Comparative Study of Distributed Computing Paradigms

Harvendra Kumar[1] and A. K. Verma[2]

Abstract - The mobile agent paradigm has revolutionized the distributed computing environment. There are different paradigms used in distributed computing, such as - client-server paradigm, remote procedure paradigm and mobile agent paradigm. The client-server is based upon the concept of a server, which serves the various request of the clients and in remote procedure call approach, a machine can connect to another machine and retrieve the information remotely. The mobile agent technology is built upon the advancement in computing and communication technology over the wired and wireless networks. Mobile agents are the software programs that can migrate from one machine to another machine in a homogeneous as well as heterogeneous environment. It can migrate in connected as well disconnected network. On each machine, the agent interacts with stationary service agents and other resources to accomplish its task. Mobile agents are particularly attractive in distributed information retrieval applications. By moving to the location of an information resource, the agent can search the resource locally, eliminating the transfer of intermediate results across the network, by this property, mobile agent reduce the end-to- end latency. In this paper, we try to point out the benefits and limitations of these paradigms.

Index Terms - Mobile agents, Client-Server, Remote Evaluation.

1. INTRODUCTION

A mobile agent is a software program that can migrate during execution from one machine to another machine in a homogeneous as well as heterogeneous network. In other words, we can say that an agent can suspend its execution, migrate to another machine, and then resume execution on the new machine from the same point at which it left off.

Mobile agent are the platform dependent, so platform should be needed on each machine, the agent interacts with stationary agents and other resources to accomplish its task. There are two alternative approaches [1] to retrieve the data – the code to data approach and the data to code approach. The mobile agent paradigm performs better if the code size is small enough; this model is being extended to support different migration strategies resulting in less network traffic and better response time. Mobile agents are not always better than client-server calls. Mobile agent is only beneficial, if the space overhead of the mobile agent code is not too large or if the wireless link connecting the mobile user to the fixed servers.

[1]*M.E. Software Engg., Thapar University, Patiala*
[2]*Faculty, Computer Science and Engg. Deptt., Thapar University, Patiala*
E-Mail: [1]*harvendra.patel81@gmail.com and*
[2]*akverma@thapar.edu*

In this paper, we compare mobile agents with classical client-server techniques and other mobile-code systems. The implementation of mobile-agent is easy than traditional client-server implementation. But one question arises, which distributive computing paradigm is better and why? So, let us consider these paradigms one-by-one.

2. CLIENT - SERVER (CS) PARADIGM

The examples of traditional client-server middleware like CORBA, RMI and DCOM.

In a classical CS paradigm, processing of the data mainly takes place in the host client. In fact, the job of the server is limited. Server executes only some basic procedures for the data retrieval and storage. Before being sent to the client, data only undergo a soft initial filtering.

The host server behaves as a simple remote storage system. Together with all of the other servers and the interconnection network, makes the whole system to form a "big repository" of information available to the different clients. The server usually makes available some procedures for handling the stored data which are designed for responding to criteria of general effectiveness. The actual data processing is therefore left to the host client, where the user can execute procedures for the kind of processing desired. This type of CS scheme is used when we want to create a very simple system from the management point of view, or structures with a high level of security. Such a paradigm depicted in Figure 1. An advantage of this architecture is the possibility of controlling the type and the ways of access to the data stored in the server. Consequently, security in the CS architecture is very high. Here, in this paper, we consider the following question- is the mobile agent paradigm "better" than traditional client-server paradigm? In the next section, we will try to find out why the mobile agent is better to other paradigms.

In fact, if the user has specific requests concerning the modes of data processing, and if the server does not provide for that specific type of operations, the only possibility commits in retrieving much more data than needed, and then to perform the operations of processing and selection in the client. In these cases, the server provides a huge amount of documents, in order to assure a wide basis of selection. Of course, all that causes an overload of both the server and the communication system. In fact, the amount of data exchanged may be considerable. Consequently, the host client must have its own processing capability.

3. REMOTE EVALUATION (REV) PARADIGM

Unlike the typical Client - Server, Remote Evaluation (REV) paradigm implies that server receives not only the processing requests from the client, but also the whole code needed for performing operations of selection on the data stored. The response of the server, with no additional overhead, is limited to

sending the information that can be actually used and required by the client. The REV is based on the code to data strategy therefore it better to CS paradigm.

Besides, since the user can use a customized code in the server, the data sent in output are ready for the use, and they only need negligible additional processing. From this point of view we can also think of an environment, host clients equipped with minimum processing potentialities. The initial cost is therefore higher in comparison with the CS paradigm, and is localized in the opening stage of sessions.

In fact, the code for the data processing can be of considerable size and we can easily assume that its size is higher than a simple retrieval request. Of course, this cost is counterbalanced by the reply stage (transmission of search results from the server to the client), because the amount of data passing through the network is more limited. During the stage of design, a system with REV must be created by considering more detailed aspect in comparison with the CS paradigm. In fact, the processing architecture of the different servers must be similar (or very well known), so that the code sent by the client can be easily executed on all of the hosts. From this point of view, we can think of a common platform for code execution. This also implies the need for creating protection elements that could assure a high level of security.

4. MOBILE AGENT (MA) PARADIGM

A Mobile Agent (MA) is an executable code that can move from a host to another host, according to the mobile agent itinerary, which may be static or dynamic itinerary. Basically Mobile agent consists of three components [2], code statement, data state, and execution state. Code is transferred during the migration; even some data state can also transfer. But execution state cannot transfer in the network. This way, there is a kind of suspension of the execution of the program, waiting for the subsequent resume state [5] on a remote machine. Both (Mobile Agent and Remote Evaluation paradigm) use the same strategy code to data. The system of Remote Evaluation is a more limited approach than the MA.

In fact, a code migration is present in the REV, but there is always a direct interaction between the client and the server. This means that the code sent by the client returns the data directly to the source. Besides (when this operation is done), the process is completed, so the context of execution of the program is limited to the single host. Conversely, the mobile agent system can be used for performing the research operations more effectively. In fact, the agent has the procedures for operating on the database according to the ways desired by the user, and can also make independent decisions, such as migration to other sites or returning the results obtained to the user, if they are considered sufficient. In this sense, the interaction between the user and the agent is limited to the stages of transmission and return of data. What takes place within this time limit depends only on the way the agent was designed.

By moving the code to the data (see in Figure. 3), a mobile agent can reduce the latency of individual steps, avoid network transmission of intermediate data, continue work even in the presence of network disconnections, and complete the overall task much faster than a traditional client/server solution.

We can therefore expect that the amount of data transferred in each migration tends to increase. The agent can decide to limit the data considered interesting for the user dynamically, even by discarding the data selected in the previous hosts. Agents with a maximum quota of user data, which can be moved in each migration, can therefore be designed.

A MA, shown in Figure 3 is an autonomous transportable program (or object) that can migrate under its own or host control from one node to another in a heterogeneous network. In other words, the program running at a host can suspend its execution at an arbitrary point, transfer itself to another host (or request the host to transfer it to its next destination) and resume execution from the point of suspension.

A MA migrates from one host to other host on the behalf of itineraries [3, 4]. It may be either static or dynamic. Itinerary defined by some parameters such as Agent_Id, State_Type, Time and Place.

When the agent reaches a server, it is delivered to an agent execution environment. Then, if the agent possesses necessary authentication credentials, its executable parts are started.

Mobile Agent paradigm of the distributed computing is different for other paradigms. In other

paradigms, independent processes collaborate by exchanging data over their network links. With Mobile agents, a process is transported, carrying with it the shared data as it visits individual processes on its itinerary.

4.1 LIFE CYCLE

The model of mobile agent paradigm is based on the migrating workflow [6, 7] system model. The resuming instance is the task executor in the migrating workflow system; it is a mobile agent in essence. Our workflow-oriented life cycle model consists of five life states, (creating, running, deleting, suspending, resuming) and a number of transitions (active, suspend, dispatch, resume, terminate) between these states. The workflow-oriented life cycle model of mobile agent is shown in Figure. 4.

To accomplish its task, the mobile agent can transport itself to another server in search of the needed resource/service, spawn new agents, or interact with other stationary agents. Upon completion, the mobile agent delivers the results to the sending client or to another server.

1. In the creating state, the agent is created but not activated yet.
2. In the running state, the agent is running, performing actions and solve it pursue.
3. In the deleting state, the agent is terminated;
4. In the suspending state, the agent can not run and still stay within the agent server;
5. In the resuming state, the agent is travelling between two server instances.

4.2 MOBILE AGENT LIFE STATE LOG STRUCTURE

The life cycle of MA begins at the moment when it is created. When MA migrating from one host to another host in order to achieving its goals; and the MA returns its server on which it was created. Two or more than two states, in life cycle of MA, may be occurred at the different time or place. The mobile agent life state log structure [7] can be defined in four-tuple:
Life_State_Log_Structure= (Agent_ld, State_Type, Time, Place), Where,

1. 'Agent_ld' identifies a log item belongs to which mobile agent;
2. 'State_Type' indicates the type of mobile agent life state, with
3. State_Type € STATUS={Creating, Running, Suspending, Migrating, Deleting }.
4. 'Time' indicates the time when the mobile agent (Agent_ld) came to the current State Type;
5. 'Place' identifies the agent server where the mobile agent (Agent_ld) came to the current State-Type at the specific time.

4.3 APPLICATION AREAS OF MOBILE AGENTS

Mobile agents provide effective and flexible mechanisms for structuring distributed systems. The Mobile agent paradigm can be exploited in a variety of ways, ranging from low level system administrator tasks, to middleware to user-level applications. They can be mapped directly to real life situations.
The concept of a mobile agent can be applied to the Information Retrieval Systems (IRS), Distributed File System Clinical Data analysis for medical diagnosis, Distributed Data Mining, Distributed Real-time systems, Mobile Wireless Environment, Mobile Smart Databases, Peer-to-Peer Computing, Network monitoring and management, Intrusion Detection System, Network routing, Performing location-dependent computations, Load balancing, Service customization, Wireless Sensor Networks(WSN)/ Remote Sensing, Wireless Ad hoc Network (WAHN), Manufacturing, Command & Control, Grid Computing/Cluster Computing, and Information dissemination etc.

5. COMPARISONS

Paradigms/ Attributes	**Mobile Agent**	**Remote Evaluation**	**Client-server**
Implementation	Hard	Easy	Very easy
Security	Very low	Low	Very high
Performance	high	Very high	Low
Elements a) Data b) Code **c)** Stack	 semimobile mobile mobile	static mobile static	mobile static static

Paradigms/ Attributes	**Mobile Agent**	**Remote Evaluation**	**Client-server**
Itinerary	Static/Dynamic	Both	Static
Mobility	Code to data	Code to data	Data to code
Platform	Dependent	Dependent	Indepedent
Programming code	Hard	Hard	Easy
Examples	Aglet	Aglet	CORBA

Table 1: Comparison between various Distributed Computing Paradigms

5. CONCLUSION

Here, in this paper we have discussed the three basic paradigms of distributive computing, namely: Client-Server, Remote Evaluation and Mobile Agent. CS implementations are suitable for small applications where a amount of information is retrieved from a few remote servers having low processing delays. However, most real-world applications require a large amount of information to be retrieved and significant processing at the server. MA's scale effectively as the size of data to be processed and the number of servers the data is obtained from increases.
We conclude that mobile agent paradigm is the best as other paradigms, it consume lesser resources but have the limitation on the size of the code. So, it can be used extensively in a code-to-data environment. This paradigm can be exploited in many application areas, such as data mining, weather forecasting etc.

REFERENCES

[1] A. Puliafito, S. Riccobene, M. Scarpa "*An analytical comparison of the client-server, remote evaluation and mobile agents paradigms*", Proc. of the First International Symposium on Agent Systems and Applications. Page: 278-84,1999. ISBN:0-7695-0340-3.

[2] Carzaniga, A.; Picco, G.P.; Vigna, G., *"Is Code Still Moving Around? Looking Back at a Decade of Code Mobility*", Software Engineering - Companion, 2007. 29th International Conference on Software Engineering, 20-26 May 2007, Pp 9–20, Digital Object Identifier 10.1109/ICSECOMPANION.2007.44.

[3] Daniela Rus, Devika Subramanian, *"Information Retrieval, Information Structure and Information Agents"*, ACM Computing Surveys (CSUR) archive Volume 27, Issue 4 , (December 1995) Pp: 627 – 629. ISSN:0360-0300.

[4] L. Miao and H. Qi and F. Wang, *"Self-deployable mobile sensor networks for on-demand surveillance"*, Sensors, and Command, Control, Communications, and Intelligence (C3I) Technologies for Homeland Security and Homeland

Defense IV, at SPIE Defense and Security Symposium, vol. 5778, 2005.

[5] Umar, A., *"A Comparison of Mobile Agent and Client-server Paradigms For Information Retrieval in Virtual Environments*", Proc. of First International Conference, "Next Generation Enterprises: Virtual Organizations and Mobile/pervasive Technologies", April 2000.

[6] Shinichi Motomura, Takao Kawamura, Kazunori Sugahara, *"Persistency for Java- based Mobile Agent Systems "*, Proc. Third International Conference on Internet and Web Applications and Services. Pp. 470-475, 2008, ISBN: 978-0-7695-3163-2.

[7] YANG Gong-ping, ZENG Guang-zhou, "*Mobile Agent Life State Management"*, IMACS Multi-conference on Computational Engineering in Systems Applications(CESA), October 4-6, 2006, Beijing, China.

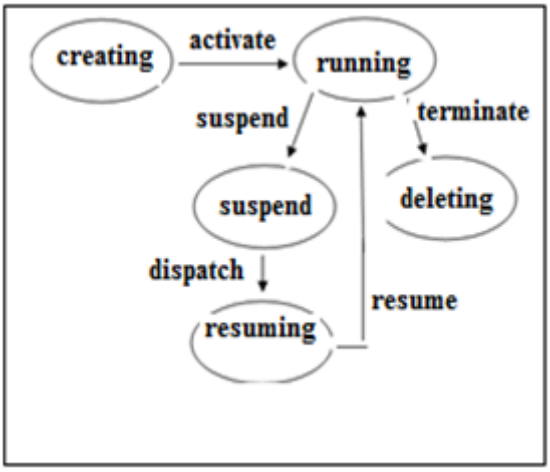

Figure 4: Life cycle of Mobile Agent

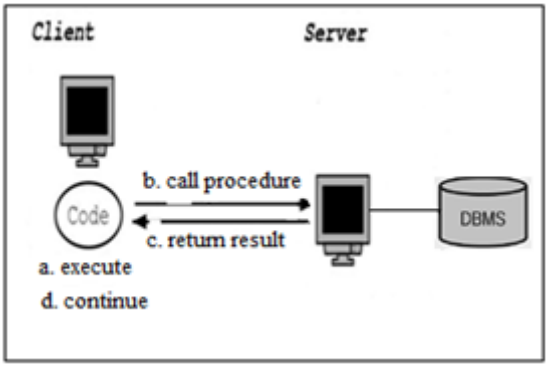

Figure 1: Client Server Architecture

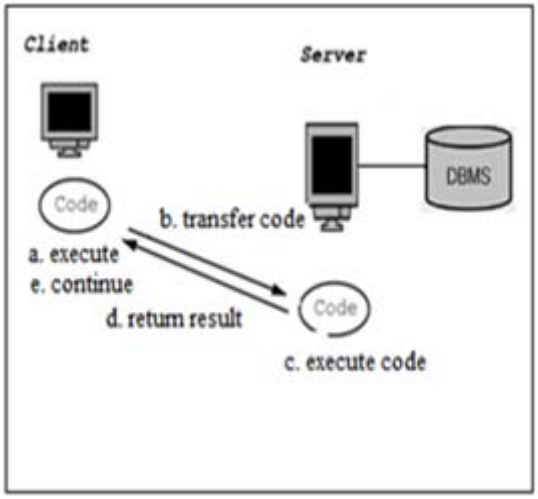

Figure 2: Remote Evaluation Architecture

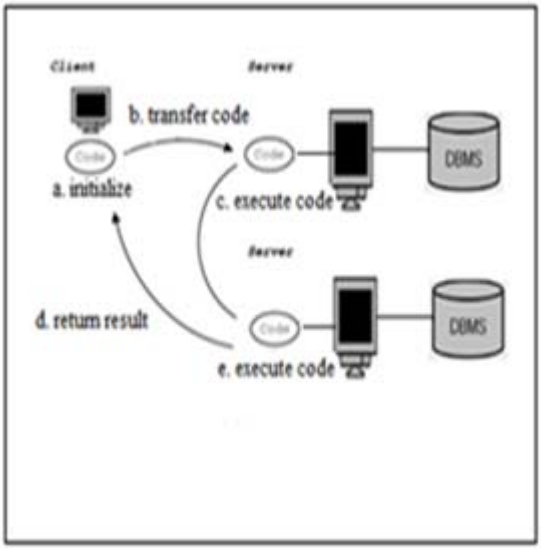

Figure 3: Mobile Agent Architecture