# IGATE Global Solutions Ltd

# RMI

## Lab Guide

# Table of Contents

# 1 LAB 1: DEVELOPING RMI APPLICATION

| | |
|---|---|
| Goals | • Writing Remote Interface, Remote Server<br>• Compiling and generating Stubs<br>• Writing Client Application<br>• Executing RMI call |
| Time | 60 Min |
| Lab Setup | Windows OS with JDK1.5, Eclipse 3.0+ installed |

Develop a simple distributed 'Hello World' application using RMI. Business logic method is greet(String name) which is implemented by the RMI Server HelloImpl.java

**Step -1: Writing Remote Interface**

Write an interface HelloIntf which extends java.rmi.Remote as follows :

```
public interface HelloIntf extends Remote
{
        public String greet(String name) throws RemoteExcetion;
}
```

**Step -2: Writing Remote Server**

```
        package IGATE.rmi;

import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.RemoteException;
import java.rmi.server.UnicastRemoteObject;

public class HelloImplServer extends UnicastRemoteObject implements HelloInft {

        protected HelloImplServer() throws RemoteException {
                super();
        }

        public String greet(String name) throws RemoteException {
                return "Hello " + name;
        }

        public static void main(String[] args) {

        // Create and install a security manager
          if (System.getSecurityManager() == null) {
            System.setSecurityManager(new RMISecurityManager());
          }

        try {
            HelloImplServer helloObj = new HelloImplServer();
            Naming.rebind("rmi://localhost/helloServer", helloObj);
        } catch (RemoteException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        } catch (MalformedURLException e) {
            // TODO Auto-generated catch block
            e.printStackTrace();
        }
    }
}
```

**Step -3: Compilation and generation of Stub/Skeleton.**

Since we are using Eclispe the classes get generated online. To generate stub/skeleton for RMI open command prompt and set path to bin directory of JDK and classpath to lib dir as follows in path of your project directory :

```
set path=C:\Program Files\Java\jdk1.5.0_06\bin
set classpath=C:\Program Files\Java\jdk1.5.0_06\lib\tools.jar;.
```

**Generate stub/skeleton as follows :**

```
rmic IGATE.rmi.HelloImplServe
```

This completes Server side implementation.

Step -4: Write a distributed client to display “Hello Mr. M. Singh” on the console window as follows :

```
    package IGATE.rmi.client;

import java.net.MalformedURLException;
import java.rmi.Naming;
import java.rmi.NotBoundException;
import java.rmi.RemoteException;

import IGATE.rmi.HelloIntf;

public class HelloClient {

    public static void main(String[] args) {

        HelloClient clientObj = new HelloClient();
        try {
            HelloIntf remote = (HelloIntf)Naming.lookup("rmi://localhost/helloServer");
            System.out.println(remote.greet("Mr. M. Singh"));
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (RemoteException e) {
            e.printStackTrace();
        } catch (NotBoundException e) {
            e.printStackTrace();
        }
    }

}
```

Step -5: Start RMI Registry and execute the RMI client. Type following at the command prompt

```
start rmiregistry
```

This will open into a new command prompt. With this service running in background, run HelloImplServer from Eclipse and simultaneously execute HelloClient from Eclispe, you must see the message “Hello, Mr. M. Singh” in console window.

**Reference: http://java.sun.com/j2se/1.3/docs/guide/rmi/getstart.doc.html**

## 2 LAB 2: DEVELOPING RMI APPLICATION WITH GUI CLIENT

| | |
|---|---|
| Goal | • Create Remote Interface, Remote Server<br>• Create Applet as Remote Client.<br>• Compilation and generation of classes and stubs<br>• Execute RMI application using Applet |
| Time | 60 Minutes |
| Lab Setup | Windows OS with JDK1.4 installed, Eclipse 3.0+. |

**Write RMI application which calculates and displays net monthly salary of an employee.**
**Client application is an Applet which accepts Employee Number, Annual Basic salary, annual tax deduction, annual medical and annual HRA components.**