Web Services Reliable Messaging TC

WS-Reliability

Committee Draft 0.52, 25 November 2003

Document identifier:
-web services reliable messaging tc-ws-reliability-0.52

Location:
http://www.oasis-open.org/(TBD)/(TBD)/

Editor:
Kazunori Iwasa, Fujitsu Limited <kiwasa@jp.fujitsu.com>

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

Status:
This document is updated periodically on no particular schedule. Send comments to the editor.

Committee members should send comments on this specification to the wsrn@lists.oasis-open.org list. Others should subscribe to and send comments to the wsrn-comment@lists.oasis-open.org list. To subscribe, send an email message to wsrn-comment-request@lists.oasis-open.org with the word "subscribe" as the body of the message.

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

The errata page for this specification is at http://www.oasis-open.org/committees/wsrn/xxx. (TBD)
# Table of Contents

1 Introduction (Need Updating) ................................................................. 3
  1.1 Purpose of WS-Reliability .............................................................. 3
  1.2 Scope and Definition of Reliable Messaging ................................. 3
  1.3 Out of the scope of this specification are: ....................................... 3
  1.4 The Goal of this specification ....................................................... 4
  1.5 Notational Conventions ............................................................... 4
  1.6 Relation to Other Specifications ................................................... 4
  1.7 Examples of Messages compliant with WS-Reliability .................. 5
  1.8 Terminology ............................................................................... 5

2 Messaging Model .............................................................................. 6
  2.1 Overview of Messaging Model (Need to include additional messaging model) ................................................. 6
  2.2 Overview of the Acknowledgment Message ................................. 6
  2.3 Message Identifier ....................................................................... 7
  2.4 Retry .......................................................................................... 7
  2.5 Message Persistence .................................................................... 7
  2.6 Duplicate Elimination ................................................................. 7
  2.7 Guaranteeing Message Order ....................................................... 7
  2.8 Sequence Number (Need updating) ............................................... 8

3 Message Format ................................................................................ 9
  3.1 MessageHeader Element .............................................................. 10
  3.2 Request Element ....................................................................... 13
  3.3 Response Element ..................................................................... 14

4 SOAP Fault (Need updating) ............................................................... 15
  4.1 SOAP Fault extension for Reliable Messaging ............................. 15
  4.2 Fault code description ............................................................... 17

5 HTTP Binding (Needs to be updated) .............................................. 19
  5.1 Reliable Messaging with Synchronous Acknowledgment or Fault Message ......................................................... 19
  5.2 Reliable Messaging with Asynchronous Acknowledgment Message ................................................................. 20

6 References ...................................................................................... 22
  6.1 Normative ................................................................................. 22
  6.2 Non-normative References ......................................................... 23
1 Introduction (Need Updating)

1.1 Purpose of WS-Reliability

The purpose of WS-Reliability is to address reliable messaging requirements, which become critical, for example, when using Web Services in B2B applications. SOAP [SOAP1.1] over HTTP [RFC2616] is not sufficient when an application-level messaging protocol must also address reliability and security. While security is getting traction in the development of Web Services standards, reliability is not. This specification is intended as an initial proposal for defining reliability in the context of current Web Services standards. The specification borrows from previous work in messaging and transport protocols, e.g., SOAP, and the ebXML Message Service [ebMS]. It proposes appropriate modifications to apply this work to Web Services.

1.2 Scope and Definition of Reliable Messaging

The focus of this specification is on the SOAP layer and envelope. The authors do not presume to cover all aspects of Reliable Messaging. Several fundamental questions on reliability need to be addressed in subsequent work, and are not addressed in this specification:

- Assuming that reliability objectives cannot always be guaranteed or attainable, should a reliability contract include advanced quality of service elements (which may translate into specifying quantitative thresholds, e.g. how large a message archive or time period a duplicate check should cover)?

- Beyond the specified qualities of message delivery (guaranteed delivery, duplicate elimination, and message ordering), should reliability also define the degree of synchronization between sender and receiver applications (i.e. the degree to which both sender and receiver parties will have same understanding of whether a request was properly received or not)?

Within the scope of this specification, the following features are investigated:

- Asynchronous messaging at the application level

  Three reliability features: Guaranteed delivery, Duplicate Elimination, and Message Ordering.

In the current specification, we will define reliable messaging as the mechanism supporting the following requirements at the application level:

Guaranteed message delivery, or At-Least-Once semantics

- Guaranteed message duplicate elimination, or At-Most-Once semantics

- Guaranteed message delivery and duplicate elimination, or Exactly-Once semantics

- Guaranteed message ordering, within a context delimited using a group id.

1.3 Out of the scope of this specification are:

- Application level synchronous messaging. Applications which intentionally use synchronous messaging at the application level, require knowledge of the error status
immediately, rather than waiting for the messaging layer to resend the message when an error occurs at the receiver side, and are out of scope of this specification.

- Routing. Other techniques can be used in conjunction with an implementation of this specification.
- Security. Other mechanisms can be used in conjunction with an implementation of this specification.

1.4 The Goal of this specification

The goal of this specification is to define:

- A mechanism to guarantee message delivery and its expression in SOAP messages.
- A mechanism to eliminate duplicate messages and its expression in SOAP messages.
- A mechanism to guarantee received message order (within a context) and its expression in SOAP messages.

1.5 Notational Conventions

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

1.6 Relation to Other Specifications

(1) W3C SOAP1.1/1.2:
SOAP1.1 [SOAP1.1] is currently the base protocol for this specification. This specification defines extensions to SOAP Header and Body elements. This specification could be updated and defined as a Reliable Messaging Feature to be compliant with SOAP 1.2 [SOAP1.2] when it becomes a W3C Recommendation.

(2) OASIS ebXML Message Service Specification 2.0:
The reliable message mechanism defined in the ebXML Message Service Specification 2.0 [ebMS] is implemented in a number of products and open source efforts, many of which have undergone interoperability testing. WS-Reliability borrows from this technology.

(3) OASIS WS-Security:
This specification can be used with WS-Security [WSS] when that effort is completed in OASIS.

(4) WS-I Basic Profile 1.0

(TBD)
1.7 Examples of Messages compliant with WS-Reliability

(To be added later.)

1.8 Terminology

NOTE: The following terminology has not yet been aligned with the terminology used across other
Web Services specifications. It is included here for purposes of establishing an understanding of
how these terms are used in this specification.

Reliable Messaging:
The set of mechanisms and procedures required to send messages reliably. This includes the
processing of Acknowledgment messages, re-sending of messages, duplicate message
elimination, and message ordering.

Reliable Messaging Processor (RMP):
A module capable of processing and enforcing Reliable Messaging as described in this
specification.

Reliable Message:
A message for which the sender requires some level of reliable delivery, typically requiring
acknowledgment for notification of delivery.

Acknowledgment Message:
A signal message sent by a SOAP node, to notify the initial sender of delivery of the message.
The Acknowledgment message indicates the receiver has received the message and the sender
is no longer responsible to persist the message. (i.e. the receiver now has responsibility for
message persistence)

Fault Message:
A message to notify the sender of the message that there was a failure to receive or process the
message.

Normal Message:
A message which is not an Acknowledgment message, and which is not a Fault message.
2 Messaging Model

The following section provides an overview of the WS-Reliability Messaging Model.

2.1 Overview of Messaging Model (Need to include additional messaging model)

In the Reliable Messaging Model described in this document, the sender node sends a message to the receiver node directly (i.e., intermediaries are assumed to be transparent in this specification). The receiver node sends back an Acknowledgment message to the sender node. Figure 1 shows this model.

![Figure 1 Messaging Model](image)

2.2 Overview of the Acknowledgment Message

When supporting reliable messaging, upon receipt of a reliable message, the server MUST send a reply. This reply MUST be either an Acknowledge message or a Fault message. A SOAP Reliable Message is used as described in Figure 2 to guarantee message delivery. The Acknowledgment is correlated with a normal message by reference to its message ID.

![Figure 2 Guaranteeing Message delivery](image)

When the Sender sends a SOAP Reliable Message to the Receiver, the Receiver MUST send back an Acknowledgment message or Fault message to the Sender.
2.3 Message Identifier

Message Identifier is a combination of GroupId element and SequenceNumber element. Every Reliable Message MUST contain a globally unique Message Identifier. The Acknowledgment message MUST contain a reference to the Message Identifier of the Acknowledged message, confirming that the receiver SOAP node has received the message. (Need update for clarify semantics of use of SequenceNumber when ordering was not used.)

2.4 Retry

If the SOAP node sending a Reliable Message does not receive an Acknowledgment message, that sender MUST resend the same message with same Message Identifier to the receiver node until (1) the sender gets an Acknowledgment message from the receiver, or (2) a specified number of resend attempts have been made without success. If the sender SOAP node fails to send the message (i.e., no Acknowledgment is received), the node MUST report the error to the application layer in some way.

2.5 Message Persistence

With Reliable Messaging, the sender is REQUIRED to persist the message until one of the following conditions are met:

- Receipt of an Acknowledgment message from receiver, indicating the message has been successfully delivered.
- All retry attempts have failed, and a delivery failure is reported to the application layer.
- The span of time indicated by the ExpiryTime element has expired.

The receiver MUST persist out of order messages to support ordered delivery. The receiver MUST persist the Message Identifier to support duplicate elimination. Both sender and receiver MUST behave as if there was no system failure or system down after recovery.

2.6 Duplicate Elimination

A number of conditions may result in transmission of duplicate message(s), e.g., temporary downtime of the sender or receiver, a routing problem between the sender and receiver, etc. In order to provide at-most-once semantics, the ultimate receiver MUST eliminate duplicate messages. Messages with the same Message Identifier value MUST be treated as duplicates.

2.7 Guaranteeing Message Order

Some applications will expect to receive a sequence of messages from the same sender in the same order these messages were sent. Although there are often means to enforce this at the application layer, this is not always possible or practical. In such cases, the messaging layer is required to guarantee the Message Order. This specification defines a model described in Figure 3 to meet this requirement.

Figure 3 Ordering Model
2.8 Sequence Number (Need updating)

A sequence number mechanism is used to track and enforce the order of a sequence of messages having a common grouping identifier value. Such a mechanism has been widely used in the past. For example, assume the sender application layer generates three messages in order of (1), (2), and (3). The sender SOAP node, with the message ordering function enabled, sends those messages in order of (1), (2), and (3), sequentially and asynchronously, with respective sequence numbers 1, 2, and 3. If the message (2) was not properly received for any reason, the sender will resend the (2) message after a timeout has occurred. The receiver’s SOAP node will finally receive these messages as a sequence: (1), (3), and (2). The receiver SOAP node, with the message ordering function enabled, may provide the application layer with message (1), but not (3). Sequence numbering allows the receiver node to easily detect a missing message in a sequence, that is (2), as soon as receiving (3). This condition is recognized by the receiver when the sequence numbers of the messages it receives are not contiguous (e.g., 1, 3, 2). The receiver SOAP node will wait for a message with sequence number 2, and then provide message (2) and then message (3) to the application layer, in order. This behavior can be subject to variants and additional rules to deal with specific failure use cases, such as when a node cannot deliver the proper-sequence of messages due to a message being lost.
3 Message Format

Figure 4 shows the structure of WS-Reliability embedded in the SOAP Envelope.

Figure 4  Structure of WS-Reliability

SOAP:Envelope

SOAP:Header

RM:MessageHeader

RM:GroupId
RM:SequenceNumber
RM:Timestamp
RM:ExpiryTime
RM:ReplyPattern
RM:ReplyTo
any

RM:Request

RM:AckRequested
RM:DuplicateElimination
any

RM:Response

RM:RefToGroupId
RM:RefToSequenceNo
any

RM:Fault

RM:FaultCode
any

any

SOAP:Body
The namespaces [XML namespaces] for reliable messaging defined in this specification are:

http://www.oasis-open.org/committees/wsrp/schema/1.1/SOAP1.1 for SOAP1.1 and
http://www.oasis-open.org/committees/wsrp/schema/1.1/SOAP1.2 for SOAP1.2

If there are additional elements that are not described in this specification present in a message, the Reliable Messaging Processor MUST ignore those elements.

In a reliable message, the following four elements are direct children of SOAP Header:

- **MessageHeader** element
- **Request** element
- **Response** element
- **Fault** element

NOTE: This Reliability specification defines elements and attributes that may also be required by functions other than reliable messaging (e.g. routing, security, choreography, etc.). When using a messaging mode combining several of these functions, such parameters SHOULD NOT be duplicated across multiple SOAP headers.

This specification groups these elements and attributes into a header block, called a "MessageHeader", and under the namespace "RM". This is for the sake of completeness of this draft specification as a stand alone module. Work for future versions should consider reuse of corresponding headers and header elements, across multiple web service specifications.

This could be achieved in a number of ways. This specification could be modularized to make these elements and attributes available for any applicable use case. Alternatively, external specifications could be defined and leveraged within future versions of this specification. In either case, in order to standardize these parameters in a reusable way future versions of this specification should be coordinated with the work of other parties involved in specifying SOAP extensions relying on these parameters.

### 3.1 MessageHeader Element

The MessageHeader element includes basic information to be used for a reliable message. This element includes the following attributes and sub-elements:

- a SOAP **mustUnderstand** attribute with a value of “1”
- **GroupId** element
- **SequenceNumber** element
- **Timestamp** element
- **ExpiryTime** element
- **ReplyPattern** element

Example 4 shows an example of a MessageHeader element.
3.1.1 GroupId Element

This REQUIRED element is used to specify a group of messages. This element MUST have a globally unique identifier as its value. The syntax of this identification is URI, as defined in [RFC2396]. It is RECOMMENDED to use a syntax of Message-ID, as defined in [RFC2392]. This element contains the following attributes:

- a removeAfter attribute

(1) removeAfter attribute

This is an OPTIONAL attribute. This attribute is used to specify the time the GroupId can be removed from the RMP tracking mechanism for GroupId and SequenceNumber elements. Both sender and receiver MUST maintain the value of GroupId element until either one of the following two events occur:

- The sender sends a Message with the value of “End” in the status attribute.
- The time specified in the removeAfter attribute has passed.

The format MUST be expressed as UTC and MUST conform to a [XML Schema] dateTime. If omitted, the value SHOULD be considered as ‘forever’.

3.1.2 SequenceNumber Element

The SequenceNumber element is a REQUIRED element for Guaranteeing Ordered Delivery. When the message is not Guaranteed Order Delivery, this element MUST NOT present in the message. The value of this element MUST be unique within a same GroupId, and the combination of GroupId and SequenceNumber MUST be globaly unique to be used for Message Identifier. When a message includes MessageOrder element, this element is used for guaranteeing the message order within the group of messages categorized by the same GroupId value. In other words, the sequence of numbered messages that the receiver node presents to the application MUST be in the same order as the sequence of numbered messages that the sender application has produced, within the group of messages having the same GroupId value.

When the sender requests Guaranteed message ordering, the sender MUST use Guaranteed message delivery and duplicate elimination for that message as well. In particular, the sender MUST include both AckRequested element and DuplicateElimination element, as well as the MessageOrder element for Guaranteed message ordering. This element includes the following attribute:

- a status attribute

If the SequenceNumber element appears in the message sent, the receiver of the message is REQUIRED to make this message available to the application layer only after all messages with lower sequence number with the same GroupId have been made available to the application. In other words, an implementation of the receiver node MUST enforce the order in which messages are made available to the application, according to the sequence number order for messages with the same GroupId value. Example 5 illustrates this:

Example 5 Example of SequenceNumber element

(Example will be added later, when the schema was decided)
When a sender node communicates with a receiver node across several GroupId values, the
sender MUST maintain a distinct counter of the value of SequenceNumber for each GroupId
independently. When sending a message containing a MessageOrder element with a new
GroupId, the sender is REQUIRED to generate a new value for the SequenceNumber element in
the GroupId.

The value of SequenceNumber MUST conform to [XMLSchema] unsignedLong. The
SequenceNumber value MUST start with a value of 0 for the initial message to be sent to the
receiver with a specific GroupId. After the initial message has been sent to the receiver, the
sender MUST increment the value by one, for each message sent. When the value of a
SequenceNumber reaches the maximum value, the sender MUST generate a new GroupId for
any following messages. This begins a new sequence that could overlap with the old in rare
circumstances. From the receiver's perspective, no link exists between the two sequences. To
improve the chances that the message ordering is maintained across this change, the sender
SHOULD wait until all Acknowledgment messages have been received for the old GroupId before
starting the new sequence.

(1) status attribute

This OPTIONAL attribute is used to specify status of the group of messages. When this attribute
is present, its value MUST be one of the following three:

- **Start**: Indicating the message is the first message for a series of messages.
- **Continue**: Indicating the message is in the middle of a series of messages.
- **End**: Indicating the message is the last message for a series of messages.

The sender node MUST send a very first message, to guarantee the order, with “Start” for this
attribute. Also, the sender MUST send subsequent messages for the same series of messages
with “Continue”, until the message sent is the last one for the series of messages, for which case
the value MUST be “End”. When omitted, the default value for this attribute is “Continue.”

**NOTE**: Because delivery between the reliable messaging provider and the sequence is not
specified, this is not a complete guarantee of ordering to the application.

3.1.3 Timestamp Element

The Timestamp element is a REQUIRED element. This element has a date value set to the time
at which the message header was generated. The format of this value MUST conform to a [XML
Schema] dateTime and MUST be expressed as UTC.

3.1.4 ExpiryTime Element

This element is used to define the expiration time of the message. This is an OPTIONAL element.
If the ExpiryTime element is present, it MUST be used to indicate the time window that a
message should be made available by a receiver node to its application layer. The time MUST be
expressed as UTC and MUST conform to a [XML Schema] dateTime. The message is
considered expired if the current time, in UTC, is greater than the value of the ExpiryTime
element. If a receiver receives an expired message, it MUST send the sender a Fault message
with Error code of “InvalidExpiryTime”.

---

25 November 2003
Copyright © OASIS Open 2002. All Rights Reserved.
Page 12 of 27
3.1.5 ReplyPattern element

The ReplyPattern element is used for a sender to indicate what reply pattern is requested. The ReplyPattern element is a MANDATORY element. This element is used to specify whether the Acknowledgment Message (or Fault messages) should be sent back directly in the reply to the reliable message, in a separate callback request, or in the response to a separate poll request. This element MUST have one of the following three values:

- **Response**: An Acknowledgment Message or Fault message MUST be sent back directly in the Reply to the Reliable Message. This pattern is not applicable for one-way application level MEP.

- **Callback**: An Acknowledgment Message (or Fault message) MUST be sent as a callback request, using the address in the ReplyTo element.

- **Poll**: An Acknowledgment Message (or Fault message) MUST be sent as a response to a poll request.

The ReplyPattern element contains the following OPTIONAL attribute:

- a ReplyTo attribute

### (1) ReplyTo attribute

This is an OPTIONAL attribute, used to specify the initial sender’s endpoint to receive a callback Acknowledgment message or Fault Message. A value of this attribute MUST be present if the ReplyPattern element value indicates that the Callback Acknowledgement pattern is requested. If present, the ReplyTo attribute is required to be URL as defined in [RFC 1738].

3.2 Request Element

The ReliableMessage element is a REQUIRED element. It includes specific information to be used for a reliable message and includes the following attributes and child elements:

- a SOAP mustUnderstand attribute with a value of “1”

- **AckRequested** element

- **DuplicateElimination** element

- **MessageOrder** element

Example 6 shows an example of ReliableMessage element.

**Example 6  Example of ReliableMessage element**

(Example are included later)

3.2.1 AckRequested Element

The AckRequested element is an OPTIONAL element. It is REQUIRED for guaranteeing message delivery and message order. This element is to be used for a sender to request the receiver to send back an Acknowledgment or RM-Fault message for the message sent. If a
receiver received a message with AckRequested element, the receiver is REQUIRED to send an
Acknowledgment message whether or not it is a duplicate.

The pattern used to send the Acknowledgement or RM-Fault is based on the value of the
RMReplyPattern element.

3.2.2 DuplicateElimination Element

The DuplicateElimination element is used to require the receiver node to identify duplicate
messages it has received and process them accordingly (see section 2.3). A duplicate message is
a message with the same Message Identifier as another message. This element is OPTIONAL. It
is REQUIRED when duplicate elimination is mandated. If the MessageOrder element is present,
the DuplicateElimination element MUST also be present.

3.3 Response Element

The Response element includes response information to be used for an Acknowledgment
message or Fault message. It is REQUIRED only when the message includes an
Acknowledgment message or a Fault message. This element includes the following attribute and
sub-elements:

- a SOAP mustUnderstand attribute with a value of “1”
- RefToGroupId element
- RefToSequenceNo element

This Response element can co-exist with ReliableMessage element, and it enables to send back
Acknowledgment message with the business response to the original message. And it also
enables receiver send an another independent message to the sender with Acknoledgment
message to reduce network traffick.

Example 7 shows an example of Response element.

Example 7  Example of Response element

(Example will be added later, when the schema was decided)

3.3.1 RefToGroupId Element

The RefToGroupId element is a REQUIRED element. This element MUST contain the value of
the original GroupId of the message received successfully when used in the Acknowledgment
message, or for the message in error, when used in the Fault Message.

3.3.2 RefToSequenceNo Element

The RefToSequenceNo element is a REQUIRED element for Acknowledgment message or Fault
message when the original message was delivered with Guaranteed Message Ordering. This
element MUST contain the value of the original SequenceNumber of the message received
successfully when used in the Acknowledgment message, or for the message in error, when used
in the Fault Message.
4 SOAP Fault (Need updating)

This section describes extensions to the fault codes defined in the SOAP 1.1 specification. Intended to carry error or status information for the SOAP layer, these fault code extensions MUST comply with SOAP Fault as defined in SOAP1.1. The SOAP Fault is used in this model for notification of only SOAP level errors and Reliable Messaging errors. Errors specific to Reliable Messaging are described in the following sections.

4.1 SOAP Fault extension for Reliable Messaging

To describe the details of the Reliable Messaging error, an additional RMFault element is defined as a sub-element of SOAP Header.

4.1.1 Fault element

This element is OPTIONAL and if present MUST appear within a SOAP Header element. It contains only one sub-element:

- FaultCode: To specify Reliable Messaging specific fault value

4.1.2 FaultCode element

This sub-element is REQUIRED and SHOULD have a value specified in Chart 1. The value should be namespace qualified. These fault codes are explained in detail in section 4.2.
## Chart 1 FaultCode values

<table>
<thead>
<tr>
<th>Value of FaultCode</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>InvalidMessageHeader</td>
<td>Content or format of the Message Header element is invalid, or it was impossible to process the MessageHeader element for some reason.</td>
</tr>
<tr>
<td>InvalidMessageIdentifier</td>
<td>Content or format of the Message Identifier is invalid, or it was impossible to process the Message Identifier for some reason.</td>
</tr>
<tr>
<td>InvalidRefToMessageId</td>
<td>Content or format of the RefToMessageId element is invalid, or it was impossible to process the RefToMessageId element for some reason.</td>
</tr>
<tr>
<td>InvalidTimestamp</td>
<td>Content or format of the Timestamp element is invalid, or it was impossible to process the Timestamp element for some reason.</td>
</tr>
<tr>
<td>InvalidExpiryTime</td>
<td>Content or format of the ExpiryTime element is invalid, or it was impossible to process the ExpiryTime element for some reason.</td>
</tr>
<tr>
<td>InvalidReliableMessage</td>
<td>Content or format of the ReliableMessage element is invalid, or it was impossible to process the ReliableMessage element for some reason.</td>
</tr>
<tr>
<td>InvalidAckRequested</td>
<td>Content or format of the AckRequested element is invalid, it was impossible to process the AckRequested element for some reason, or the receiver couldn’t send back Acknowledgment Message as it was specified in the synchronous attribute.</td>
</tr>
<tr>
<td>InvalidMessageOrder</td>
<td>Content or format of the MessageOrder element is invalid, or it was impossible to process the MessageOrder element for some reason.</td>
</tr>
</tbody>
</table>
4.2 Fault code description

The following sections describe, in more detail, use of the error codes in Chart 5-1.

4.2.1 InvalidMessageHeader

This is an error message to be used when the content or format of the MessageHeader is invalid. This error message will be used also when the type attribute specified in the Service Element was not found. This error message also will be used to report receipt of invalid information in the From element or the To element.

4.2.2 InvalidMessageId

This is an error message to be used when the content or format of the Message Identifier is invalid.

4.2.3 InvalidRefToMessageId

This is an error message to be used when the content or format of the RefToMessageId element is invalid. This is also for use when no message with a specific Message Identifier, as referred to by the RefToMessageId element, is found.

4.2.4 InvalidTimestamp

This is an error message to be used when the content or format of the Timestamp element is invalid.

4.2.5 InvalidEpiryTime

This is an error message to be used when the content or format of the ExpiryTime element is invalid. This will be used also when a message is expired according to the value of ExpiryTime element.

4.2.6 InvalidReliableMessage

This is an error message to be used when the content or format of the ReliableMessage element is invalid.

4.2.7 InvalidAckRequested

This is an error message to be used when the content or format of the AckRequested element is invalid.
4.2.8 InvalidMessageOrder

This is an error message to be used when a content or format of MessageOrder element is invalid. This includes an error for wrong SequenceNumber element or its attributes, and the value of the SequenceNumber.
5 HTTP Binding (Needs to be updated)

This section describes the HTTP binding for SOAP, when the original message is sent asynchronously at the application level. The WS-Reliability is expecting that the transport layer will not deliver a corrupted message to the reliability layer. When supporting reliable messaging, upon receipt of a reliable message, the server MUST send a reply. This reply MUST be either an Acknowledgment or a Fault message. This reply MUST be sent either synchronously or asynchronously.

5.1 Reliable Messaging with Synchronous Acknowledgment or Fault Message

The Reliable Messaging Acknowledgment or Fault message MAY be sent back on the same HTTP connection as the HTTP Request that included the message being acknowledged or faulted. This is illustrated in Figures 5 and 6. The SOAP Fault MAY also be sent back asynchronously on a different HTTP connection, as illustrated in Figure 7.

(1) Synchronous Acknowledgment Message Sequence

Figure 5 Synchronous Acknowledgment Message

1) The sender initiates an HTTP connection, and sends a Message using the HTTP POST Request. The Example 9 is an example of such a message.

2) The receiver sends back an Acknowledgment message to the sender on the same connection, as the HTTP response. There are two options – synchronous and asynchronous – for sending back the SOAP Fault to the sender. The following sections describe it in detail.

(2) Synchronous SOAP Fault Message Sequence

Figure 6 Synchronous SOAP Fault Message

Copyright © OASIS Open 2002. All Rights Reserved.

Page 19 of 27
(3) Asynchronous SOAP Fault Message Sequence

Figure 7 Asynchronous SOAP Fault Message

(1) HTTP Request
(2) HTTP Response
(3) HTTP Request
(4) HTTP Response

5.2 Reliable Messaging with Asynchronous Acknowledgment Message

The Reliable Messaging Acknowledgment message MAY also be sent back on a different HTTP connection from the HTTP connection used to send the message being acknowledged. This is illustrated in Figure 8 and 9.

(1) Asynchronous Acknowledgment Message Sequence

Figure 8 Asynchronous Acknowledgment Message

(1) The sender initiates a HTTP connection, and sends a Message using HTTP POST Request. Example 9 is an example of this message.

(2) The HTTP response to the (1) has no content. Example 10 is an example of this HTTP response.

(3) The Acknowledgment Message is sent with the other HTTP connection. Example 11 is an example of this message.

(4) The HTTP response for (3) has no content. Example 10 is an example for this HTTP Response.

There are two options – synchronous and asynchronous – for sending back a SOAP Fault to the sender when an error is detected by the receiver. These are illustrated in figure 9 and 10 respectively.
(2) Synchronous Fault Message Sequence

Figure 9 Synchronous SOAP Fault Message

(3) Asynchronous Fault Message Sequence

Figure 10 Asynchronous SOAP Fault Message

Example 9 Example of Reliable Message within HTTP POST Request

( To be added later after schema was fixed.)
6 References

6.1 Normative

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

[ RFC2119 ] "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, Bradner, S., IESG and IETF, March 1997. Available at
http://www.ietf.org/rfc/rfc2119.txt

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

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

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

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

http://www.w3.org/TR/2000/REC-xml-20001006/

http://www.w3.org/TR/1999/REC-xml-names-19990114/
6.2 Non-normative References

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

http://www.w3.org/TR/2002/CR-soap12-part1-20021219/

http://www.oasis-open.org/committees/wss/

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

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

- David Ingham, Arjuna Technologies Limited
- Joseph Chiusano, Booz Allen Hamilton
- Peter Furniss, Choreology Ltd
- Jeff Turpin, Cyclone Commerce
- Pramila Mullan, France Telecom
- Jacques Durand, Fujitsu
- Kazunori Iwasa, Fujitsu
- Tom Rutt (chair), Fujitsu
- Jishnu Mukerji, Hewlett-Packard
- Robert Freund, Hitachi
- Eisaku Nishiyama, Hitachi
- Nobuyuki Yamamoto, Hitachi
- Ben Bloch, Individual
- Mark Hansen, Individual
- Paolo Romano, Individual
- Dock Allen, Mitre Corporation
- Junichi Tatemura, NEC Corporation
- Alan Weissberger, NEC Corporation
- Magdolna Gerendai, Nokia
- Szabolcs Payrits, Nokia
- Sunil Kunisetty, Oracle
- Jeff Mischkinsky, Oracle
- Marc Goodner, SAP
- Pete Wenzel, SeeBeyond Technology Corporation
- Doug Bunting, Sun Microsystems
- Chi-Yuen Ng, University of Hong Kong
- Patrick Yee, University of Hong Kong
In addition, the following people made contributions to this specification:

(TBD)
Appendix B. Revision History

[This appendix is optional, but helpful. It should be removed for specifications that are at OASIS Standard level.]

<table>
<thead>
<tr>
<th>Rev</th>
<th>Date</th>
<th>By Whom</th>
<th>What</th>
</tr>
</thead>
<tbody>
<tr>
<td>WD-0.5</td>
<td>2003-09-04</td>
<td>Kazunori Iwasa</td>
<td>Initial version</td>
</tr>
</tbody>
</table>
Appendix C. Notices

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

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

Copyright © OASIS Open 2003. All Rights Reserved.

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

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

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