Applying Remote Method Invocation in Mesh-based Searching Model

Aung Aung
Computer University (Mandalay)
bayintnawn@gmail.com

Abstract

Distributed Object-oriented platforms have become important components for parallel and distributed computing and service frameworks. Among distributed objects oriented software, Remote Method Invocation (RMI) is one of the key methods for performing parallel and distributed computing in Java environments. The system is concerned mainly with the software aspects of searching remote objects on a cluster of servers that are connected in mesh-based or two dimensional model. In each server, remote objects (e-books) are stored. The communications among these servers are achieved through the Remote Method Invocation (RMI).In this system, mesh-based searching model is used to support e-books (remote objects) through the two stages: Unfolding and folding stages. Unfolding stage is the keywords dispatching to each servers to search in their related databases. Folding stage is the results collecting phase from each server to return to the client. When the user types keywords to search remote objects (e-books), these keywords are used to compare with the book title in each server's database. If keywords are found, the results together with book-related data are returned to the server1 connected in mesh-based model.Server1 finally return e-books related data from other servers together with its database results to the client.

1. Introduction

Java Remote Method Invocation ,which is distribution middleware that enables developers to create distributed Java to Java applications, in which the methods of Java objects can be invoked from other JVMs,possibly on different hosts[7].

Java RMI extends the Java object model to provide support for distributed objects in the Java language. In particular, it allows objects to invoke methods in remote objects using the same syntax as for local invocations. Objects are considered remote if they reside in a different Java Virtual Machine (JVM) .Waldo et al [3] say that the difference between local and remote object should be expressed at the remote interface. The remote object may be in a different JVM on the same computer or on a remote host connected by a network. Java RMI supports a new distributed application in the form of object-based client-server model [6].

Java RMI system assumes that the client and the server are both Java classes running in a Java Virtual Machine (JVM), which makes the network a homogeneous collection of (virtual) machines. The RMI system takes homogeneity one step further that all objects constituting the distributed system written in Java [3]. With this single-language assumption, the RMI system does not need a language-neutral Interface Definition Language (IDL). RMI simply uses the Java interface construction to declare remotely accessible interfaces. Remote objects are defined through a remote interface, which can be exported to remote clients to abstractly define communicate.

Currently, there is a growing interest in using remote method invocation as part of the solution to implement more flexible distributed applications. Java RMI is an effective choice for developing distributed applications for several reasons, including improving in flexibility. Thus, Java RMI is a key enabling technology for creating the distributed Java-based application range from information retrieval systems to mobile computing.

In this paper, remote service locating system to give the user some required service based on user's keywords is explored. Compared to sockets RMI offers a higher-level interface. Clients can truly make procedure calls directly to their server. [1]. One of the middleware instances, the remote method invocation (RMI) uses protocol based on messages between processes to provide its higher level abstractions such as remote invocation. The purpose of this paper is to assist the user to get services according to their keywords.

2. Related Work

In wide area networks, distributed applications must be capable of dealing with search response times and search optimization. The mechanism of search optimization and search response times in [6] allows the programmer to deal with such situations; allows the user to search file on a decentralized peer to peer system using Java RMI.

The need for search optimization is a well-known target in distributed applications. These applications

are difficult to develop. For this purpose, a system called peer to peer file search using RMI that (i) allows the user to search or invoke remote object without worrying about search timeout and hops made for a search to improve search response times, when the applications are interactive and requires low response time, efficient implementations of RMI are needed. Krishnaswamy et al [7] explored both transport level protocols as well as object caching in the RMI framework to meet the performance requirements of interactive application. This paper also discusses a prototype system that offers new transport protocols and allows objects to be cached at client nodes.

3. Background Theories

3.1 Remote Method Invocation (RMI)

Remote Method Invocation lets java objects on different host communicate with each other in a way that is similar to how objects running in the same Virtual machine communication with each other: by calling methods in objects. A remote object lives on a server. Each remote object implements a remote interface that specifies which of its methods can be invoked by clients. Client invokes local methods [3]. Remote method invocation uses a standard mechanism for communicating with remote objects: stubs and skeletons. A stub for a remote object acts as a client's local representative or proxy for the remote object. The caller invokes a method on the local stub which is responsible for carrying out the methods call on the remote object. In RMI, there are three main layers: the first is stub for a remote object implements the same set of remote interface that a remote object implements. When a stub's method is invoked, it does the following that is shown in Figure 1.

-Initiate a connection with the remote Java Virtual Machine (JVM) containing the remote object.
-marshals (write and transmit) the parameters to the remote JVM
-waits for the result of the method invocation
-unmarshals (read) the return value or exception returned, and
-returns the value to the caller.

The stub hides the serialization of parameters and the network level communication in order to present a simple invocation mechanism to the caller.

In the remote JVM, each remote object may have a corresponding skeleton. The skeleton is responsible for dispatching the call to the actual remote object implementation. When a skeleton receive an incoming method invocation it does the following

-unmarshals (read) the parameters for the remote method
-invokes the method on the actual remote object implementation and
- Marshals (writes and transmits) the result (return value or exception) to the caller [5].

The next layers are remote reference and transport layers. Remote reference layer is responsible for translating between local and remote object reference and for creating remote object reference. The third layer is transport layer that is responsible for setting up the connections to the remote address space (virtual machine).This layer monitor the "live ness" between the two address spaces.

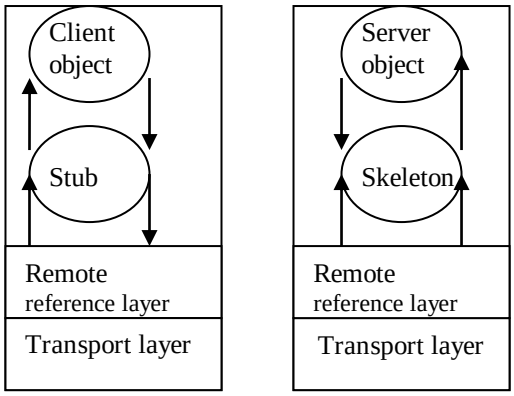

Figure 1: RMI layers.

3.2 Mesh –based Searching Model

In general Single Instruction Multiple Data are thought of (and constructed) as fine-grained machines. Where all processors are operate in a lockstep fashion on the contents of their own small local memory.

Mesh-based searching model has recently emerged as a way of workload share provision to support users based on their keywords. In general, there are other searching model-tree searches in which user submitted keywords are searched through the three stages nodes, which are parent node, intermediate node and leaves nodes. Mesh-based model's behavior is superior to that of tree model under the linear propagation time assumption. Propagation time is the time required for a signal or wave to travel from one point of a transmission medium to other receiving medium. Mesh-based searching model in Single Instruction Multiple Data typically consists of n processors, a control unit, and an interconnection function. The control unit stores the program and broadcasts the instructions to all processors simultaneously. Each processor is connected via a unit-time bidirectional communication link to each of its neighbors. Unit

time is generally defined to be the time necessary for each processor to execute some fixed number of arithmetic and Boolean operations on the contents of its local memory, as well as to send and receive a piece of data from each of its neighbors.

4. System Architecture

In this section the main components of system architecture for e-books (remote objects) searching model on servers that are connected in mesh-based model is discussed.

4.1 Deployment of RMI in mesh-based searching model

Distributed systems require that computations running in different address spaces, potentially on different hosts, be able to communicate. For a basic communication mechanism, the Java programming language supports sockets, which are flexible and sufficient for general communication. But, sockets require the client and server to engage in applications-level protocols to encode and decode messages for exchange, and the design of such protocols is cumbersome and can be error-prone. An alternative to sockets is Remote Method Invocation (RMI) that is used to match the communication model between program-level objects residing in different address spaces.

In this system, the Remote Method Invocation as a communication model to the mesh- connected computers is applied. Mesh-based searching model has two stages: Unfolding and folding stage [8]. Unfolding stage is the keywords dispatching to other servers and folding is the results collecting from each server. Remote objects (e-books) are stored in each server's database. When the user wants to search the e-books, the user typed keywords are sent to server1 using the remote method invocation (RMI) mechanism. To retain mesh-based model, these keywords are first dispatch to other servers to search in their related databases. Each server will search these keywords by comparing with the book titles stored in their database.

- If the user typed keywords arrives server1, it will send these keywords to server2 for performing searching task in its databases. Server1 wait other database results to return to the client even if its database results were or not found.

-Now simultaneously, server1 and server2 will send the keywords to server3 and server4.

- But the keywords arriving flow to the server4 is from the server2, Server4 return its database results to server3 according to the mesh-searching model.

-Server3 sends the server4 results together with its database results to server1.

-Server2 also returns results to server1.

-Server1 will finally return all servers' results together with its database results to client. All servers' or some server database results may be not found value if the matched e-books not found as show in Figure 2.

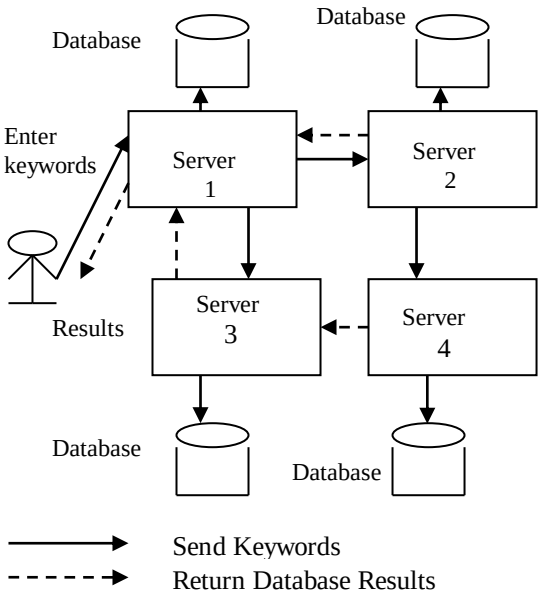

Figure2: Overall System Design using Mesh-based model

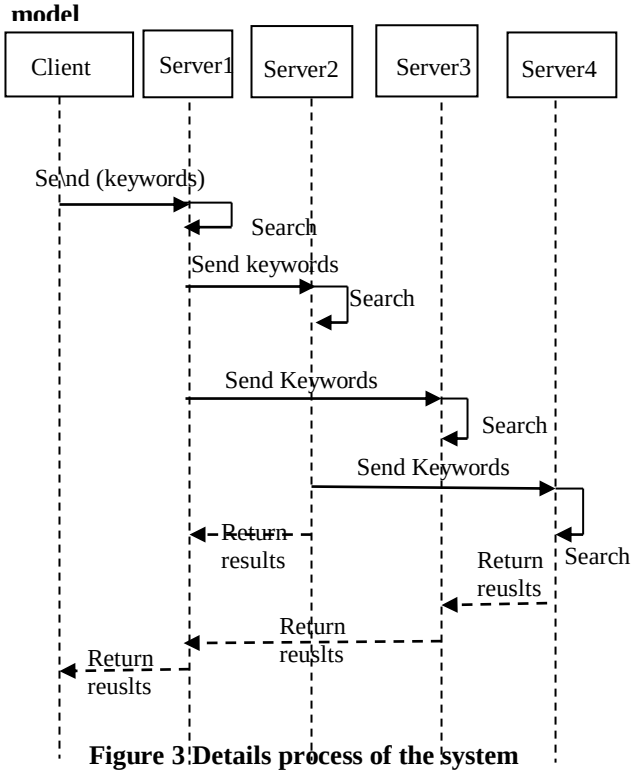

Figure 3 Details process of the system

4.1.1. Server 1

Server1 receives user keywords and then searches these by comparing book titles that are stored in its database. Then the server1 sends these keywords to

server2 and then server3 to search in their database. After that server1 wait the returned results from all servers'.Server1 reply the returned results or not found values together with its database results to the client.

4.1.2. Server 2

Server 2 receives user keywords from server 1. Server 2 searches these keywords in its database. Then it will also send these keywords simultaneously with server1 to server 3 and server4 for searching in their databases. If the searching process in server 2 is completed, it will send return value or not found value to server 1.

4.1.3. Server 3

Server 3 receive user keywords form server 1. Server 3 searches these keywords in its database.Server3 receives server4's database results. The server3 returns results from server4 together with its database results to server1.

4.1.4 Server 4

Server 4 receives user keywords from server 2. Server 4 searches these keywords in its database. Server 4 will return results to server 3 to retain the mesh model's folding stage.

4.2 System related Algorithm

The following algorithms are to use for server 1 and other three servers connected in two dimensional arrays. Let 'key' be the input (book title) from the user

(i) Server 1 (s1)

Procedure s1 (string key)
Begin
(1) receivekey () //server 1 receive keys from user
(2) do Search()
(3) SendKey () // s1 send key to s2 and s3
RESULT=s2 (key)
RESULT=s3 (key)
(4) receiveResult () //s1 receive results from other servers
(5) sendResult () //s1 send result to user
End

(ii) Server 2 (s2)

Procedure s2 (String key)
Begin
(1)receiveKey()//s2receive key from s1
(2)dosearch()
(3)sendkey()//s2 send key tos4
End

(iii) Server3 (s3)

Procedure s3 (string key)
Begin
(1)receivekey()//s3receive key from s1
(2)doSearch()
(3)sendKey()//s3send key to s1
End

(iv) Server4 (s4)

Procedure s4(String Key)
Begin
(1) receiveKey ()//s4 receive key from s2
(2) doSearch ()
(3) sendResult ()//s4 send result to s3
End

5. Conclusions

In this paper we presented an objects searching model in mesh-connected computers to provide the user to get remote objects. The theories of remote method invocation (RMI), mesh searching model are applied .Today, Java RMI technology is the most flexible and effective to create distributed applications .Mesh-connected searching model is to provide workload share among the servers. The system can give the user required objects(e-books) by comparing the book titles that are stored in each server database. Therefore mesh-based searching model helps the user to reduce their workload for searching some remote services that user requires.

6. References

[1] Dan-Adrian German "Observing the Distributed Pattern" 150 S. Woodlawn Ave., Bloomington, IN 47405

[2] E.R Harold," Java Network programming", third edition.

[3] J Waldo, "Remote Procedure Calls and Remote MethodsInvocation", http://www.cse.ttu.edu.tw/~cheng/courses/Java/Reading05RMIRPC.pdf

[4] Richard E. Schantz, Douglas C. Schmidt,"Middleware for Distributed Systems", http://cse.wustl.edu/~schmidt/PDF/middleware-chapter.pdf.

[5] S.AL SALAIMEH, "Information Technologies of the distributed application design", Leonardo Journal of Sciences, Issue 10, January-June 2007, P. 41-46.

[6] Rangarajan, Divya and Ravindranathan “Peer to Peer file Search using RMI”, Project Report.

[7]. V.Krishnaswamy, D.Walther, S .Bhola, E Bommaiah, G.Riley, B Topsoil and M.Ahamad, “Efficient Implementation of Java Remote Method Invocation (RMI) “,in Proceedings of the 4 th USENIX Conference on Object-Oriented Technologies and Systems (COOTS) Santa Fe,New Mexico ,April , 1998.

[8] G.Akl, Selim *The Design and Analysis of Parallel Algorithms*, Prentice Hall, Inc, 1989.