PERFORMANCE EVALUATION ISSUES OF QUERY PROCESSING THROUGH CLIENT SERVER ARCHITECTURE

Mr. Manish Srivastava
Asst. Professor, ABES Engg. College
manishzee33@gmail.com

Mr. Amit Sinha
Sr. Asst. Proffessor, ABES Engg. College
Sinha_mca@rediffmail.com

Mr Kundan Kumar Chandan
Sr. Asst. Proffessor, AIT Engg. College
kundankrchandan@gmail.com

ABSTRACT

The development of distributed systems has been affected by the need to accommodate an increasing degree of flexibility, adaptability, and autonomy. The Mobile Agent technology is important technology and an alternative to construct a smart generation of highly distributed systems and for this quantitative measurements are performed to compare Java RMI and mobile agents technique In this work, we are investigating the performance aspect of request and response time of a query which is compressed by a definite ratio by any user on client server technology. So presently we are comparing performance evaluation of query processing model of Remote Method Invocation by means of an analytical approach that is by using client server technology. We demonstrate the effectiveness of query processing for dynamic code deployment and remote data processing by reducing total latency and at the same time producing minimum network traffic.

KEYWORDS

Client server, mobile agent, Java RMI.

INTRODUCTION

A mobile agent consist of two entities, these are software and data. These entities can move completely from one host to another host

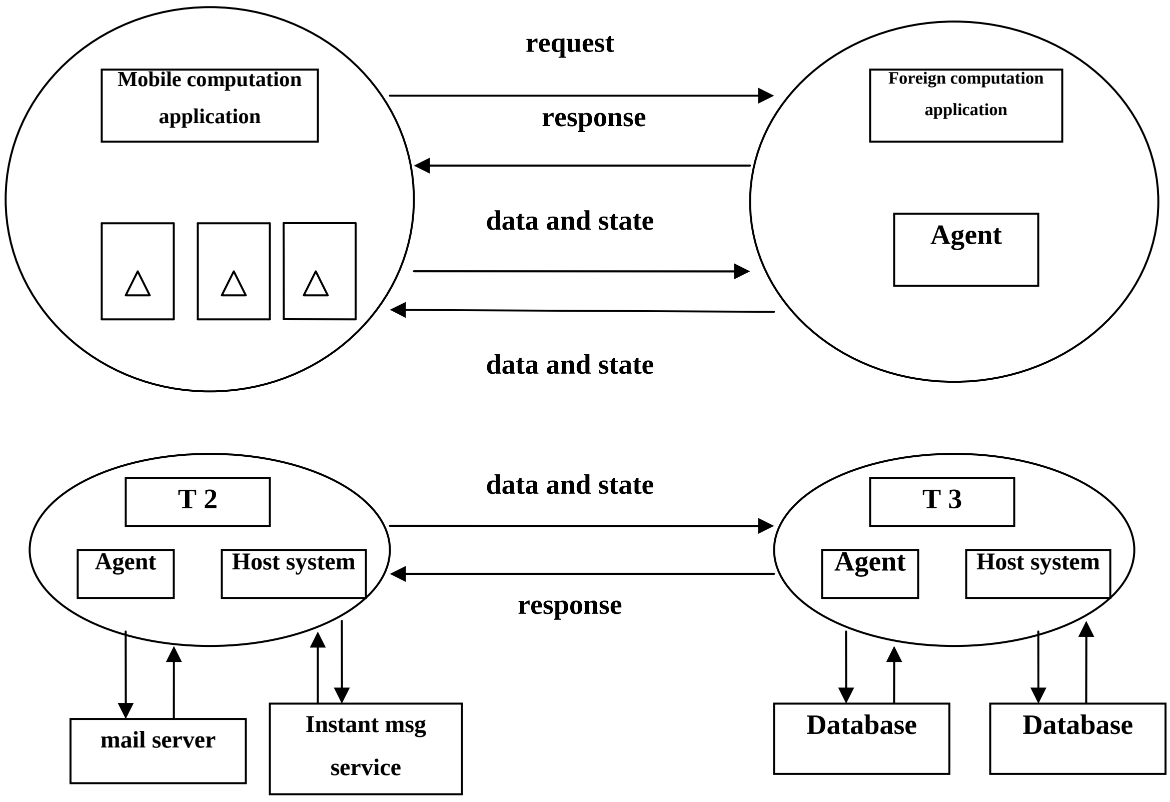


Fig. 1: Mobile agent based architecture

automatically. It may also be defined as an independent software program that may be used to perform an user's task [1]. It moves from a home location to a foreign location through agent discovering. The agent enables it to get a care of addresses (COA) . The mobility of code and data can be performed upon the following advertising. In the above diagram, mobile agent based architecture communicates by processing a request in which agents move from one location T1 to another locations say T2 and T3. Whenever agent move from one location to another it save its own state and transmit the saved state to the next host in order to resume its execution from this saved state to another starting host. The mobile agent allows asynchronous running of codes on different host and reduces the computational and data requirement. **Thus it is the responsibility of mobile agent to allows the resources of its host to the other resources scarce devices.**

JAVA RMI

RMI is native to java. It is a java technology which allows applications to call remotely located object method, thus enabling to share resources and processing load across system. In other words one JVM can invoke methods belonging to an object stored in another JVM. For ex, Client A writes a scenario for some functions and can updates his services by adding new features and improving existing one. If a client B wishes to use services developed by Client A, than developer A places the new classes in a web directory and hence there after RMI can fetch the updates as when required.

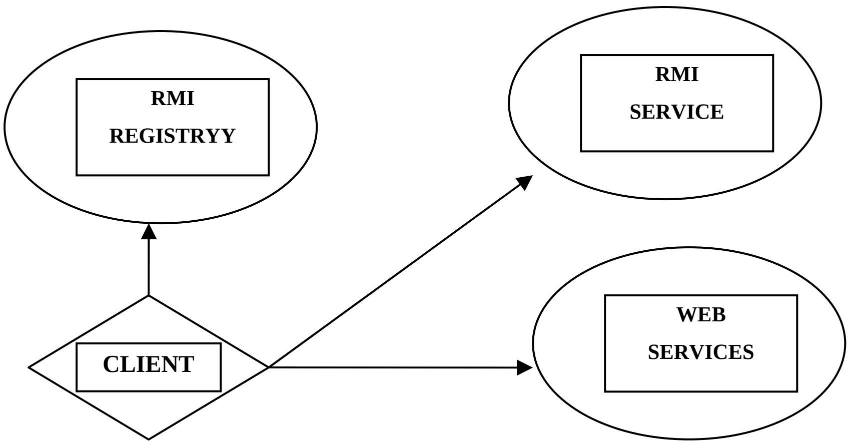


Fig. 2: Connection made with client using RMI

In the above figure, client A must contact with an RMI registry and request the name of services, thus the services are now available to it registry . Any client now only need to contact to the client A’s registry.

DEPENDANCE CONSRAINT

The rate of data traveled in medium depends upon factors like bandwidth, marshalling of data and the network factors like congestion, routing technologies. Network devices such as router, switch, and concept of bridge can be used to reduce the amount of traffic flow in the network. This paper [1] describes the quantitative measures for comparing JAVA RMI and mobile agents using communication time, code size, Data size and number of nodes . It compares both mobile agent and client server paradigm in java environment using RMI.
The main difference lies in how the code and data is being distributed in both technologies. In RMI model, client can use the objects defined on the server as if it were running on its own machine [2][7]. RMI allows the code that define the behavior and the code that implement the behavior to remain separate and to run on separate JVM [4],[6]. Mobile agent on the other hand migrates through many nodes transmitting the data code from one server to another till the data is not find as shown in fig 2.b/

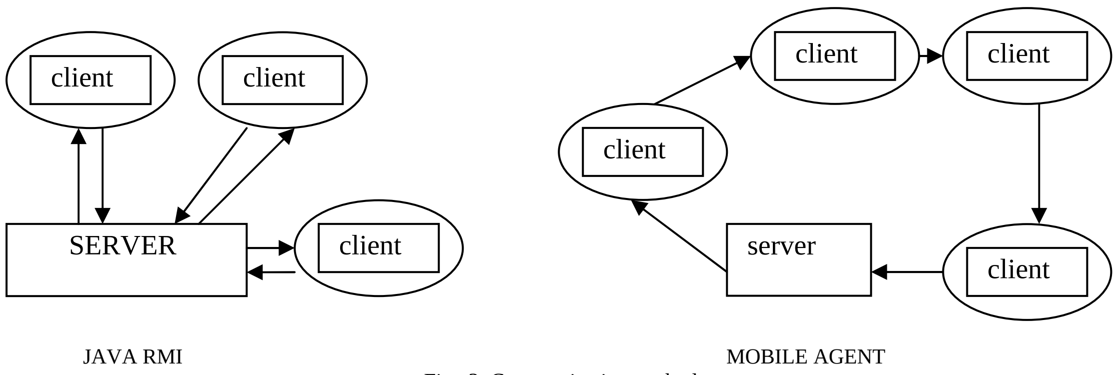


Fig. 3: Communication method

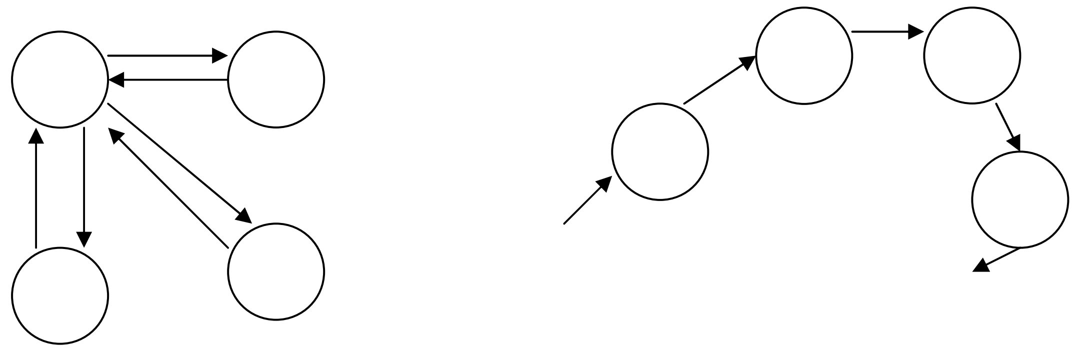
For comparing the two technologies, following parameter were used: Communication time and Data size invocation cost. the time interval between when a query is sent from the source and when response is found at the host is called communication time. In RMI model a request is send to server, if the object is not found on server than a report is send back to client after which the client contact with the other servers until the object is found[7].

$$T_{comm}^{RMI} = t_{req} + t_{res} \quad [1]$$

Where T_{comm} = Communication time, t_{req} = request time and t_{res}= response time

For n nodes,

$$T_{comm}^{RMI} = n(t_{req} + t_{res}) \quad [2]$$



a) Data migration in client server b) Data migration in mobile agent

These parameter are analyzed as follows:-

client server	Mobile agent
$T_{comm}^{RMI} = n(t_{req} + t_{res})$	$T_{comm}^{MA} = n(t_{r} + t_{y})$
$C_{rmi} = 2n\bar{\alpha} + n\dot{\varphi} + £$	$C_{ma} = (n+1)\bar{\alpha} + n\sigma + C$

This paper [2] describes the effectiveness of mobile agent for data processing by reducing the total latency and at the same time producing minimum network traffic and it also prove that mobile agent have a better fault tolerance when compared with RMI based applications. This paper describe how the mobile agent carries the data using strategies like a) Push to all next b) Push all to all and c) Push all units. It describes all the main component of mobile agent: code segment, data state and execution state which convert data mobility into strong migration. On the other hand java does not provide strong migration as it does not provide sufficient mechanisms for capturing the execution state of the agent [5].
In client server architecture using RMI, each client send the request to a server and than server responses back to the client while Mobile Agent compresses the data by a ratio (σ) . Mobile agent consists of code Bc , data state Bd and execution state Bs .
The network load can be compared by analyzing these approaches. In RMI, the client sends a request to server. The server searches the data locally and replies back to client by the found data or returns a NOT-FOUND reply. So network load is given as :

$$B_{cs} = p_1(B_{req} + B_{res}) + p_2(1-p_1)(2B_{req} + B_{res} + B_{NF}) + \ldots . p(1-p)(1-p)..(1-p)(nB_{req} + B_{res} + (n-1)B_{NF})$$

$$B_{cs} = p(1-p)(nB_{req} + B_{res} + (n-1)B_{NF}) \quad [3]$$

In case of mobile agent, the mobile **agent migrates** to new location with push to all next migration strategy . So in this kind of situation network load in this case will be given as :

$$\mathbf{B_{ma}} = p_1\,(B_c + 2B_s + B_{req} + (1-\sigma)B_{res}) + p_2\,(1-p_1)\,(2B_c + 3B_s + 2B_{req} + (1-\sigma)B_{res}) + \text{-----} + p_n\,(1-p_1)\,(1-p_2)\,(1-p_3)\ldots(1-p_{n-1})\,(nB_c + (n+1)B_s + nB_{req} + (1-\sigma)B_{res}) \quad [4]$$

RESPONSE TIME

In client server technology, the response time is the time taken by the server to find the data. Let time tb01 be the processing time on server 1 and BNF is for when data is not found on server. Thus the response time will be:

$$t_{CS} = p_1\,((B_{req} + B_{res})t_{01} + t_p) + p_2\,(1-p_1)\,((B_{req} + B_{NF})t_{01} + (B_{req} + B_{res})t_{02} + 2t_p) + \ldots + p_n\,(1-p_1)(1-p_2)\ldots(1-p_{n-1})\,((B_{req} + B_{NF})(t_{01} + t_{02} + \ldots + t_{n-1}) + (B_{req} + B_{res})t_{0n} + n\,t_p) \quad [5]$$

In case of Mobile agent, the response time is given as

$$t_{MA} = p_1\,(B_{ma}\,(t_{B01} + t_m) + (B_s + B_{req})(t_{B01} + t_m) + t_s + t_p) + p_2\,(1-p_1)(B_{ma}\,(t_{B01} + t_m) + (B_s + (1-\sigma)B_{req})(t_{B02} + t_m) + 2t_s + 2t_p) \quad [6]$$

The parameter are analyzed as follows

$\mathbf{t_{CS} = p_1\,((\sigma B_{req} + \sigma B_{res})t_{01} + t_p) + p_2\,(1-p_1)\,((\sigma B_{req} + B_{NF})t_{01} + (\sigma B_{req} + \sigma B_{res})t_{02} + 2t_p) + \ldots + p_n\,(1-p_1)(1-p_2)\ldots(1-p_{n-1})\,((\sigma B_{req} + B_{NF})(t_{01} + t_{02} + \ldots + t_{n-1}) + (\sigma B_{req} + \sigma B_{res})\,t_{0n} + n\,t_p)}$	$\mathbf{t_{MA} = \Sigma\, p_i\, \Pi(1-p_j)(B_{ma}\,(\Sigma\, t_{B0n} + t_m) + (B_s + (1-\sigma)B_{req})(t_{B0i} + t_m) + i\,t_s + i\,t_p)}$

This paper [3] focuses on querying the data which a mobile user needed using the aglet []

Based upon the study of the following paper, we suggest a method in order to reduce the processing time in the client server technology. In client server technology, processing time is the time when a client request a query to server and server response back to client.

In order to minimize the processing time, we compress the data by a factor σ at the client side which reduce the size of query and hence bandwidth can be preserved.

When a client request for the query to a particular server than server response back to client if query is not found, so client contact other server till the query is not found.

$$\mathbf{T_{process} = (T_{req} + T_{res})}\text{, \textbf{where}}$$

$\mathbf{T_{process}}$ = It is the total time taken in responding to a query when a corresponding request is made.
$\mathbf{T_{req}}$ = It is the time when a client or user request a query.
$\mathbf{T_{res}}$ = It is the time by the server to respond to a particular query which is requested by client.

$$B_{cs} = p_1(\sigma B_{req} + \sigma B_{res}) + p_2\,(1-p_1)\,(2\sigma B_{req} + \sigma B_{res} + B_{NF}) + \ldots\ldots + p_n(1-p_{n-1})\ldots(1-p_2)\,(1-p_1)(n\,\sigma B_{req} + \sigma B_{res} + (n-1)B_{NF}) \quad [7]$$

Where, $p_1, p_2, p_3 \ldots\ldots p_n$ are probability that response would be sent to client in first attempt then probability will be p_1 if not then failed message is received i.e. if client respond to n^{th} request then probability will be p_n.

σ = It is the compressed ratio according to which data is compressed.
n = the no. of times request is sent to the server
in order to get a response.

RESPONSE TIME

In client server technology, the response time is the time taken by the server to find the data. Let time tb01 be the processing time on server 1 and BNF is for when data is not found on server. Thus the response time will be:

$$t_{CS} = p_1 ((\sigma B_{req} + \sigma B_{res})t_{01} + t_p) + p_2 (1-p_1) ((\sigma B_{req} + B_{NF})t_{01} + (\sigma B_{req} + \sigma B_{res})t_{02} + 2 t_p) + ... + p_n (1-p_1)(1-p_2).. (1-p_{n-1}) ((\sigma B_{req} + B_{NF})(t_{01} + t_{02} + ...+ t_{n-1}) + (\sigma B_{req} + \sigma B_{res}) t_{0n} + n t_p)$$ [8]

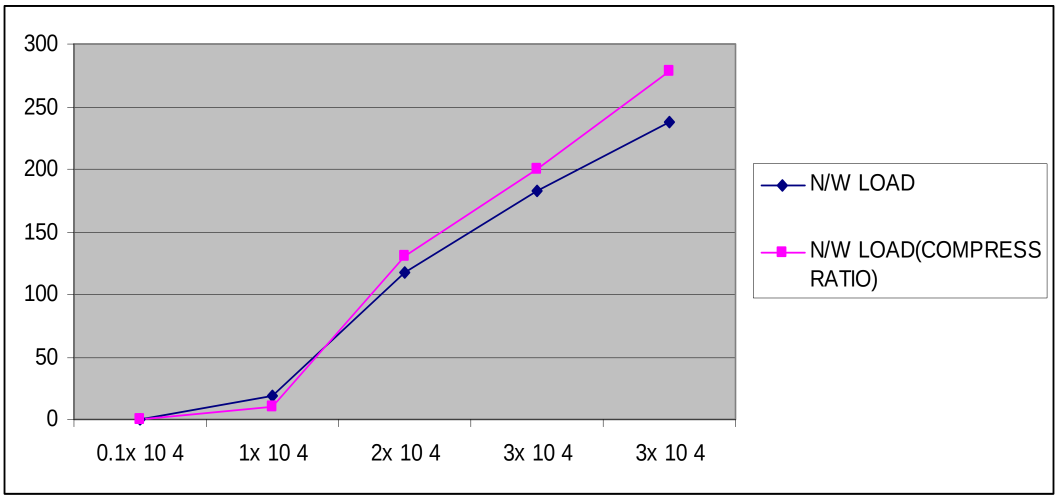
Now, we will compare the above two methods to improve the response time i.e. the time taken by server to respond to a request. If the time can be reduced to some extent then response can be sent at a faster rate and this will improve its efficiency. We have taken 5 files of size 0 KB, 19 KB, 118 KB, 183 KB, 238 KB before compression and after compressing the size becomes 0 KB, 10 KB, 130 KB, 180 KB, 278 KB and then calculated the time taken by server and plotted a graph and then we have compressed the file with a compressed ratio 0.6, probability p= 0.5, and then calculated the time taken in getting response.

COMPARISON OF THE ABOVE TWO METHODS

NETWORK LOAD V/S SIZE OF FILE

Table 1:

N/W LOAD	SIZE OF FILE
$1x 10^3$	**05 KB**
$1x 10^4$	**19 KB**
$2x 10^4$	**118 KB**
$3x 10^4$	**183 KB**
$4x 10^4$	**238 KB**

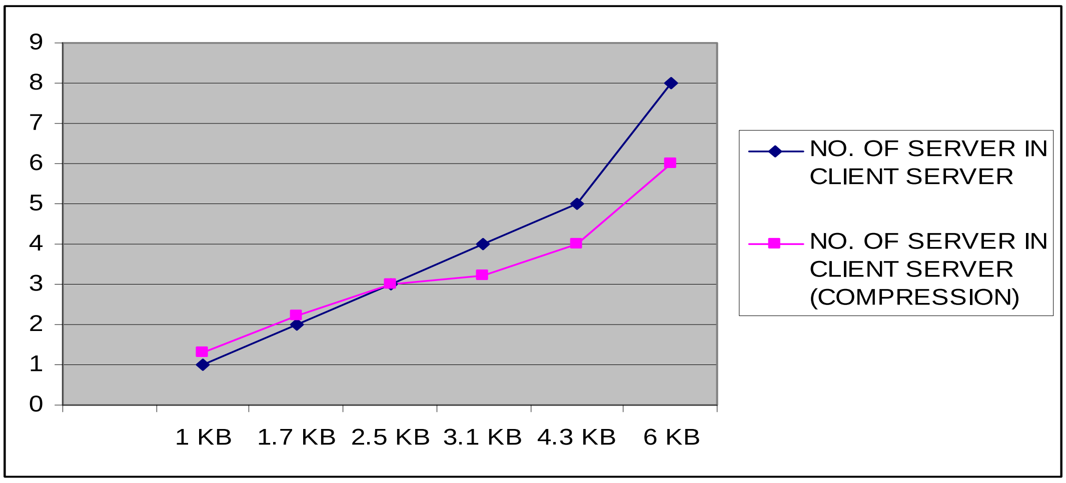


The above figure suggest that the method of compressing the size of file work well enough as the size of data increases ,the network load can be easily seen with the help of table and above diagram. Initially the network load upto the file with size 50 KB was less for the client server technology but as the size of file increases than the technique of compressing the file first before sending works well.

The network load after compressing the size of files:

Table 2:

N/W LOAD	**SIZE OF FILE**
1x 10^{3}	**05**
2x 10^{4}	**10 KB**
2x 10^{4}	**110 KB**
3x 10^{4}	**180 KB**
4x 10^{4}	**208 KB**



From the above table we conclude that the time taken in responding compressed data is less as compare to original data. This has been proved in the following manner:
we are calculating the response time of original and compressed data. the response time of original data can b calculated by tcs and that of compressed data is by tnew. Some parameters like probability pn= 0.5, t01,t02,……,t0n, Bres= 2 ms , Breq= 1 ms , BNF= 1 ms , tp= 1 ms.

We have considered the file sizes 19 KB, 130 KB, 238 KB which after compression becomes 10 KB, 110 KB, 208 KB respectively.
If response is received from 3rd server then the response time for 19 KB file i.e. before compression is tcs= 8.375 ms and the response time for 19 KB file after compression is σ = .526 so tnew = 5.6625 ms.
If response is received from 2rd server then the response time for 130 KB file i.e. before compression is tcs= 6.5 ms and the response time for 110 KB file after compression is σ = .846 so tnew = 5.73 ms.
If response is received from 2rd server then the response time for 238 KB file i.e. before compression is tcs= 6.5 ms and the response time for 19 KB file after compression is σ = .873 so tnew = 5.556.
So this clearly shows that the response time for compressed file much lesser than the original file and so is our purpose.

REFERENCES:

[1] Aderounmun, G.A. (2004). "Performance comparison of remote procedure calling and mobile agent approach to control and data transfer in distributed computing environment". Journal of Network and Computer Applications, I I3 - 129.
[2] Agent Communication Language, FIPA 2002 Specification, http://www.fipa.org
[3] Shi Youqun, ”The Research and Implementation Research of Mobile Agent Technology in Commercial Data Query System”.
[4] Erich Gamma, R. Helm, R. Johnson, and J. Vlissides. *Design Patterns: Elements of reusable object-oriented software*. Addison-Wesley, 1994.
[5] Giovani Vigna,”Mobile Agents: Ten Reason For Failure.
[6] Theodore Kotsilieris, George T. Karetsos “A Mobile agent enabled wireless sensor network for river water monitoring”.The Fourrth International Conference on Wireless and Mobile Communication.
[7] Balazs Goldschmidt, Zoltan Laszlo, Mobile Agents in a Distributed Heterogeneous Database System.
[8] http://en.wikipedia.org/wiki/Mobile_agent
[9] Joseph J. Martinka , Requirements for Client / Server Performance Modeling.
[10] Özgür Ulusoy, Geneva Belford, *A Simulation Model for Distributed Real-Time Database Systems*, Proc. of 25th Annual Simulation Symposium, Orlando, Florida, April 6-9, 1992, pp 232-240.
[11] Vidar Vetland, *Measurement-Based Composite Computational Work Modelling of Software*, PhD Thesis, Norwegian Institute of Technology, The University of Trondhem, 1993.