RM4GS Overview

March 2004

FUJITSU LIMITED　Hitachi, Ltd.　NEC Corporation

1. Introduction

The RM4GS alpha version is a sample implementation of the reliable messaging specification “WS-Reliability”[1] for Web Service. This guide describes why reliable messaging is required in Web Service, gives an overview of WS-Reliability and an overview of the RM4GS.

Note: Since this software is still an alpha version, its API is not described in contained documents in this package. The API will be described in the next version of RM4GS.

2. SOAP Reliability Issue

SOAP[2] has been widely used as a standard communication protocol for Web Services. Because the standard organization W3C released SOAP specification as W3C Note, and because many IT vendors implemented it and executed interoperability test among different SOAP implementations, some degree of interoperability among Web Services has been achieved at SOAP level.

However applying SOAP to Web Services does not help in developing reliable Web Service systems: message data may be lost if the partner’s server goes down during communication, or message data may be duplicated when restarting. Because of these problems, SOAP is not reliable.

When trying to resolve this SOAP issue, Internet specific requirements and context need be taken into consideration. The characteristics of distributed systems on the Internet are different from those on intranet. Not only the business partners belong to separate companies, but also the partners have different system management rules and middlewares from yours. Thus the reliability on the Internet should be realized in a flexible way, with loose coupling.

3. What is Reliable Messaging ?

“Reliable messaging” in this overview assumes a technology that allows applications to exchange business documents without direct access to the partner’s application. Instead, the sending application passes its message to a middleware, and the middleware delivers the message to the receiving application (see Figure 1). The middleware that provides the reliable messaging functions is called MOM (Message Oriented Middleware). It has been used in intranet. Reliable messaging has the following advantages:

- Reliability

  The middleware stores messages in a persistent storage, and re-send messages even if a communication error occurs. More precisely, this means that the middleware supports three fundamental functions for reliability: guarantee of message delivery, prevention of message duplication and guarantee of message order.
- Flexibility

  Communication can operate without requiring both parties to be operational or available at the same time. This semantics is called asynchronous communication.

Since reliable messaging implies both the reliability and flexibility, it has been applied to Web Services in order to resolve the previous SOAP issue.

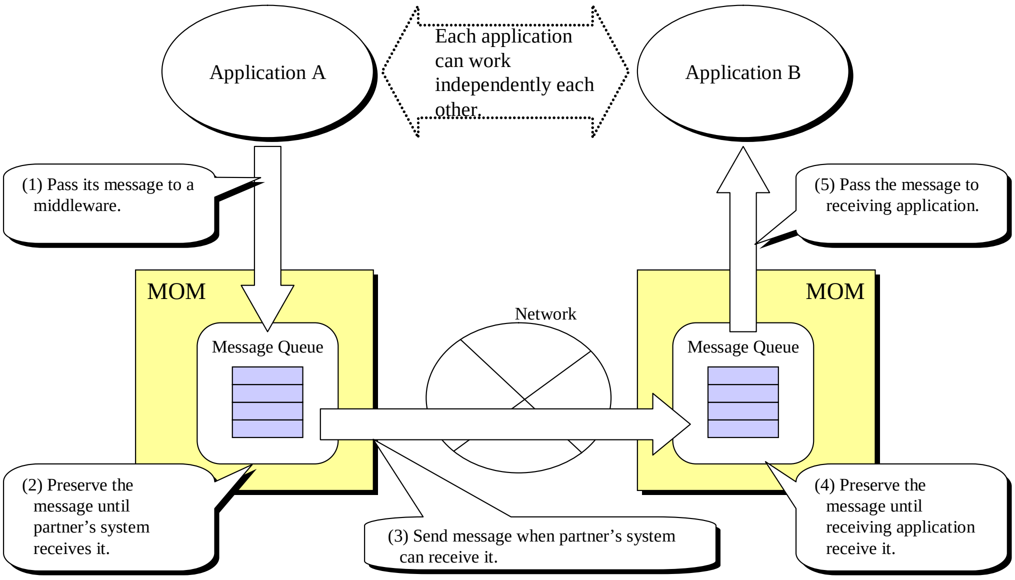
Application A
Each application can work independently each other.
Application B
(1) Pass its message to a middleware.
(5) Pass the message to receiving application.
MOM
Network
MOM
Message Queue
Message Queue
(2) Preserve the message until partner's system receives it.
(3) Send message when partner's system can receive it.
(4) Preserve the message until receiving application receive it.

Figure 1 Principle of Reliable Messaging

4. The Reliable Messaging Specification for Web Service : WS-Reliability

Reliable messaging to Web Services originated in the B2B area. The reliable messaging protocol "ebXML Messaging Service"[3], adopted in the B2B-focused standard "ebXML", is an extension of SOAP including reliable messaging functions and security functions. Many IT vendors have implemented it in their products and some consortiums (ex. ebXML Asia) have organized interoperability test among different vendor's implementations. The

WS-Reliability specification was created based on the ebXML Message Service's reliability functions for all types of Web Service. The reliability techniques developed in ebXML were not limited in scope to B2B applications. They have been improved in WS-Reliability. WS-Reliability has the following characteristics:

(1) Standardized by OASIS

WS-Reliability has been created by the Web Services Reliable Messaging Technical Committee in the standard organization OASIS. It is an open and royalty free specification. It is in the final stage of its standardization process in OASIS and it is expected that OASIS will adopt WS-Reliability as an OASIS Standard in the 2nd quarter of this year.

(2) Support for the most common Reliable Messaging features

WS-Reliability provides required QoS (Quality of Service) for reliable messaging:

- Guarantee of message delivery

  When a message fails to be delivered to the target system, the failure is detected and the message is resent automatically, until the message arrives or a timeout is reached.

- Prevention of Message Duplication

  When duplicate messages are received by the target system, duplicate messages are removed automatically. The combination of the "Guarantee of message delivery" and this function "Prevention of Duplication Message" realizes "exactly-once" semantics.

- Guarantee of Message Order

  Messages are delivered to the target application in the same order they were submitted by the sending application.

(3) Support for both Synchronous and Asynchronous communication

WS-Reliability supports not only asynchronous communication semantics for reliable messaging but also synchronous communication semantics.

(4) Compliance with SOAP 1.1

WS-Reliability is compliant with SOAP 1.1 specification.

(5) Combination with other Web Service Specifications

WS-Reliability can be used with other Web Service specifications (ex. WS-Security).

5. Reliable Messaging Implementation for Web Service: RM4GS

RM4GS is an implementation of the WS-Reliability described above with Java. RM4GS has the following features:

(1) Compliance with WS-Reliability

RM4GS is compliant with WS-Reliability V0.52[4]. It supports reliability functions specified in the specification.

(2) 100% Pure Java

RM4GS was entirely developed with Java language.

(3) Interworking with EJB

RM4GS is implemented as a resource adapter defined in JCA (J2EE Connector Architecture) specification. It can work with EJB. Received messages can be handled by MDB (Message Driven Bean).

(4) Extensibility

RM4GS adopts a layer structure with internal interface. Its functions can be extended by adoption or change of components in every layer (see Figure 2).

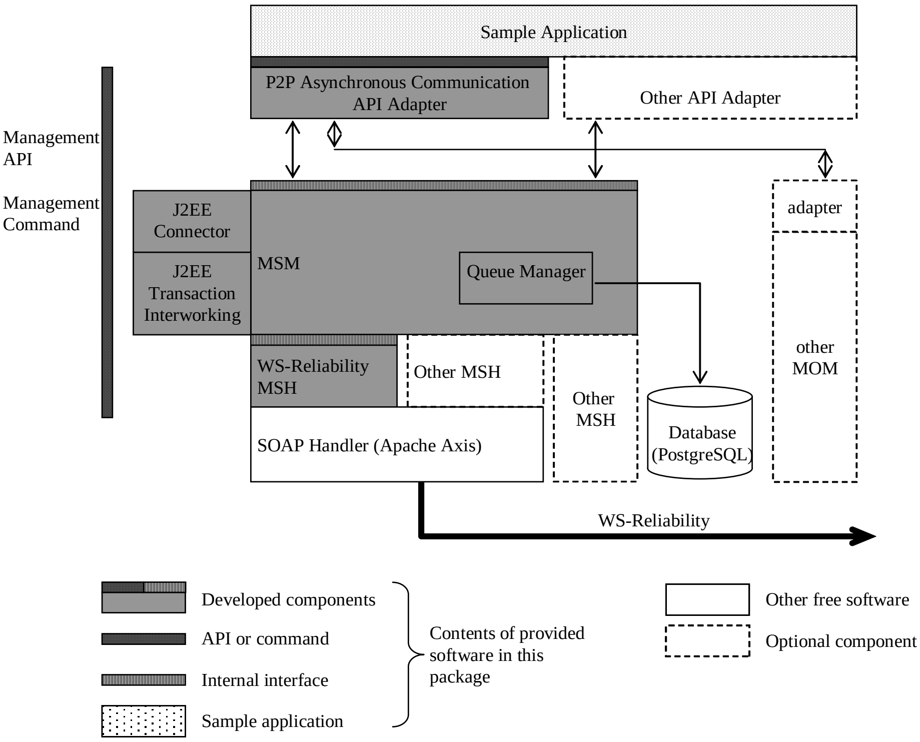

Figure 2 Architecture of RM4GS

Each RM4GS instance supports the following components and functions:

- P2P Asynchronous Communication API Adapter

  This component provides an API for asynchronous communication. It has both sending message functions and receiving message functions. Since it provide asynchronous semantics, the sending application can work independently from the eceiving application. Thus the API call from the sending application for sending a message will complete before the receiving application receives the message.
- MSM (Message Service Manager)

  This component provides asynchronous messaging functions.
- MSH (Message Service Handler)

  This component provides a packaging function for specific communication protocol format. It is located at the lowest layer and work with an underlying protocol handler such as SOAP handler. WS-Reliability MSH generates message packaging conforms to the WS-Reliability specification.

The provided RM4GS alpha version package includes a sample application which demonstrates reliable messaging functions. The package allows for: initiating messaging between two machines, simulating network troubles, and provides monitoring functions that allow for verifying how reliability functions work. Please read the “RM4GS Install Guide” (install.pdf) for installation of RM4GS, and the “RM4GS Sample Applications” (application.pdf) for details on the sample application.

References

[1] Kazunori Iwasa et al.: WS-Reliability Working Draft 0.98, OASIS (2004), <http://www.oasis-open.org/committees/download.php/5674/WS-Reliability-2004-02-26.pdf> (latest working draft)

[2] Don Box et al.: W3C Note - Simple Object Access Protocol (SOAP) 1.1, The World Wide Web Consortium (2000), <http://www.w3.org/TR/soap/>

[3] Ian Jones et al.: Message Service Specification Version 2.0. OASIS (2002), <https://www.oasis-open.org/committees/ebxml-msg/documents/ebMS_v2_0.pdf>

[4] Kazunori Iwasa et al.: WS-Reliability Working Draft 0.52, OASIS (2003), <http://www.oasis-open.org/committees/download.php/4474/WS-Reliability-2003-11-25%28CD%29.pdf> (latest committee draft)