

A COMPARISON OF MOBILE AGENT AND CLIENT-SERVER PARADIGMS FOR INFORMATION RETRIEVAL TASKS IN VIRTUAL ENTERPRISES

Ravi Jain, Farooq Anjum and Amjad Umar
Applied Research
Telcordia Technologies, Inc. (formerly Bellcore)
Morristown, NJ.

Abstract—In next-generation enterprises it will become increasingly important to retrieve information efficiently and rapidly from widely dispersed sites in a virtual enterprise, and the number of users who wish to do using wireless and portable devices will increase significantly. This paper considers the use of mobile agent technology rather than traditional client-server computing for information retrieval by mobile and wireless users in a virtual enterprise. We argue that to be successful mobile agent platforms must coexist with, and be presented to the applications programmer side-by-side with, traditional client-server middleware like CORBA and DCOM, and we sketch a middleware architecture for doing so. We then develop an analytical model that examines the claimed performance benefits of mobile agents over client-server computing for a mobile information retrieval scenario. Our evaluation of the model shows that mobile agents are not always better than client-server calls in terms of average response times; they are 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 of the virtual enterprise is error-prone.¹

I. INTRODUCTION

Next generation enterprises will be characterized by new organizational structures, such as increased reliance on virtual operations and “electronically glued” organizations. It is also clear that next-generation enterprises will have an increasing percentage of personnel, be they employees, suppliers or customers, who are using a wide range of wireless technology and devices. One of the key facilities personnel in next generation enterprises will require is the ability to efficiently retrieve, organize, manage and leverage information and knowledge from widely dispersed sources within, as well as from outside, the virtual organization. This activity represents a significant challenge as it tends to require a great deal of human involvement, which increases the cost and delay associated with it. Thus any tool or technology that can help automate this activity has a potential for a significant payoff. This paper considers the tools and technologies that can be used to automate information retrieval tasks for mobile and wireless users in next-generation enterprises.

There has been tremendous interest in the past few years in using mobile agent technology for next-generation enterprises. In particular, mobile agents seem have been proposed for automating the task of retrieving, organizing and filtering information located at widely dispersed sites [8], [3]. Mobile agent systems have been built and demonstrated that indicate mobile agents can be used for information retrieval activities. It has been ar-

gued that mobile agents are particularly suitable for supporting mobile and wireless users, who may be subject to frequent voluntary or involuntary disconnections from the network and may be operating using mobile devices with limited resources and communicating over links of limited bandwidths and potentially high costs [8], [2], [3]. In that case, it may be beneficial to have the mobile user simply launch a mobile agent on the fixed network, and then disconnect, where the mobile agent performs a search for the desired information and retrieves it, rather than having the mobile user stay continuously connected.

In this paper, we consider the following question: is the mobile agent paradigm “better” than using traditional client-server computing for information retrieval tasks? Of course, a question that immediately arises is, what is a mobile agent [4]? As a working definition, in this paper a *mobile agent* consists of a self-contained piece of software that can migrate and execute on different machines in a dynamic networked environment, and that senses and (re)acts autonomously and proactively in this environment to realize a set of goals or tasks.

Most justifications for using mobile agents [1], [2] can be divided into two broad categories, which we call *performance benefits* and *software engineering* benefits. The performance benefits of mobile agents include reduction in network bandwidth consumption, reduced latency, reduced computation, and increased fault-tolerance. The software engineering benefits state that the mobile agent paradigm can help application programmers and designers conceptualize solutions better in that the paradigm may be more naturally suited to certain types of applications, that they can help improve code modularity and reusability, that they can help hide network, system and protocol heterogeneity, etc. In this paper we focus on quantifying the performance benefits of mobile agents over traditional client-server computing. As a concrete case study of considerable importance to next-generation enterprises, we quantitatively study the performance benefits of mobile agents for information retrieval tasks. To our knowledge there has been no similar direct quantitative comparison of mobile agent and client-server computing in the literature.

We also briefly consider the software engineering aspects of mobile agents. However, instead of comparing them with client-server computing, we propose that in fact to obtain the bene-

¹ Copyright 2000 Telcordia Technologies, Inc. All Rights Reserved

fits of software modularity and reuse, mobile agent technology should be presented to the application programmer as a form of middleware, on par with other middleware technologies such as CORBA or DCOM [9], and the Java family of technologies (RMI [6], Jini [13], Enterprise JavaBeans [12], etc.) Middleware is particularly needed for developing applications for mobile and wireless users, which are expected to be an increasingly important segment of the workforce in next-generation enterprises. It seems likely (to us) that mobile agent technology will not replace “traditional” middleware technologies (e.g. Java, CORBA) in the near-term or the medium term, but must coexist with it. We propose a middleware architecture for developing applications where the applications programmer can leverage traditional middleware technologies or mobile agent technologies.

This paper is organized as follows. In the following section we describe a middleware architecture to allow mobile agents to be used for applications supporting mobile and wireless users. In sec. III we describe a simple analytical model for comparing the performance benefits of using mobile agents rather than client-server computing. In sec. IV we describe the results of evaluating that model for an example scenario, and in sec. V we end with some conclusions.

II. UNIFIED MIDDLEWARE

It is clear that middleware will be an important part of the software and IT infrastructure of next-generation enterprises. Middleware can be regarded as a layer of software which provides the common software functions and building blocks required by a wide range of applications and which reside at a layer above device and protocol drivers, operating systems and other system software. Middleware is thus particularly useful for providing software reuse and for hiding the heterogeneity of underlying system software, protocols and devices from applications.

Middleware is especially relevant for mobile and wireless access to enterprise information systems. In fact, we have argued that next-generation enterprise information systems should be designed assuming that mobile and wireless users are the norm rather than the exception, and assuming that every user is (potentially) a mobile and/or wireless user [10]. However, study shows that existing commercial middleware platforms, like CORBA and DCOM, do not provide the functions commonly required by applications for mobile and wireless users. These functions include profile management (e.g., a profile that describes what communication devices the user is using at different times of the day, or what information services the user subscribes to), mailbox management (for the large variety of mailboxes – email, fax, voicemail, etc. – that users may have), cross-media translation and notification (text-to-speech, email-to-fax, etc.) and so on. Thus we have developed a layer of middleware, called *Mobitrix* that provides these functions in a well-defined, unified manner on top of a CORBA platform.

In addition, we find that commercial middleware platforms use protocols (e.g. CORBA’s IIOP on top of TCP [9]) that are inefficient when run over wireless links, and are not designed to handle user mobility well. Thus below the commercial middleware is a relatively thin layer of software which we call the

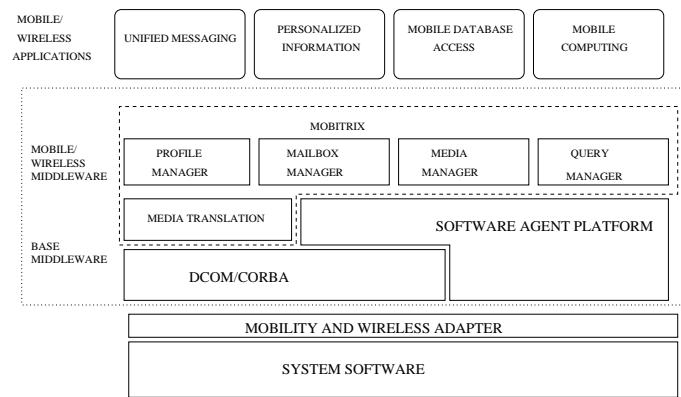


Fig. 1. Architecture for a Unified Middleware

wireless and mobility adapter that hides the effects of wireless links and mobility from the middleware.

In this paper we argue that to be successful mobile agent technology needs to be integrated into the mainstream of industrial software development rather than being an exotic and specialized offshoot. To achieve this mobile agent technology should be presented to the applications programmer as another useful tool in the set of tools available for rapidly creating robust applications from reusable software components. Thus mobile agent platforms should become integrated, or at least be available side-by-side, with commercial middleware platforms. Thus a programmer may choose to use mobile agent technology for some applications and client-server technology for others, since, as we will show later, there are some situations where each has performance benefits over the other.

We thus propose the middleware architecture shown in Figure 1. Applications for mobile users can be written using the existing Mobitrix facilities, which run on top of CORBA. However, they may also utilize mobile agent platforms (e.g. Aglets [5] or Concordia [11]). In general a mobile agent platform could itself run on top of a commercial middleware platform like CORBA (e.g., the MASIF proposal [7] for CORBA) or run directly on top of the mobility and wireless adapter.

III. PERFORMANCE MODEL

In this section we develop a simple analytical model in order to study the performance of both the mobile agents and a client-server based mechanism. We consider N virtual enterprises each having its own web server. A user connected over a wireless link is then assumed to require information stored on one of these servers. Note that this user can represent a mobile user where the mobility of the user is neglected for ease of analysis. This user might require to access information from the servers since the user wishes to either purchase a product or requires some information about a product. The web servers are assumed to be connected to the internet. The mobile user can connect through a basestation to the internet and thereby to these web servers.

We consider two paradigms which can be used to access information from the web servers by the user. In the first method called the client server method the user can do repeated client server calls. So this requires message passing from the user’s

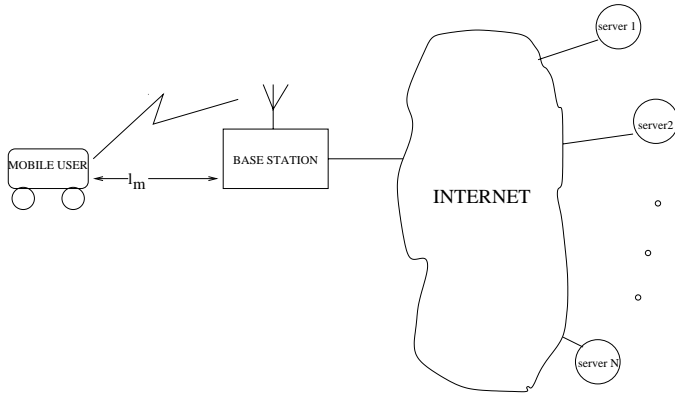


Fig. 2. Illustration of the scenario of the mobile user trying to access the web servers of different virtual enterprises

terminal over the wireless link and through the basestation on to the different servers. The response is then returned back to the user over the same path. If the response does not fetch the information desired then the cycle has to be repeated over the other servers. The second method is called the mobile agents method. In this case the user is assumed to have access to a mobile agent on the user's terminal. The mobile agent is embedded with enough intelligence and launched by the user into the wireline network through the basestation. The mobile agent is then assumed to visit the different servers sequentially until it obtains the information of interest. Once the information of interest is obtained the mobile agent returns back to the user over the wireless link. We would also like to remark that since mobile agents are autonomous entities, they can also perform compression on the data prior to return but we do not consider this in the paper. We also neglect the security aspects of a mobile agent visiting different servers and using the resources locally.

We consider that each of the the client-server calls constitutes a message. In the sequel we consider the error rate over the wireless link in terms of packet loss rate instead of the bit error rate. Note that it is straightforward to convert the bit error rate over a wireless link into the packet error rate given the packet size. We also assume that the code for a mobile agent is equivalent to M client-server calls or M messages.

The sequence in which the servers are searched is assumed to be determined a priori. Thus a figure representing this scenario is shown in Figure 2. The probability of succeeding in finding the relevant information at a server i is p_i whereas the time taken to process the query at server i is denoted by t_i . The latency over the wireless link from the mobile user to the basestation is denoted by l_m while the latency of the links from the basestation to the i th server is denoted by $l_{0,i}$. The latency of the links from the i th server to the $i + 1$ th server is denoted by $l_{i,i+1}$. Let p denote the probability of losing a message over the wireless link.

Let \hat{l}_{0i} denote the sum of the latencies due to the wireless link and the wireline link between the i th site and the basestation. Then the expression for the average delay D_{cs} in finding the information using client-server calls is given by

$$D_{cs} = p_1(2\hat{l}_{01} + t_1) + p_2(2\hat{l}_{01} + t_1 + 2\hat{l}_{02} + t_2) + \dots$$

$$\begin{aligned} &+ p_k(2\hat{l}_{01} + t_1 + 2\hat{l}_{02} + t_2 + \dots + 2\hat{l}_{0k} + t_k) + \\ &\dots + p_N(2\hat{l}_{01} + t_1 + \dots + 2\hat{l}_{0N} + t_N) \\ &= \sum_{j=1}^N p_j(2\hat{l}_{01} + t_1 + 2\hat{l}_{02} + t_2 + \dots + 2\hat{l}_{0j} + t_j) \end{aligned} \quad (1)$$

while in case of mobile agents the corresponding expression will be

$$\begin{aligned} D_{ma} &= p_1(2\hat{l}_{01} + t_1) + p_2(\hat{l}_{01} + t_1 + l_{12} + t_2 + \hat{l}_{02}) + \dots + \\ &p_k(\hat{l}_{01} + t_1 + l_{12} + t_2 + \dots + l_{(k-1)k} + t_k + \hat{l}_{0k}) \\ &+ \dots + p_N(\hat{l}_{01} + t_1 + \dots + t_N + \hat{l}_{0N}) \\ &= \sum_{j=1}^N p_j(\hat{l}_{01} + t_1 + l_{12} + t_2 + \dots + l_{(j-1)j} + t_j + \hat{l}_{0j}) \end{aligned} \quad (2)$$

where

$$\hat{l}_{0i} = l_{0i} + \frac{2pl_m}{1-p} + l_m \quad (3)$$

This equation is obtained by considering the fact that a message might have to be retransmitted over the wireless link if the message does not go through successfully. Here we assume that the message is retransmitted until it is transferred from the mobile user's terminal to the basestation successfully. We also assume that the sender knows after a round trip time whether the message has reached the receiver successfully or will have to be retransmitted.

In the next section we look at these equations to understand the conditions under which mobile agents are advantageous.

IV. MODEL EVALUATION

In this section we compare the client-server and mobile agent paradigms based on the equations given in the earlier section. We consider a scenario with N servers. For the purpose of analysis each of the parameters p_i , $l_{i,i+1}$ is assumed to be independently distributed between 0 and 1. The sequence in which to search the web servers is obtained very simply by ordering the web servers in terms of decreasing p_i . Thus, the web server searched first is the one offering the highest probability of success in terms of the information to be found. We have to remark here that in the real world the assumption that the probability of success at a web server is known is not always true. But we do not concern ourselves with how to obtain this probability nor with estimating this probability adaptively in this paper.

In Figure 3 we consider the effect of the probability of loss over the wireless link on the client server and mobile agent paradigms. On the x-axis we plot the probability of packet loss over the wireless link while on the y-axis we plot the expected latency as obtained from the equations given in the previous section. In this figure we assume that the size of the messages for the client server calls and the size of the mobile agent is the same. It is to be remarked that the size of the mobile agent might increase especially when complex tasks are to be performed. We will look into this aspect later. We would also like to remark

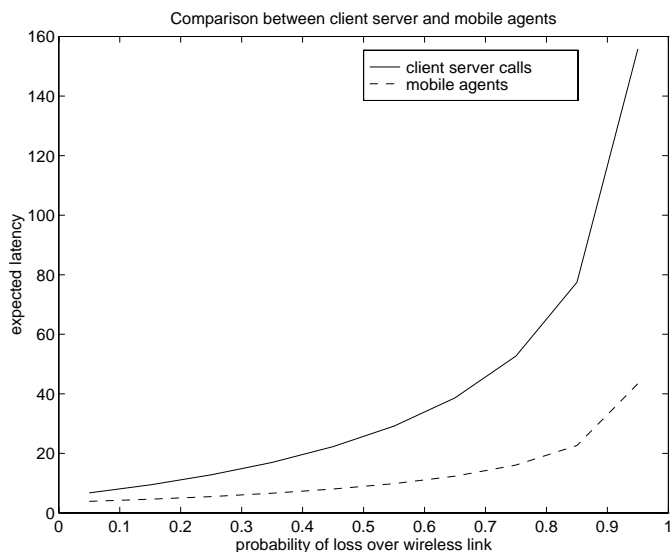


Fig. 3. Effects of wireless losses over mobile agent and client-server paradigms with 10 servers

that we also neglect the fact that client-server calls might result in large sized results being returned to the user. With these assumptions, we can see from Figure 3 that as the probability of loss over the wireless link increases the expected latency when using mobile agents is far less than the expected latency with client-server calls.

In Figure 4 we also consider the fact that the size of the mobile agent may be M times the size of the client-server calls. In this case we also assume that the processing time for the mobile agents is also going to be M times the processing time required for client-server calls. Note that even in this case the different servers are visited until success is obtained in the form of the information being looked for being obtained. We plot the scale factor M on the x-axis while the expected latency is again plotted on the y-axis. The probability of loss over the wireless link is fixed at 0.5. We have again neglected the increased size of the responses in case of the client-server calls. With this, we can see from the figure that as the size of the mobile agent increases, the advantage of mobile agents over client-server calls is lost.

V. CONCLUSIONS

We have two main contributions in this paper. First, we have briefly argued that mobile agent technology should be presented to the applications programmer, from a software engineering point of view, as a form of middleware. We have presented a high-level view of a unified middleware architecture for developing applications for mobile users that places mobile agents in the context of traditional commercial middleware platforms like CORBA and DCOM.

Secondly, we have developed an analytical model to quantify the performance benefits of using mobile agent technology rather than traditional client-server techniques for retrieving information on behalf of a mobile user. We have evaluated the model for an example scenario, focusing on the response time to an information retrieval query as the metric of interest. The results show that it is not clear that mobile agent are always preferable to client-server computing, or vice versa. In particu-

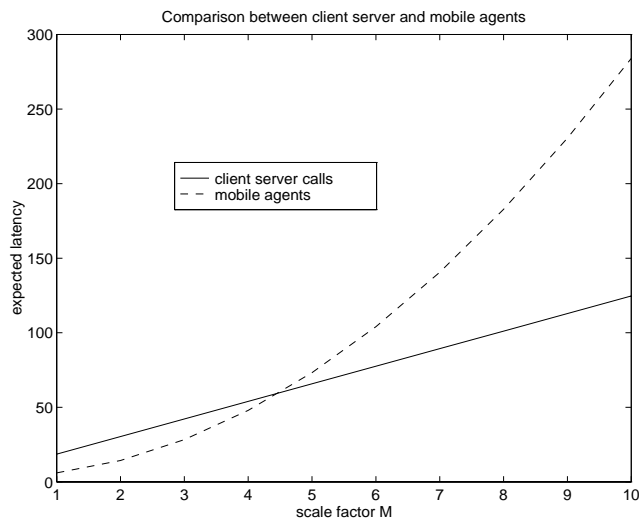


Fig. 4. Effects of the mobile agent size considering 10 servers

lar, if, for a given query, the code size of the mobile agent exceeds that of the corresponding client-server request message, the response time using mobile agents can be significantly more than using client-server calls. On the other hand, if the user has a wireless link connecting the client device to the fixed wired network of the virtual enterprise, the improvement in response time using mobile agents compared to client-server computing increases as the quality of the link decreases.

We are continuing to address the issues raised in this paper. Along one thread, we are investigating integration of mobile agent technology with the CORBA-based Mobitrix platform. Along another thread, we are developing the simple analytical performance model to improve its accuracy and broaden its scope.

REFERENCES

- [1] Harrison C.G. Chess D.M and A. Kershenbaum. *Mobile Agents: Are they a good idea*. IBM Research Report RC 19887, Oct 1994.
- [2] Lange D.B. and Oshima M. Seven good reasons for mobile agents. *Comm. ACM*, 42(3):88–89, March 1999.
- [3] Brewington B. et.al. Mobile agents for distributed information retrieval. *Intelligent Information Agents, Springer-Verlag*, pages 355–395, 1999.
- [4] S. Franklin and A. Graesser. Is it an agent, or just a program?: A taxonomy for autonomous agents. Proc. Third Intl. Workshop on Agent Theories, Architectures and Languages, Springer-Verlag, 1996.
- [5] Tai H. and K. Kosaka. The aglets project. *Comm. ACM*, 42(3):100–101, Mar 1999.
- [6] Farley J. *Java Distributed Computing*. O'Reilly, 1998.
- [7] et al Milojicic, D. Masif: The omg mobile agent system interoperability facility. pages 50–67. Proc. Second Intl. Workshop on Mobile Agents, Springer-Verlag, Sept 1998.
- [8] Comm. of the ACM. *Multiagent Systems on the Net and Agents in E-commerce*, volume 42. Mar 1999.
- [9] R. Orfali and D. Harkey. *Client/server Programming with Java and CORBA*. Wiley, 1999.
- [10] Jain. R. Middleware for nomadic access to enterprise information systems. Proc. IEEE Enterprise Networking and Computing Workshop (ENCOM), June 1998.
- [11] Koblick R. Concordia. *Comm. ACM*, 42(3):96–97, March 1999.
- [12] Valesky T. *Enterprise Javabeans*. Addison-Wesley, 1999.
- [13] Edwards W.K. *Core Jini*. Prentice-Hall, 1999.

Ravi Jain received a Ph.D in computer science from the University of Texas at Austin in 1992. Prior to that he worked for several years on developing communications and systems software, performance modeling and parallel programming. Cur-

rently he is director of the Middleware and Mobile Applications Research group at Telcordia Technologies. His interests include programmability, middleware and applications for next generation networks, mobile Internet access and applications, and mobile and wireless networking.

Farooq Anjum is a Research Scientist at Telcordia. He is currently active in several research projects including the design of a compensating middleware namely the IDM (ARL funded), investigations into the use of agent technology, analysis of TCP behavior over wireless links using cross layer techniques as well as activities like Wireless Application Protocol (WAP) and Java APIs for Advanced Integrated Networks (JAIN). Farooq joined Telcordia after completing a Ph.D. in Electrical and Computer Engineering from the University of Maryland at College Park.

Amjad Umar is the Director of the Advanced Distributed Systems Group at Telcordia Technologies and an Adjunct Professor at Rutgers University. His more than 20 years of experience includes software development, research, management and consulting assignments in the telecommunications industry, manufacturing organizations, educational institutions, and organizations in England, Singapore, China, Italy, Argentina, and Canada. He is the author of three Prentice Hall books: "Application (Re)Engineering: Building Web-based Applications and Dealing with Legacies", "Object Oriented Client/Server Internet Environments", and "Distributed Computing and Client-Server Systems". He has an M.S. in Computer, Information and Control Engineering and a Ph.D. in Information Systems Engineering (Industrial Engineering and Operations Research Department) from the University of Michigan.