RM4GS Sample Applications Guide

March 2004

FUJITSU LIMITED Hitachi, Ltd. NEC Corporation

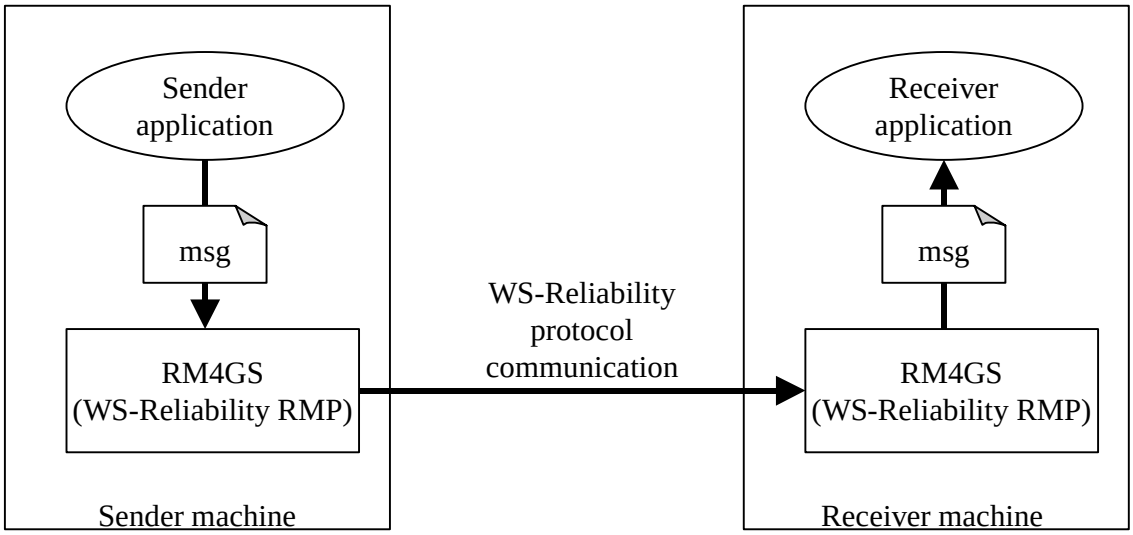
Abstract

This document contains information about RM4GS sample applications.

1. System configuration

1.1 System structure

This system is consisted from two machines. One is sender, and the other is receiver.



First, install RM4GS for two machines. Please read RM4GS install guide.

1.2 Setting applications

1.2.1 Unpacking sampleapplications.tar

Before Unpacking, change Linux user to root user.

Unpack sampleaplications.tar at both machines.

It contains:

1) /sampleapplications/build.xml (ant build file)
2) /sampleapplications/build.properties (ant properties file)
3) /sampleapplications/samplereceivebean.ear (receiver application)
4) /sampleapplications/samplesendbean.ear (sender application)

1.2.2 Edit the build.properties file

Edit the build.properties file at both machines.

#	Property	Explanation
1	rm4gs.home	Set RM4GS install directory
2	j2ee.home	Set J2EE install directory
3	j2ee.domain	Set J2EE domain name
4	admin.user	Set J2EE admin user name
5	admin.password	Set J2EE admin user password
6	receiver. DestinationBaseURL	Set receiver machine's DestinationBaseURL. (See RM4GS install guide 4.2.1)

		This property is required only on the sender machine

1.2.3 Install sender application

These operations are performed at the sender machine.

1) Change Linux user to root user.

2) Starting RM4GS.

3) Change directory to /sampleapplications.

4) Enter command:

```
asant install_samplesendbean
```

1.2.4 Install receiver application

These operations are performed at the receiver machine.

1) Change Linux user to root user.

2) Starting RM4GS.

3) Change directory to /sampleapplications.

4) Enter command:

```
asant install_samplereceivebean
```

1.2.5 Uninstall sender application

These operations are performed at the sender machine.

1) Change Linux user to root user.

2) Restart RM4GS.

3) Change directory to /sampleapplications.

4) Enter command:

```
asant uninstall_samplesendbean
```

1.2.6 Uninstall receiver application

These operations are performed at the receiver machine.

1) Change Linux user to root user.

2) Restart RM4GS.

3) Change directory to /sampleapplications.

4) Enter command:

```
asant uninstall_samplereceivebean
```


2. System startup

2.1 Starting the sender application

These operations are performed at the sender machine.

1) Change Linux user to root user.

2) Restart J2EE application server.

3) Change directory to

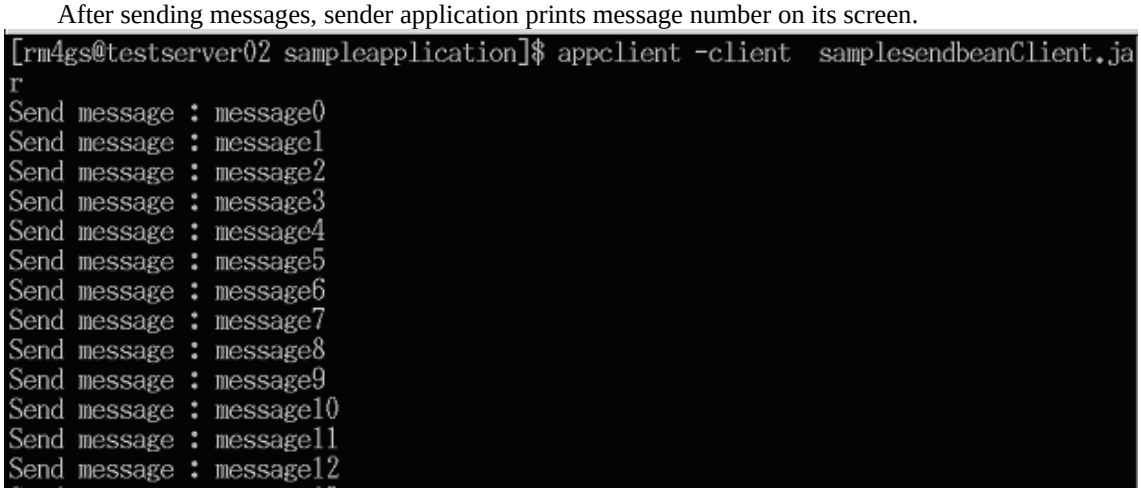
`$J2EE_HOME/domains/domain1/applications/j2ee-apps/samplesendbean`

4) Enter command:

`appclient –client samplesendbeanClient.jar`

Sender application sends text-type messages for RM4GS, every 5 seconds.

After sending messages, sender application prints message number on its screen.



2.2 Starting the receiver application

These operations are performed at the receiver machine.

1) Change Linux user to root user.

2) Restart J2EE application server.

3) Change directory to

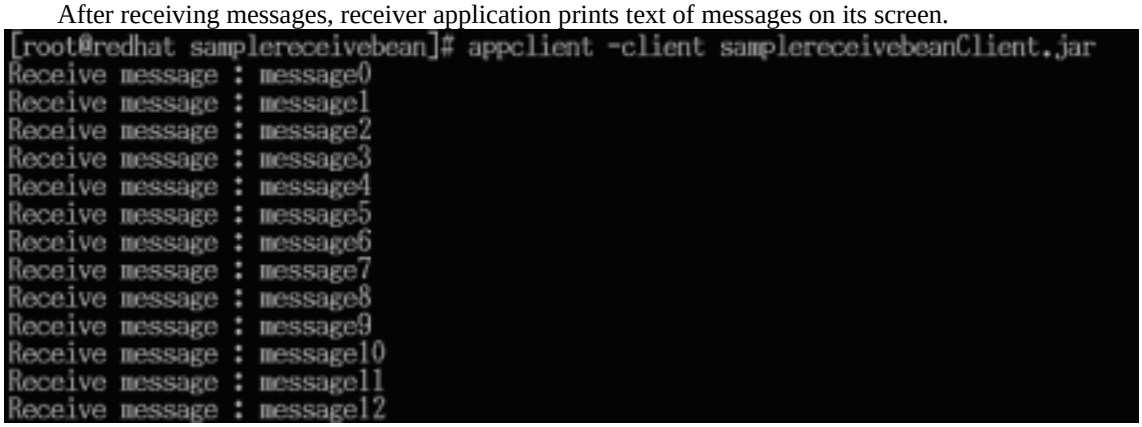
`$J2EE_HOME/domains/domain1/applications/j2ee-apps/samplereceivebean`

4) Enter command:

`appclient –client samplereceivebeanClient.jar`

Receiver application receives messages from RM4GS.

After receiving messages, receiver application prints text of messages on its screen.



2.3 Log files of RM4GS

RM4GS records a small amount of information, referred to as "log", as to transferred messages when sending or receiving via the network. .Messaging logs will appear in the application server's log file, which is often found at ${J2EE_HOME}/domains/domain1/logs/server.log.

RM4GS writes three log records:, a record to show message destination, a record to show message identity and a record to show the SOAP headers of sent messages.

Example of Sending Records follows:

First Record Example:

[#|2004-02-24T11:51:22.583+0900|INFO|j2ee-appserver1.4|org.bizgrid.rm4gs.msh|_ThreadID=12;|RM4GS sending message to http://receiver:8080/rm4gs/SimpleQueue|#]

This record shows the destination of the message. Destination is displayed as a URL, "http://receiver:8080/rm4gs/SimpleQueue".

Second Record Example:

[#|2004-02-24T11:51:22.583+0900|INFO|j2ee-appserver1.4|org.bizgrid.rm4gs.msh|_ThreadID=12;|ACK Message: RefToGroupId = rm4gs10775920349921: RefToSequenceNumber = 0|#]

This record shows the message type , "Normal Message" or "ACK Message". .”Normal Message” is sent on invocation of the send API method and contains the application's message content. “ACK Message” is sent automatically to comply with WS-Reliability protocol.

In the case of a “Normal Message”, Message identity, GroupdId value and SequenceNumber follow in order. After an “Ack Message”, follows the identity of the message to which the “Ack Message” is sent.

Third Record Example:

The third record consists of a SOAP header image.

[#|2004-02-24T11:51:22.698+0900|INFO|j2ee-appserver1.4|org.bizgrid.rm4gs.msh|_ThreadID=12;|Sending WS-Reliability Message.Its SOAP Header is: <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:RM="http://www.oasis-open.org/committees/wsrm/schema/1.1/SOAP1.1">
<soapenv:Header>
<RM:MessageHeader soapenv:mustUnderstand="1"/>
<RM:RequestElement soapenv:mustUnderstand="1"/>
<RM:ResponseElement soapenv:mustUnderstand="1">
<RM:RefToGroupId>rm4gs10775920349921</RM:RefToGroupId>
<RM:RefToSequenceNumber>0</RM:RefToSequenceNumber>
</RM:ResponseElement>
</soapenv:Header>
<soapenv:Body><ns1:e1 xmlns:ns1="urn:rm4gs">rm4gs-dummy-message</ns1:e1> </soapenv:Body>
</soapenv:Envelope>|#]

On receipt of a WS-Reliability message, RM4GS also writes three records to the log, but in a different order.

First Record:
On the receiver side,the first record consists of a SOAP Header.

[#|2004-02-24T11:51:21.207+0900|INFO|j2ee-appserver1.4|org.bizgrid.rm4gs.msh|_ThreadID=11;|Received WS-Reliability Message.Its SOAP Header is: <soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:RM="http://www.oasis-open.org/committees/wsrm/schema/1.1/SOAP1.1">
<soapenv:Header>
<RM:MessageHeader soapenv:mustUnderstand="1">
<RM:GroupId removeAfter="2004-02-25T03:07:15.113Z">rm4gs10775920349921</RM:GroupId>
<RM:SequenceNumber status="Start">0</RM:SequenceNumber>
<RM:Timestamp>2004-02-24T03:07:15.113Z</RM:Timestamp>
<RM:ExpiryTime>2004-02-24T04:07:15.113Z</RM:ExpiryTime>
<RM:ReplyPattern ReplyTo="http://sender:8080/rm4gs/SimpleQueue">Callback</RM:ReplyPattern>
</RM:MessageHeader>
<RM:RequestElement soapenv:mustUnderstand="1">
<RM:AckRequested/>

<RM:DuplicateElimination/>
</RM:RequestElement>
<RM:ResponseElement soapenv:mustUnderstand="1"/>
</soapenv:Header>
<soapenv:Body><ns1:e1 xmlns:ns1="urn:rm4gs">rm4gs-dummy-message</ns1:e1> </soapenv:Body>
</soapenv:Envelope>|#]

Second Record:
[#|2004-02-24T11:51:21.590+0900|INFO|j2ee-appserver1.4|org.bizgrid.rm4gs.msh|_ThreadID=11;|RM4GS message received for http://receiver:8080/rm4gs/SimpleQueue|#]

The second record in the receiver's log contains the destination which is specified by the sender in URL form.

Third Record:
[#|2004-02-24T11:51:21.591+0900|INFO|j2ee-appserver1.4|org.bizgrid.rm4gs.msh|_ThreadID=11;|Normal Message: GroupId = rm4gs10775920349921: SequenceNumber = 0|#]

The third record in the receiver's log contains type of message and message identity which cooresponds to that which is described in the sender's second record above.

2.4 Verify reliable messaging

After system startup, if one causes a network fault such as unplugging a network cable etc, ,RM4GS retries the send operation.
RM4GS ensures message delivery.
Verify correct operation by examining the application consoles and RM4GS log files.

To run the verification operations again, you need to shutdown the whole system (both RM4GS and applications on both machines) completely.
After all shutdown operations have completed, start from 2. System startup operations again.

3. Applications source code

3.1 Sender application

3.1.1 Samplesendbean

```
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;
import javax.ejb.SessionContext;
import javax.ejb.SessionBean;
import javax.ejb.CreateException;
import java.rmi.RemoteException;

import org.bizgrid.rm4gs.Connection;
import org.bizgrid.rm4gs.ConnectionFactory;
import org.bizgrid.rm4gs.MessageType;
import org.bizgrid.rm4gs.Policy;
import org.bizgrid.rm4gs.SendingSession;
import org.bizgrid.rm4gs.TextMessage;
import org.bizgrid.rm4gs.P2PDestination;
import org.bizgrid.rm4gs.RM4GSException;

public class samplesendbean implements SessionBean {
   private SessionContext sc     = null;
   private Connection conn       = null;
   private P2PDestination dest   = null;
   private SendingSession session = null;
   private TextMessage msg       = null;
   private long count           = 0;

   public void ejbCreate() throws CreateException, RemoteException {
      try {
         InitialContext ctx = new InitialContext();
         Object obj = ctx.lookup("eis/rm4gs");
         ConnectionFactory cf =
             (ConnectionFactory) PortableRemoteObject.narrow(
               obj,ConnectionFactory.class);
         conn = cf.getConnection(false, ConnectionFactory.AUTO_ACKNOWLEDGE);
         dest = conn.createP2PDestination("samplequeue");
         Policy[] policies = new Policy[] {Policy.IN_ORDER};
         session = conn.createSendingSession(dest, policies);
```



```
            msg = (TextMessage)conn.createMessage(MessageType.TEXT);
        } catch(Exception e) {
            throw new CreateException(e.toString());
        }
    }

    public void setSessionContext(SessionContext sc) {
        this.sc = sc;
    }

    public String doSend() throws RemoteException {
        try {
            msg.setMessage(new String("message") + count);
            session.send(msg);
            ++count;
            return msg.getMessage();
        } catch(Exception e) {
            throw new RemoteException(e.toString());
        }
    }

    public void ejbRemove() {
        try {
            msg.release();
            session.close();
            conn.close();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }

    public void ejbActivate() {   }
    public void ejbPassivate() {   }
    public samplesendbean() {   }
}
```


3.1.2 samplesendclient

```
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;
import javax.ejb.CreateException;
import java.rmi.RemoteException;

public class samplesendclient {
    public static void main(String[] args) throws Exception {
        try {
            InitialContext ctx = new InitialContext();
            Object obj = ctx.lookup("ejb/samplesendbean");
            samplesendbeanhome home =
                (samplesendbeanhome)PortableRemoteObject.narrow(
                obj,samplesendbeanhome.class);
            samplesendbeanremote sendbean = home.create();
            while(true) {
                try {
                    String msg = sendbean.doSend();
                    System.out.println("Send message : " + msg);
                    Thread.sleep(5000);
                } catch(RemoteException e) {
                    sendbean.remove();
                    throw e;
                } catch(Exception e) {
                    sendbean.remove();
                    throw e;
                }
            }
        } catch(CreateException e) {
            e.printStackTrace();
            throw new Exception(e.toString());
        } catch(Exception e) {
            e.printStackTrace();
            throw e;
        }
    }
}
```


3.2 Receiver application

3.2.1 Samplereceivebean

```
import java.rmi.Naming;
import java.rmi.RemoteException;
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;
import javax.ejb.SessionContext;
import javax.ejb.SessionBean;
import javax.ejb.CreateException;

import org.bizgrid.rm4gs.Connection;
import org.bizgrid.rm4gs.ConnectionFactory;
import org.bizgrid.rm4gs.ReceivingSession;
import org.bizgrid.rm4gs.TextMessage;
import org.bizgrid.rm4gs.Destination;
import org.bizgrid.rm4gs.ReceiveTimeoutException;
import org.bizgrid.rm4gs.RM4GSException;

public class samplereceivebean implements SessionBean {
   private SessionContext sc      = null;
   private Connection conn        = null;
   private Destination dest       = null;
   private ReceivingSession session = null;
   private long count             = 0;

   public void ejbCreate() throws CreateException, RemoteException {
      try {
         ConnectionFactory cf = (ConnectionFactory)Naming.lookup("rm4gs");
         conn = cf.getConnection(false, ConnectionFactory.AUTO_ACKNOWLEDGE);
         dest = conn.createP2PDestination("samplequeue");
         session = conn.createReceivingSession(dest, null);
         conn.start();
      } catch(Exception e) {
         throw new CreateException(e.toString());
      }
   }

   public void setSessionContext(SessionContext sc) {
```



```
        this.sc = sc;
    }

    public String doReceive() throws RemoteException {
        TextMessage msg = null;
        while(true) {
            try {
                Object obj = session.receive(5000);
                if (obj instanceof TextMessage) {
                    msg = (TextMessage)obj;
                    return msg.getMessage();
                }
            } catch(ReceiveTimeoutException e) {
                continue;
            } catch(Exception e) {
                throw new RemoteException(e.toString());
            } finally {
                if (msg != null) {
                    msg.release();
                }
            }
        }
    }

    public void ejbRemove() {
        try {
            session.close();
            conn.stop();
            conn.close();
        } catch(Exception e) {
            e.printStackTrace();
        }
    }

    public void ejbActivate() {   }
    public void ejbPassivate() {   }
    public samplereceivebean() {   }
}
```

3.2.2 samplereceiveclient

```
import javax.naming.InitialContext;
import javax.rmi.PortableRemoteObject;
import javax.ejb.CreateException;
import java.rmi.RemoteException;

public class samplereceiveclient {
    public static void main(String[] args) throws Exception {
        try {
            InitialContext ctx = new InitialContext();
            Object obj = ctx.lookup("ejb/samplereceivebean");
            samplereceivebeanhome home =
                (samplereceivebeanhome)PortableRemoteObject.narrow(
                obj,
                samplereceivebeanhome.class);
            samplereceivebeanremote receivebean = home.create();
            while(true) {
                try {
                    String msg = receivebean.doReceive();
                    System.out.println("Receive message : " + msg);
                } catch(RemoteException e) {
                    receivebean.remove();
                    throw e;
                } catch(Exception e) {
                    receivebean.remove();
                    throw e;
                }
            }
        } catch(CreateException e) {
            e.printStackTrace();
            throw new Exception(e.toString());
        } catch(Exception e) {
            e.printStackTrace();
            throw e;
        }
    }
}
```

