University of Ruhuna
Faculty of Technology
Department of Informtion and Communication Technology

Advanced Programming Practicum
(ICT3122)

Lab Sheet 13 18th March 2024

Remote Method Invocation (RMI) in Java

Task 1

1. Introduction to RMI:

- Briefly explain the concept of distributed computing and the need for RMI.
- Describe how RMI allows Java objects to communicate across different JVMs (Java Virtual Machines) over a network.
- Discuss the components of RMI: remote interface, remote object, stub, and skeleton.

2. Setting Up the Environment:

- Create separate packages for the server and client applications.

3. Defining the Remote Interface:

- Create a new Java interface named **RemoteCalculator** that extends the **java.rmi.Remote** interface.
- Define the following methods within the **RemoteCalculator** interface:
    - **int add(int a, int b) throws java.rmi.RemoteException;**
    - **int subtract(int a, int b) throws java.rmi.RemoteException;**
    - **int multiply(int a, int b) throws java.rmi.RemoteException;**
    - **int divide(int a, int b) throws java.rmi.RemoteException;**

4. Implementing the Remote Object:

- Create a class named **CalculatorImpl** that implements the **RemoteCalculator** interface.
- Implement the methods defined in the **RemoteCalculator** interface within the **CalculatorImpl** class.
- Annotate the **CalculatorImpl** class with **java.rmi.server.UnicastRemoteObject.exportObject()**.

5. Creating the Server Application:

- Develop a server application named **CalculatorServer** that binds the **CalculatorImpl** object to the RMI registry.
- Start the RMI registry using the **rmiregistry** command or programmatically within your **CalculatorServer** application.
- Register the **CalculatorImpl** object with the RMI registry using **java.rmi.registry.Registry.bind()**.

6. Developing the Client Application:

- Write a client application named **CalculatorClient** that looks up the **CalculatorImpl** object from the RMI registry.
- Obtain a reference to the **CalculatorImpl** object using **java.rmi.registry.Registry.lookup()**.
- Invoke methods on the **CalculatorImpl** object as if it were a local object.

7. Testing and Debugging:

- Run the RMI registry, server, and client applications.
- Ensure that the client can successfully invoke methods on the server and receive the expected results.
- Debug any issues that arise during testing, such as network connectivity problems or RMI configuration errors.

Task 2

1. **Setting Up the Environment:**
    - Create separate packges for the server and client applications.
2. **Define the Remote Interface:**
    - Create a Java interface named **BookstoreService** that extends the **java.rmi.Remote** interface.
    - Define the following methods within the **BookstoreService** interface:
        - **List<Book> searchBooks(String keyword) throws java.rmi.RemoteException;**
        - **boolean buyBook(int bookId, int quantity) throws java.rmi.RemoteException;**
3. **Implement the Remote Object:**
    - Create a class named **BookstoreServiceImpl** that implements the **BookstoreService** interface.
    - Implement the methods defined in the **BookstoreService** interface within the **BookstoreServiceImpl** class.
    - Annotate the **BookstoreServiceImpl** class with **java.rmi.server.UnicastRemoteObject.exportObject()**.
4. **Create the Server Application:**
    - Develop a server application named **BookstoreServer** that binds the **BookstoreServiceImpl** object to the RMI registry.
    - Start the RMI registry using the **rmiregistry** command or programmatically within your **BookstoreServer** application.
    - Register the **BookstoreServiceImpl** object with the RMI registry using **java.rmi.registry.Registry.bind()**.
5. **Develop the Client Application:**
    - Write a client application named **BookstoreClient** that looks up the **BookstoreServiceImpl** object from the RMI registry.

- Obtain a reference to the **BookstoreServiceImpl** object using **java.rmi.registry.Registry.lookup()**.
- Implement a user interface where clients can search for books by keyword and purchase them.

6. **Testing and Debugging:**
   - Run the RMI registry, server, and client applications.
   - Ensure that the client can successfully search for books and make purchases through the server.
   - Debug any issues that arise during testing, such as incorrect search results or purchase failures.