Distributed Information Retrieval using Mobile Agents

J. Vasudeva Rao¹, S.Ravishankar¹

¹GMRIT, Rajam
A.P., India

jvr.vasu@gmail.com

Abstract. Fast and efficient information retrieval from large and heterogeneous distributed environments is a challenging task because of heavy network traffic. The currently available client-server paradigms are not much flexible, efficient and scalable. For these environments, mobile agent technology is more promising than the conventional client-server computing. A mobile agent is an autonomous and intelligent program that moves through a network. A mobile agent migrates from a host node acting as server to required destination node(s), performs data processing locally, and sends the results back to the server, which reduces the network traffic. In this paper we discussed how mobile agents can be used for information retrieval in a distributed environment. We have also shown the differences between using multiple agents for multiple nodes and using a single agent that makes a tour of all nodes. IBM’s Java based mobile agent framework called Aglets Software Development Kit (ASDK) is used for creation and deployment of Aglets. Experimental results are presented and a detailed analysis is carried out.

1 Introduction

Information retrieval from relational databases in large and heterogeneous distributed and web systems both efficiently and rapidly is most necessary in the current scenario of fast growing world. At the same time this task is very challenging due to the heavy network traffic. Quality of Service (QoS) in terms of maximum performance and security is not really guaranteed. The currently available client-server paradigms are not much flexible, efficient and scalable in this regard. In this paper a mobile agent based framework for distributed information retrieval is discussed. Usage of mobile agents is proven to be more efficient than the client-server counterpart.

Mobile Agents are autonomous and intelligent programs that move through a network. They interact with the various nodes and their resources in this way and perform various services on behalf of the user. The agent can suspend its execution, migrate to another machine, and then resume execution on the new machine from the point at which it left off. Thus these agents are also called as software robots. Specialized servers control and coordinate the movement and activities of the mobile agents.

More specifically, a mobile agent is a process that can transport its state from one environment to another, with its data intact, and be capable of performing appropriately in the new environment. They interact with the various nodes in networks and their resources in this way [7].

Mobile agents differ from remote procedure calls, where a process invokes procedures of a remote host, mobile agents allow executable code to travel and interact with databases, file systems, information services and other
agents [3]. A comparison of Client – Server paradigm and mobile – agent paradigm is shown in Fig 1.

![Fig.1. The client-server paradigm vs the mobile agent paradigm](image)

Mobile agents have several strengths. They can interact with the resources locally and thereby eliminate the need of transferring large amounts of intermediate data. Also in case of unreliable links, a mobile agent continues to work even if the link goes down. This makes the mobile agents particularly attractive in mobile computing environments [1]. Agents save time, refining the search over time and making decisions based on past experiences. Also the mobile agents have adaptive learning feature with artificial intelligence for information retrieval

Mobile agents have different characteristics. They are autonomous, mobile, goal-oriented, adaptive, communicative, flexible, active and collaborative.

2 Implementation Issues

As the mobile agents move freely through a network, they should be platform independent [4]. Thus mobile code technologies like Java, TCL and Telescript are widely used for development of the agents. Various Java base mobile agent frameworks are ASDK (Aglets Software Development Kit), Concordia, Voyager etc. Other frameworks viz. Ara, DAgents and Telescript are based on TCL and Telescript. In this work we have used IBM’s ASDK.

Aglets are Java based autonomous and intelligent mobile agents. An aglet (agile applet or agent + applet) carries its state as well as data along with it while moving in a network. Aglets Software Development Kit (ASDK) developed by IBM is a Java based framework for implementing Aglets [8][9].

It goes through various methods in its life cycle. They are listed as follows:

- **Creation** - created in a context and starts running.
- **Cloning** - Aglet duplicated, but with new ID and execution restarted.
- **Dispatching** - ‘Pushed’ out from current context and restarted in another with state maintained.
• **Retraction** - ‘Pulled’ from current context and restarted with state maintained.
• **Deactivation** - Aglet is temporary halted and stored to secondary storage. Activation restores Aglet to same context.
• **Disposal** - Halts execution and removes Aglet from context.

Each aglet will have an associated aglet proxy which controls and calls the aglet’s life cycle methods as per the need. Aglet’s life cycle is depicted in Fig.2

Tahiti is an application program that runs as an agent server. You can run multiple Tahiti servers on a single computer by assigning them different port numbers. Tahiti provides a user interface for monitoring, creating, dispatching, and disposing of agents and for setting the agent access privileges for the agent server [10].

![Fig.2. Aglet’s life cycle](image)

3 Distributed Information Retrieval

Information retrieval from relational databases in large and heterogeneous distributed and web systems needs dynamic interaction of programs with usually heavy network traffic in terms of data transfer between the nodes in the network. This may lead to reduced Quality of Service (QoS) in terms of decreased performance, increased latency and security problems.

Consider this typical situation: You have a central data server that has to collect data from several computers. Examples of which could be a WWW search engine collecting data from web servers all over the world and a central information server in a company collecting data from different departments.

There are different solutions to this problem. In the conventional client – server computing model, a central information server collects data from several computers which can be data servers. This leads to transfer of large amounts of data as the central server
collects all the data before processing. It also leads to more time delay. This can be modified by having local search engines at the local data servers. The local data servers process the data and the central server collects only filtered data. The network traffic will be reduced. But the local search engines in all the nodes have to be modified, if they need update.

A mobile agent based solution is more appropriate and efficient here. In this scenario, an aglet moves through various nodes in the network, connects to the databases residing in them and retrieves the confined information as depicted in Fig.3. When the search algorithm changes a new agent is sent on his way [2][5][6].

Fig.3. Information Retrieval using mobile agents

There are two different ways to retrieve the required information. One way is to dispatch multiple agents to multiple nodes. Each agent visits a particular node, retrieves the desired information and returns to the parent node. The second way is to use a single agent that visits the different nodes one-by-one. It collects data from one node and then visits the next node and so on. After visiting all nodes it returns to the parent node finally. The two methods differ by the computing overhead and latency. The databases used may be homogeneous or heterogeneous databases.

Mobile agents have several advantages in distributed information-retrieval applications. Network traffic is decreased drastically as an agent migrates to the location of a needed resource and interacts with the local resources without transmitting intermediate data across the network. This also reduces latency. Also an agent can respond to user actions rapidly, by migrating to the location of the user. Even the network connection need not be maintained continuously. These features make mobile agents particularly attractive in mobile computing applications, which often must deal with low-bandwidth, high-latency, and unreliable network links [6].

4 Results and Discussion

The network environment is simulated here through different ports in the same system. In this example, the aglet moves to a different port in the same system i.e. local host and connects to the database and retrieves the needed information. In this way it can also connect to other databases and collect the required information. As mentioned earlier there are two ways to acquire the
required information: One way is to dispatch multiple aglets to multiple nodes and the second way is to dispatch a single aglet which moves through the nodes one-by-one.

The first method to use multiple aglets is described as follows. Three different aglets are created and dispatched to three nodes in the network indicated by different ports viz. 2000, 3000 and 4000. They connect to the local databases containing information about some students and retrieve the toppers list based on the criterion of highest aggregate percentage. They return to the parent server and handover the collected information. Now the parent server can find the overall topper from the filtered data collected. The parent in this case indicates administration authority of a college, while the three nodes indicate three departments. The parent node is indicated by the default port 4434. Fig.4 shows the deployment of three aglets and dispatching to different nodes indicated by different ports. Fig.5 shows the results of database connection at one of the remote nodes and retrieved records from the database residing in that node.

![Image of command prompt showing deployment and results of aglets](image-url)
The second method of using single aglet is described as follows. In this case only one aglet is dispatched from the parent server to the data nodes indicated by ports 2000 and 3000. The databases indicate sales data of various branches of a retail store. The product with highest sales is to be determined. The aglet visits the two nodes one-by-one and returns the data to the parent. Now the parent can find the product with highest sales. Fig.6 shows the final retrieved result after the aglet returns to the parent node. Fig.7 shows the final retrieved result after the aglet returns to the parent node.

Fig.5. Retrieving records from the database

Fig.6. Retrieved data from one of the remote nodes
In the first method we used homogeneous databases at different ports, while in the second method heterogeneous databases are used at different ports.

5 Conclusion

Fast and efficient information retrieval in distributed environment is indeed a challenging task. In this paper we discussed how a mobile agent based framework can be used effectively for such distributed database access. It is proven to be better in performance than the conventional client – server methodology. We used the IBM’s Aglet Software Development Kit (ASDK) for creation and deployment of mobile agents. We have also elaborated the differences between usage of multiple agents and a single agent. Mobile agents are gaining importance in variety of applications that include mobile computing, data mining, e-commerce and network management.

References


International Conference on Information Systems Design and Intelligent Applications-2012 (INDIA 2012)
Academia/ Industry working conference, Research challenges, 2000


