1648 2000. Available at  
1649 <http://www.w3.org/TR/2000/NOTE-SOAP-20000508/>
- 1650 [SOAP 1.2 Part 1] "SOAP 1.2 Part 1: Messaging Framework", Martin Gudgin, Marc Hadley, Noah  
1651 Mendelsohn, Jean-Jacques Moreau, Henrik Frystyk Nielsen, eds., W3C Recommendation, 24  
1652 June 2003. Available at  
1653 <http://www.w3.org/TR/2003/REC-soap12-part1-20030624/>

- 1654 [SOAP 1.2 Part 2] "SOAP 1.2 Part 1: Adjuncts", Martin Gudgin, Marc Hadley, Noah Mendelsohn,  
1655 Jean-Jacques Moreau, Henrik Frystyk Nielsen, eds., W3C Recommendation, 24 June 2003.  
1656 Available at
- 1657 <http://www.w3.org/TR/2003/REC-soap12-part2-20030624/>
- 1658 [SOAP with Attachments] "SOAP Messages with Attachments", John J. Barton, Satish Thatte,  
1659 Henrik Frystyk Nielsen, W3C Note, 11 December 2000, Available at
- 1660 <http://www.w3.org/TR/SOAP-attachments>
- 1661 [XML] "Extensible Markup Language (XML) 1.0 (Third Edition)", Tim Bray et al, eds., W3C  
1662 Recommendation, first published 10 February 1998, revised 4 February 2004. Available at
- 1663 <http://www.w3.org/TR/2004/REC-xml-20040204>
- 1664 [XML Namespaces] "Namespaces in XML", Tim Bray et al., eds., W3C Recommendation, 14  
1665 January 1999. Available at
- 1666 <http://www.w3.org/TR/1999/REC-xml-names-19990114/>
- 1667 [XML Schema Part 1] "XML Schema Part 1: Structures", Henry S. Thompson, David Beech,  
1668 Murray Maloney, Noah Mendelsohn, eds., W3C Recommendation, 2 May 2001. Available at
- 1669 <http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/>
- 1670 [XML Schema Part 2] "XML Schema Part 2: Datatypes", Paul V. Biron and Ashok Malhotra, eds.  
1671 W3C Recommendation, 2 May 2001. Available at
- 1672 <http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/>
- 1673 [XPath 1.0] "XML Path Language (XPath) Version 1.0", James Clark, Steve DeRose, eds., W3C  
1674 Recommendation, 16 November 1999. Available at
- 1675 <http://www.w3.org/TR/1999/REC-xpath-19991116>
- 1676 [WSDL 1.1] "Web Services Description Language (WSDL) 1.1", Erik Christensen, Francisco  
1677 Curbera, Greg Meredith, Sanjiva Weerawarana, eds., W3C Note, 15 March 2001. Available at
- 1678 <http://www.w3.org/TR/2001/NOTE-wsdl-20010315>
- 1679 [WS-I BP 1.1] "Basic Profile Version 1.1", Keith Ballinger, David Ehnebuske, Christopher Ferris,  
1680 Martin Gudgin, Mark Nottingham, Prasad Yendluri, eds., WS-I specification, 8 August 2004.  
1681 Available at
- 1682 <http://www.ws-i.org/Profiles/BasicProfile-1.1-2004-07-21.html>
- 1683 [WSS] "OASIS Web Services Security: SOAP Message Security 1.0 (WS-Security 2004)", Chris  
1684 Kaler, Phillip Hallam-Baker, Ronald Monzillo, eds, OASIS Standard 200401, March 2004.  
1685 Available at
- 1686 <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-soap-message-security-1.0.pdf>

---

## 1687 Appendix A.Schema (Normative)

1688 The schemas for this specification have the following URLs and are located using the filenames  
1689 shown in the table:

Schema Namespace URL	File name	Prefix
<a href="http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd">http://docs.oasis-open.org/wsrn/2004/06/ws-reliability-1.1.xsd</a>	ws-reliability-1.1.xsd	wstrn
<a href="http://docs.oasis-open.org/wsrn/2004/06/reference-1.1.xsd">http://docs.oasis-open.org/wsrn/2004/06/reference-1.1.xsd</a>	reference-1.1.xsd	ref
<a href="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd">http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd</a>	fnp-1.1.xsd	fnp
<a href="http://docs.oasis-open.org/wsrn/2004/06/wstrnfp-1.1.xsd">http://docs.oasis-open.org/wsrn/2004/06/wstrnfp-1.1.xsd</a>	wstrnfp-1.1.xsd	wstrnfp

**Table 28 WS-Reliability Schema Prefixes**

1690 RMPs MUST include the SOAP mustUnderstand attribute (defined in the same namespace used  
1691 for the soap:Envelope element) in all Reliable Messaging specified header blocks and MUST  
1692 observe the following restrictions:

- 1693
- For SOAP 1.1, the mustUnderstand attribute value is restricted to "1".
- 1694
- For SOAP 1.2, the mustUnderstand attribute value is restricted to "1" or "true".



---

1695 **Appendix B.WS-Reliability Features, Properties**  
1696 **and Compositors (Normative and Optional)**

1697 **B.1. Introduction**

1698 Users of a Web Service need to be aware of the reliability capabilities (RM capabilities) the  
1699 service supports or requires. One practical location to advertise these capabilities is in the  
1700 service description (WSDL document), which allows publishing both abstract service definitions  
1701 and concrete protocol details (bindings). This allows clients (including other Web services) to  
1702 easily obtain information about specific capabilities (such as guaranteed delivery, duplicate  
1703 elimination, message ordering, and the supported reply patterns) of a specific Web service  
1704 before calling the service. While bundling RM capabilities with the service description may not  
1705 be desirable in all cases, this convenient approach often should be appropriate. The WSDL  
1706 annotation mechanism described here adds such capability assertions in a flexible way.

1707 WS-Reliability uses the WSDL 1.1 extensibility points to define an extensible framework  
1708 consisting of features, properties and compositors. This framework addresses the needs of a  
1709 reliable Web service to advertise its capabilities and the composability of those capabilities.

1710 The following extensibility elements are relevant to RM capabilities:

- 1711 • **feature** – see [Appendix B.3.2](#).
- 1712 • **property** – see [Appendix B.3.3](#).
- 1713 • **compositor** – see [Appendix B.3.1](#).

1714 An annotation composed with the above extensibility elements will specify the reliability features  
1715 and properties associated with specific WSDL constructs. Features and properties represent RM  
1716 capabilities; compositors specify how these capabilities are composed.

1717 This would, for example, allow a Web service description to advertise that clients invoking the  
1718 service must use duplicate elimination or message ordering.

1719 **B.2. Conformance**

1720 Implementations of WS-Reliability are expected (though not required) to understand the WSDL  
1721 extensibility points defined in this section.

1722 Understanding these extensibility points promotes interoperability: a service advertises its  
1723 supported and required features when its WSDL document contains these extensibility points.  
1724 Therefore it is RECOMMENDED that implementations recognize, understand and support these  
1725 extensibility points.

1726 It is also possible for services to advertise features through other channels (such as UDDI) in  
1727 addition to these extensibility points.

## 1728 B.3. WSDL Extensibility Elements

### 1729 B.3.1. Composer

1730 The compositor semantics describe how features and properties are composed for the enclosing  
1731 component (or WSDL 1.1 element). The compositor's semantics determine whether the usage of  
1732 composed elements by a client to the service is required or optional. All of the RM capabilities  
1733 represented by these elements must be supported by the service. A compositor element can  
1734 occur as a child element of wsdl11:portType, wsdl11:operation (which itself may be a child of  
1735 wsdl11:portType or wsdl11:binding), wsdl11:binding, wsdl11:service and wsdl11:port. The  
1736 compositor element uses the extensibility defined by WSDL 1.1. A compositor element specifies  
1737 the semantics for combining its children elements. These children elements can be additional  
1738 compositors, features, properties or extensibility elements.

1739 A compositor element is expressed by the following pseudo-syntax:

```
1740 <fnp:compositor uri="..." name="NCName"?>  
1741 [fnp:feature/> | <fnp:property/> | <fnp:compositor/> |  
1742   <extensibility-element/>]+  
1743 </fnp:compositor>
```

1744 The uri attribute of the compositor specifies its semantics. Four different compositors (URIs) and  
1745 their capability-related semantics are described below. It is possible to provide additional  
1746 compositors by using other URIs. The possibility of additional compositors and the existence of  
1747 extensibility points (represented by "<extensibility-element>") make the framework extensible.  
1748 The optional @name identifies the compositor. An element built with such compositors  
1749 represents an RM capability.

- 1750 • **all**: this compositor specifies that a service invocation MUST comply with all of the  
1751 children elements representing RM capability assertions. This compositor is identified  
1752 by the URI:  
1753 <http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositors/all>
- 1754 • **choice**: this compositor specifies that a service invocation MUST comply with exactly  
1755 one of the possibly many children elements representing RM capability assertions. This  
1756 compositor is identified by the URI:  
1757 <http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositors/choice>
- 1758 • **one-or-more**: this compositor specifies that a service invocation MUST comply with at  
1759 least one of the possibly many children elements representing RM capability assertions.  
1760 This compositor is identified by the URI:  
1761 <http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositors/one-or-more>
- 1762 • **zero-or-more**: this compositor specifies that a service invocation MAY comply with one  
1763 or more of the children elements representing RM capability assertions. This compositor  
1764 is identified by the URI:  
1765 <http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositors/zero-or-more>

1766 Examples for each compositor are provided in **Appendix B.7** below.

1767 Compositors specified at different WSDL components are implicitly aggregated using the 'all'  
1768 compositor at the dependent WSDL component. Consider the example below:

```

1769 <wsdl11:definitions>
1770   ...
1771   <wsdl11:portType name="myPortType">
1772     <fnp:compositor uri="..." name="A">
1773       ...
1774     </fnp:compositor>
1775     ...
1776   </wsdl11:portType>
1777   <wsdl11:binding name="myBinding" type="myPortType">
1778     <fnp:compositor uri="..." name="B">
1779       ...
1780     </fnp:compositor>
1781     ...
1782   <wsdl11:binding>
1783     <wsdl11:service name="myService">
1784       <wsdl11:port name="myPort" binding="myBinding">
1785         ...
1786       </wsdl11:port>
1787     </wsdl11:service>
1788   </wsdl11:definitions>

```

1789 The compositor specified at the wsdl11:portType "myPortType" and the compositor specified at  
1790 wsdl11:binding "myBinding" are aggregated at the dependent wsdl11:port "myPort" using the 'all'  
1791 compositor. The equivalent compositor at "myPort" is

```

1792 <fnp:compositor
1793   uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1794   <fnp:compositor uri="..." name="A">
1795   </fnp:compositor>
1796   <fnp:compositor uri="..." name="B">
1797     ...
1798   </fnp:compositor>
1799 </fnp:compositor>

```

### 1800 **B.3.2. Feature**

1801 A feature describes an abstract RM capability or assertion associated with a WSDL element. A  
1802 feature can occur only as a child of a compositor.

1803 The enclosing compositor(s) define(s) whether or not the usage of a feature is required. A  
1804 feature is identified by a URI. Recognizing the URI of a feature implies understanding the feature  
1805 identified by that URI.

1806 A feature element is expressed by the following pseudo-syntax:

```

1807 <fnp:feature uri="...">
1808   [<fnp:compositor/> | <extensibility-element/>]*
1809 </fnp:feature>

```

### 1810 **B.3.3. Property**

1811 A property is identified by a QName. A property is an assertion or constraint on a specific RM  
1812 capability and its value(s). A property can occur only as a child of a compositor.

1813 Typically, properties are (but are not required to be) associated with a feature and are described  
1814 in a feature specification. The QName identifier of a property uniquely identifies the property.  
1815 Recognizing the property QName identifier implies understanding the semantics associated with  
1816 that property. The property QName identifier typically points to a global XML Schema element  
1817 declaration. A property specification typically specifies the schema containing this global element  
1818 declaration. There may be a constraint on the set of values a property can have; such a  
1819 constraint is specified by a QName identifying an XML Schema type.

```
1820 <fnp:property name="xs:QName">  
1821   [<fnp:value>xs:anyType</fnp:value> |  
1822     <fnp:constraint>xs:QName</fnp:constraint>]  
1823   [<extensibility-element/>]*  
1824 </fnp:property>
```

### 1825 **B.4. WS-Reliability Feature**

1826 The WS-Reliability feature is identified by the URI

1827 <http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd>

1828 This feature URI identifies the WS-Reliability specification. Understanding this URI implies  
1829 understanding the WS-Reliability specification.

### 1830 **B.5. WS-Reliability Properties**

1831 This section identifies properties for the WS-Reliability specification. Typically these properties  
1832 are scoped within the feature identified by the URI

1833 <http://docs.oasis-open.org/wsrn/2004/06/wsrmp-1.1.xsd>

#### 1834 **B.5.1. Guaranteed Delivery Property**

1835 This property is identified by the QName "wsrmp:GuaranteedDelivery" and corresponds to the  
1836 semantics specified by the WS-Reliability guaranteed delivery semantics. The type of this  
1837 property is "xs:boolean".

#### 1838 **B.5.2. Duplicate Elimination Property**

1839 This property is identified by the QName "wsrmp:NoDuplicateDelivery" and corresponds to the  
1840 semantics specified by the WS-Reliability duplicate elimination semantics. The type of this  
1841 property is "xs:boolean".

#### 1842 **B.5.3. Message Ordering Property**

1843 This property is identified by the QName "wsrmp:OrderedDelivery" and corresponds to the  
1844 semantics specified by the WS-Reliability message ordering semantics. The type of this property  
1845 is "xs:boolean".

## 1846 B.5.4. Reply Pattern Property

1847 This property is identified by the QName "wsmfp:ReplyPattern" and corresponds to the  
1848 semantics specified by the WS-Reliability reply pattern options. The type of this property is  
1849 "xs:string". (values: Response, Poll, Callback)

## 1850 B.6. Compositor Examples

### 1851 B.6.1. Example for the "all" compositor

```
1852 <wsdl11:portType name="Example-1">  
1853   <fnp:compositor  
1854     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">  
1855     <fnp:feature  
1856       uri="http://docs.oasis-open.org/wsrn/2004/06/wsmfp-1.1.xsd"  
1857       <fnp:compositor uri=  
1858         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">  
1859         <fnp:property name="wsmfp:NoDuplicateDelivery">  
1860           <fnp:value>true</fnp:value>  
1861         </fnp:property>  
1862         <fnp:property name="wsmfp:OrderedDelivery">  
1863           <fnp:value>true</fnp:value>  
1864         </fnp:property>  
1865         <fnp:property name="wsmfp:GuaranteedDelivery">  
1866           <fnp:value>true</fnp:value>  
1867         </fnp:property>  
1868       </fnp:compositor>  
1869     </fnp:feature>  
1870   </fnp:compositor>  
1871   ...  
1872 </wsdl11:portType>
```

1873 In the example above, the reliability feature identified by URI "[http://docs.oasis-](http://docs.oasis-open.org/wsrn/2004/06/wsmfp-1.1.xsd)  
1874 [open.org/wsrn/2004/06/wsmfp-1.1.xsd](http://docs.oasis-open.org/wsrn/2004/06/wsmfp-1.1.xsd)" is required by the portType. This feature consists of  
1875 three properties, all of which are required because of the semantics of the 'all' compositor that  
1876 composes the three properties.

1877 **B.6.2. Example for the "choice" compositor:**

```
1878 <wsdl11:binding name="Example-2">
1879   <fnp:compositor
1880     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1881     <fnp:feature
1882       uri="http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd"
1883       <fnp:compositor uri=
1884         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositors/choice">
1885         <fnp:property name="wsrnf:ReplyPattern">
1886           <value>Response</value>
1887         </fnp:property>
1888         <fnp:property name="wsrnf:ReplyPattern">
1889           <value>Callback</value>
1890         </fnp:property>
1891         <fnp:property name="wsrnf:ReplyPattern">
1892           <value>Poll</value>
1893         </fnp:property>
1894       </fnp:compositor>
1895     </fnp:feature>
1896   </fnp:compositor>
1897   ...
1898 </wsdl11:binding>
```

1899 In the example above, the reliability feature identified by URI "[http://docs.oasis-](http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd)  
1900 [open.org/wsrn/2004/06/wsrnfp-1.1.xsd](http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd)" is required by the portType. This feature consists of  
1901 three properties composed by the 'choice' compositor; the client must choose one.

### 1902 **B.6.3.Example for the "one-or-more" compositor:**

```
1903 <wsdl11:portType name="Example-3">
1904   <fnp:compositor
1905     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1906     <fnp:feature
1907       uri="http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd" >
1908       <fnp:compositor uri=
1909         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/one-or-more">
1910         <fnp:property name="wsrnfp:NoDuplicateDelivery">
1911           <fnp:value>true</fnp:value>
1912         </fnp:property>
1913         <fnp:property name="wsrnfp:OrderedDelivery">
1914           <fnp:value>true</fnp:value>
1915         </fnp:property>
1916         <fnp:property name="wsrnfp:GuaranteedDelivery">
1917           <fnp:value>true</fnp:value>
1918         </fnp:property>
1919       </fnp:compositor>
1920     </fnp:feature>
1921   </fnp:compositor>
1922   ...
1923 </wsdl11:portType>
```

### 1924 **B.6.4.Example for the "zero-or-more" compositor:**

```
1925 <wsdl11:portType name="Example-4">
1926   <fnp:compositor
1927     uri="http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/all">
1928     <fnp:feature
1929       uri="http://docs.oasis-open.org/wsrn/2004/06/wsrnfp-1.1.xsd"
1930       <fnp:compositor uri=
1931         "http://docs.oasis-open.org/wsrn/2004/06/fnp-1.1.xsd/compositor/zero-or-more">
1932         <fnp:property name="wsrnfp:NoDuplicateDelivery">
1933           <fnp:value>true</fnp:value>
1934         </fnp:property>
1935         <fnp:property name="wsrnfp:OrderedDelivery">
1936           <fnp:value>true</fnp:value>
1937         </fnp:property>
1938         <fnp:property name="wsrnfp:GuaranteedDelivery">
1939           <fnp:value>true</fnp:value>
1940         </fnp:property>
1941       </fnp:compositor>
1942     </fnp:feature>
1943   </fnp:compositor>
1944   ...
1945 </wsdl11:portType>
```

---

## 1946 **Appendix C.Acknowledgments**

1947 The following individuals were members of the committee during the development of this  
1948 specification:

- 1949 □ David Ingham, Arjuna Technologies Limited
- 1950 □ Joseph Chiusano, Booz Allen Hamilton
- 1951 □ Peter Furniss, Choreology Ltd
- 1952 □ Jeff Turpin, Cyclone Commerce
- 1953 □ Pramila Mullan, France Telecom
- 1954 □ Jacques Durand, Fujitsu
- 1955 □ Kazunori Iwasa (Secretary), Fujitsu
- 1956 □ Tom Rutt (Chair), Fujitsu
- 1957 □ Jishnu Mukerji, Hewlett-Packard
- 1958 □ Robert Freund, Hitachi
- 1959 □ Eisaku Nishiyama, Hitachi
- 1960 □ Nobuyuki Yamamoto, Hitachi
- 1961 □ Ben Bloch, Individual
- 1962 □ Mark Hansen, Individual
- 1963 □ Paolo Romano, Individual
- 1964 □ Dock Allen, Mitre Corporation
- 1965 □ Junichi Tatemura, NEC Corporation
- 1966 □ Alan Weissberger, NEC Corporation
- 1967 □ Magdolna Gerendai, Nokia
- 1968 □ Szabolcs Payrits, Nokia
- 1969 □ Mark Peel, Novell
- 1970 □ Sunil Kunisetty (Secretary), Oracle
- 1971 □ Anish Karmarkar, Oracle
- 1972 □ Jeff Mischkinsky, Oracle
- 1973 □ Marc Goodner (Secretary), SAP
- 1974 □ Pete Wenzel, SeeBeyond Technology Corporation
- 1975 □ Doug Bunting (Secretary), Sun Microsystems



- 1976 □ Tony Graham, Sun Microsystems
- 1977 □ Chi-Yuen Ng, University of Hong Kong
- 1978 □ Patrick Yee, University of Hong Kong
- 1979 □ Prasad Yendluri, webMethods, Inc.
- 1980 □ Scott Werden, WRQ, Inc.

1981 And the following people made contributions to produce Ver 1.0 of this specification:

- 1982 Colleen Evans, Sonic Software Corporation / Dave Chappell, Sonic Software Corporation / Doug
- 1983 Bunting, Sun Microsystems, Inc. / George Tharakan, Sun Microsystems, Inc. / Hisashi
- 1984 Shimamura, NEC Corporation / Jacques Durand, Fujitsu Software Corporation / Jeff Mischkinsky,
- 1985 Oracle Corporation / Katsutoshi Nihei, NEC Corporation / Kazunori Iwasa, Fujitsu Limited /
- 1986 Martin Chapman, Oracle Corporation / Masayoshi Shimamura, Fujitsu Limited / Nicholas
- 1987 Kassem, Sun Microsystems, Inc. / Nobuyuki Yamamoto, Hitachi Limited / Sunil Kunisetty, Oracle
- 1988 Corporation / Tetsuya Hashimoto, Hitachi Limited / Tom Rutt, Fujitsu Software Corporation /
- 1989 Yoshihide Nomura, Fujitsu Limited / Akira Ochi, Fujitsu Limited / Hirotaka Hara, Fujitsu Limited /
- 1990 Hiroyuki Tomisawa, Hitachi Limited / Katsuhisa Nakazato, Fujitsu Limited / Masahiko Narita,
- 1991 Fujitsu Limited / Nobuyuki Saji, NEC Corporation / Shuichi Imabayashi, Fujitsu Limited

1992

---

## Appendix D. Notices

1993 OASIS takes no position regarding the validity or scope of any intellectual property or other  
1994 rights that might be claimed to pertain to the implementation or use of the technology described  
1995 in this document or the extent to which any license under such rights might or might not be  
1996 available; neither does it represent that it has made any effort to identify any such rights.  
1997 Information on OASIS's procedures with respect to rights in OASIS specifications can be found  
1998 at the OASIS website. Copies of claims of rights made available for publication and any  
1999 assurances of licenses to be made available, or the result of an attempt made to obtain a  
2000 general license or permission for the use of such proprietary rights by implementors or users of  
2001 this specification, can be obtained from the OASIS Executive Director.

2002 OASIS invites any interested party to bring to its attention any copyrights, patents or patent  
2003 applications, or other proprietary rights which may cover technology that may be required to  
2004 implement this specification. Please address the information to the OASIS Executive Director.

2005 **Copyright © OASIS Open 2003-2004. All Rights Reserved.**

2006 This document and translations of it may be copied and furnished to others, and derivative works  
2007 that comment on or otherwise explain it or assist in its implementation may be prepared, copied,  
2008 published and distributed, in whole or in part, without restriction of any kind, provided that the  
2009 above copyright notice and this paragraph are included on all such copies and derivative works.  
2010 However, this document itself does not be modified in any way, such as by removing the  
2011 copyright notice or references to OASIS, except as needed for the purpose of developing OASIS  
2012 specifications, in which case the procedures for copyrights defined in the OASIS Intellectual  
2013 Property Rights document must be followed, or as required to translate it into languages other  
2014 than English.

2015 The limited permissions granted above are perpetual and will not be revoked by OASIS or its  
2016 successors or assigns.

2017 This document and the information contained herein is provided on an "AS IS" basis and OASIS  
2018 DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO  
2019 ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE  
2020 ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A  
2021 PARTICULAR PURPOSE.