Remote Method Invocation Based on Personal Data Management System (Other Rank of Army Unit)

Kyu Kyu Win, Mya Than Hnin
University of Computer Studies, Mandalay
layeikpyar6@gmail.com

Abstract

This paper is intended to get transparent and efficient object invocation in distributed personal data management system. The system has used RMI framework. Distributed Object-Oriented Architectu--re, Remote Reference Layer and Remote Interface to develop distributed Personal Data Management System. This paper is concerned with method invocations that require the participation of multiple units where distributed personal data management system is located. A unit located at a network computer is exploited to access transparent remote method and require information from remote server by using the middle layer, command server.
Keyword: Distributed Object-Oriented Architect-ure, Remote Invocation, Remote Reference Layer, Remote Interface

1. Introduction

The key success of the distributed computing on distributed architecture is the ability to reduce the use of processing time and resources, produce a better performance than the traditional computing. A distributed system requires computations that are running in different address spaces, practically on different machines, must be able to communicate between one machines to another. High-performance computing applications so require the efficient management of information and also the reusability, efficient use of time and resources and the portability of the applications in wide-area distributed computing environments. Inexpensive machines interconnected by high speed communication networks are currently widely used for parallel computation and as backend processing servers for a growing number of commercial applications. Objects with high reliability requirements should be placed on machines with high estimated reliability. Moreover object technology provides a uniform mechanism for accessing local and remote resources and reduces the complexity of application of interactive applications.

This paper concerns an object/method invocation between personal data management system that are located in the three-tiered PDMS. In this system, there are two kinds of entities: Local personal data management process and COMMAND server. Local personal data management process concerns with running user interface, controlling the personal local information and invoking remote method from remote server via the COMMAND server.

The rest of this paper is structured as follows. The Distributed Object Oriented Architecture (DOOA) is expressed in section 2. In section 3, Remote Method Invocation (RMI) is presented. RMI based Personal Data Management System (PDMS) Architecture and components of Personal Data Management System (PDMS) are described in section 4. Finally conclusion is presented in section 5.

2. Distributed object-oriented architect-ure (DOOA)

Distributed Object-Oriented Architecture (DOOA) is rooted in distributed object-orientation and clients can consume services by invoking discreet object/method calls directly. It can be defined as a way of designing and implementing enterprise applications that deals with the intercommunication of loosely coupled, coarse grained (business level), reusable artifacts (objects/services). Determining how to invoke these objects should be through the object interfaces.

Distributed object-oriented architecture in which new services and in some cases, new types of object can be instantiated and immediately be made available for invocation. It includes peer to peer (P2P) architecture, client-server architecture, three-tiered architecture, cluster architecture and grid architecture. Advantages of the DOOA are

- Component reuse

- Less time & resources
- Less complexity
- Less storage space

Possible technologies to object invocation for DOOA are sockets, remote procedure call, remote method invocation, common object request broker architecture, distributed component object model.

3. Remote method invocation (RMI)

Remote method invocation (RMI), is a mechanism for invoking an object's methods, even though the object is executing on a foreign Java Virtual Machine (JVM). RMI is similar to remote procedure calls (RPCs), but has an added advantage - method signatures can contain Java objects as well as primitive data types. Even objects that a foreign JVM has never encountered before can be used, so new tasks and methods can be passed across a network RMI provides the user with facilities to make method calls remotely. RMI is transparent to the local modules of the application. Thus, there would be no difference between making a call to a local method and making a remote method call to a method on a remote machine using RMI. The user is provided with the same interface by the underlying layer [5].

3.1. RMI framework

The RMI framework in Java allows distributed application components to communicate via remote object invocations. In particular, a client running at one node can access a remote service by invoking a method of the object that implements the service. Thus, the RMI framework enables applications to exploit distributed object technology rather than low level message passing (e.g., sockets) to meet their communication needs [2].

All objects that can be invoked remotely must implement the interface Remote. This interface is just a tag that is used to distinguish remote objects from normal objects. Remote objects implement one or more interfaces and only through these interfaces they are visible to the outside world. The rmic tool is used to generate the skeleton and stub classes for a given remote object interface. For a given remote object impl, the stub, impl_Stub, the skeleton, impl_Skel, have the same set of methods that are defined in the interface of impl. Impl provides the actual implementations of the methods defined in its interface [3].

A high level architecture of the RMI framework is shown in Figure 1[2].

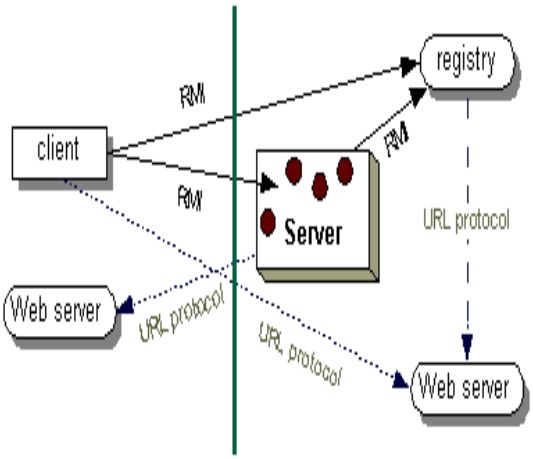

Figure 1. RMI framework

3.2. Object invocation in RMI

The object that can receive remote invocations is called remote objects. In Figure: 2, the objects B and F are remote objects. All objects can receive local invocations, although they can receive them only from other objects that hold references to them. For example, object C must have a reference to object E so that it can invoke one of its methods.

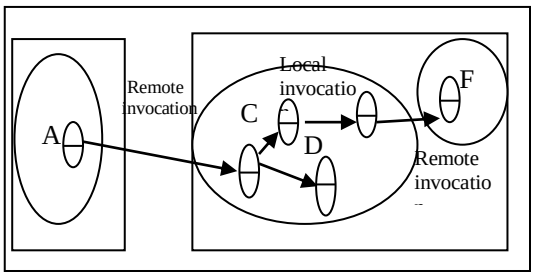

Figure 2. Remote and local method invocation

3.3. Basic patterns in RMI

Java RMI is comprised of three layers that support the interface as shown in Figure 4. The first layer is the Stub/Skeleton Layer. This layer is responsible for managing the remote object interface between the client and server.

The second layer is the remote Reference Layer (RRL). This layer is responsible for managing the “liveliness” of the remote objects. It also manages the communication between the client/server and virtual machines,(e.g., threading, garbage collection, etc.) for remote objects.

The third layer is the transport layer. This is the actual network/communication layer that is used to

send the information between the client and server over the wire. It is currently TCP/IP based. RPC is a UDP-based protocol which is fast but is stateless and can lose packets. TCP is a higher-level protocol that manages state and error correction automatically, but is correspondingly slower than UDP [4]. The architecture of RMI is shown in Figure 4 [1].

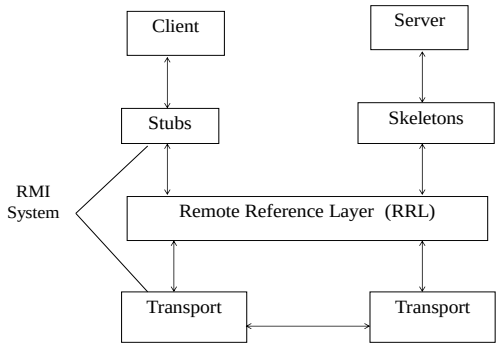

Figure 3. RMI architecture

4. RMI based PDMS architecture

The PDMS system concerns the management of personal data in the three-tiered distributed environment. This system involves distributed units located distant places and the number of units is increasing day by day. Here, the efficient management of these units is really needed. The three-tiered architecture of RMI is shown in Figure 5.

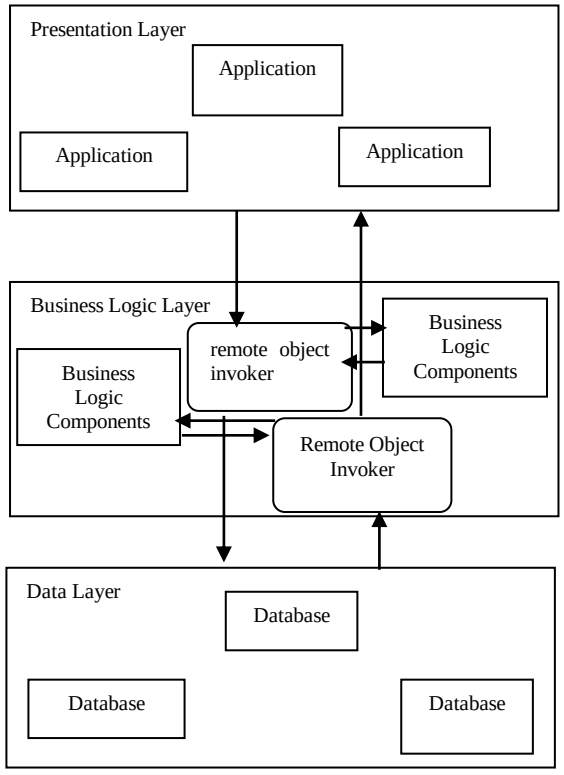

Figure 4. Three-tiered architecture of RMI

It includes three layers:
Presentation layer- includes multiple distributed applications which access the remote distributed processes.
Business logic layer- includes business logic components (e.g., middle manager/remote process invoker) that manage the available access to the remote process.
Data layer- includes heterogeneous processes of the distributed databases.
Using this architecture, remote method invocation is carried out as follow:

- RMI Client- looks up and fetches the remote interface from the remote RMI Server. The methods in the remote reference are available to the client.
- Command Server- use global information of personal data and units to analyze authorized process and invoke the required method from the corresponding remote RMI Server.
- RMI Server-implement the remote interfaces and registers them to the rmiregistry. By running rmiregistry, create stubs and skeletons that provide the network connection operations and idle remote object as another local object on local machine**.**

4.1. Architecture of PDMS

The architecture of the system is scoped in the three-tiered distributed architecture. In this system, there are two kinds of entities: Local Personal Data Management Process and Command server.

Local Personal Data Management Process (PDMS) is responsible for processing transactions. Both local client and remote server have this process and they maintain the local information of its personal data. When local client requests local transaction process, local PDMS processes local transaction and updates the information of local person. When local client requests remote transaction process, local PDMS requests to the COMMAND server by the use of RMI and returns the response to the local client.

The COMMAND server that is the essential element in distributed three tiered architectural system maintains the repository of global information of local persons and processes of each unit. When local PDMS makes request, it accepts the request, and invokes to the remote server by the use of RMI and then response to the local PDMS.

When Remote server receives the request of COMMAND server, its PDMS processes the requested transaction and updates the information of its personal data and then returns the response to the COMMAND server.

4.2. System design of PDMS

In this system, local client's PDMS concerns with running user interface, managing the personal local information and invoking remote method from remote server (i.e. it is also Local process itself) by using RMI.

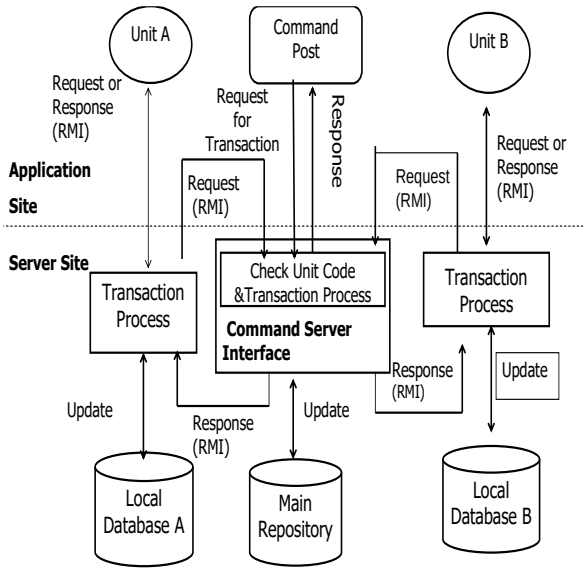

Figure 6. System design of PDMS

Local client(unit A) –When the user chooses Transaction process, local personal profile manager accesses the name of specific person to process the transaction and searches the name in the local database. If the name is in the local database, local Transaction process executes local process. If the name is in the remote database, local Transaction process requests to the COMMAND by the use of RMI. If local client receives the response of COMMAND, it updates its database.

COMMAND-When the COMMAND receives the request of local client; it searches the code of the unit, checks the relevant information for the local client from the repository and inserts a new record of transaction in its repository. Then it requests the remote Transaction process by using RMI. If it receives the response of remote server, it returns to the local client.

Remote server (unit B)- When the remote server receives the COMMAND'S request, it executes its local Transfer process and updates its database.

By setting the COMMAND site in this system, any client can quick access the global information of any person at any unit. This result also facilitates the transparency of distributed systems. The distributed remote method invocation allows this simplification against traditional approaches that are forced confusing as well as tedious and error-prone to execute code on other machine across a network. The input requests from local unit with relevant information are accessed by the COMMAND. The requests are possible either local or remote object/method invocations.

4.3. Components of PDMS

The main components of PDMS are:

- Local client component
- Command server component
- Remote server component

During transaction processes of PDMS (such as Transfer, Promotion, Demotion and Enquiry) all these components perform personal data management
process in parallel local site (for example unit server 1) and Remote Site (for example unit server 2) contains their related Personal Data Manager table. On the other hand, command site includes Letter Manager table and Unit Data table.

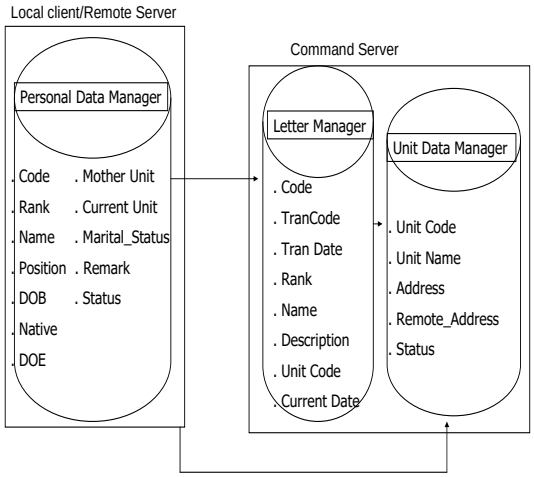

Figure 7. Database design of PDMS

At the end of transaction process, the user can see the information in the corresponding unit's server and command's server log window as shown in figure 8 and 9.

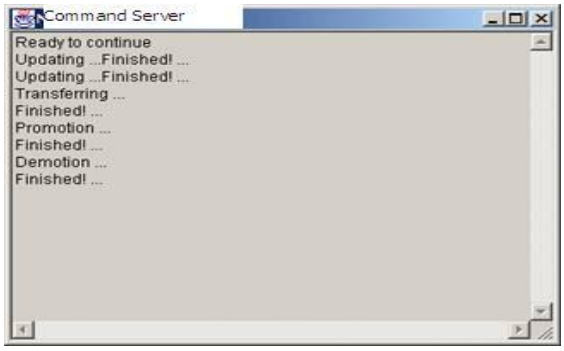

Figure 8. Command's server log window

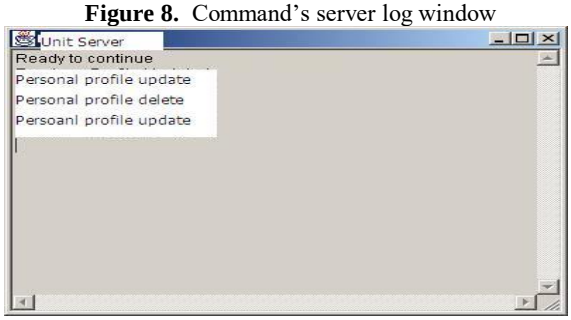

Figure 9. Unit's server log window

5. Conclusion

The system is implemented to use the infantry personal data management process in distributed areas. The theory of Java's RMI (Remote Method Invocation) is applied for the communication of distributed system in this system. It is the effective communication technique between JVMs (JavaVirtual Machine). It is suitable for this kind of application, PDMS on distributed architecture.

This application is useful to the developers who need to know only one programming language (Java) to develop both client and server process without having to worry about the underlined operating system and hardware because Java/RMI is platform independent. User can compose the processes by invoking the methods from the remote addressed space transparently.

Therefore, the PDMS system provides the transparency of the possibility of concurrent access to remote objects.

6. References

[1] A. Wllrath, R. Riggs, J. Waldo "Distributed Object Model for the Java System" in the *Proc* of the USENIX 1996 Conference on Object-Oriented Technnlogies, Toronto, Ontario, Canada, June 1996

[2] B. Tarr "Designs Patterns In Java"

[3] B.Topol, D.Walther, E. Bommaiab "Efficient Implementation of Java Remote Method Invocation (RMI)" in *Proc*. the 4th USENIX Conference on Object-Oriented Technologies and Systems (COOTS) , Santafe, New Mexico, Georgia Institute of Technology, April 27-30,1998

[4] G.Coulouris, J Dollimore, T. Kinberg "*Distributed Systsems Concepts and Designs"*, 0201-61918-0, Addison Wesley 2001, 3rd edition

[5] Liu, Ni "*Solution Distributed Computing Problem using Remote Method Invocation(RMI)"* Operating system (5024),229-87-3341, Dec 3, 2000