Observations on Client-Server and Mobile Agent Paradigms for Resource Allocation

M. Bakhouya, J. Gaber and A. Koukam
Laboratoire SeT
Université de Technologie de Belfort-Montbéliard
90000 Belfort, France
{bakhouya, gaber, abder.koukam}@utbm.fr

Abstract

In this paper, we analyze the performance of a mobile agent based approach for discovering and allocating resources in large scale networks. We compare this approach with the traditional client/server approach by considering four scenarios for processing requests in the distributed network.

1. Introduction

Large scale networked environment, such as Internet, offers a high degree of distribution for data, control and resources access. This potential properties gives users a powerful mechanisms for discovering and accessing resources distributed across the network. However, large scale networks still require a new operational model to use resources efficiently and reduce the need for administration necessary in the classical client-server paradigm (C/S). This traditional approach becomes impracticable for very large networks as it assumes that users should be able to locate servers holding the required resources. More precisely, when a user submit a request, he should be able to specify the destination server holding the appropriate resource. For this purpose, a user needs to maintain a location list of servers containing the specific resources he attempts to use in a local database. The client can also stores a repository list from which the servers can be selected. Distributed environments, like CORBA and TINA[6], provide this function via a repository-type service (e.g., naming service) for locating appropriate server components. This service type is a fundamental part of these distributed environments based on the Client/Server approach.

Mobile Agent based approach (MA) has received great attention in the last years as a promising alternative to the traditional client-server paradigm[2]. The original motivation for mobile agents was in Web-related information processing, especially in search-retrieval process[1, 5, 7, 8, 10].

In this paper, we analyze the performance of a mobile agent based approach for discovering and allocating resources on large scale networks. We compare this approach with the traditional Client/Server approach by considering four scenarios for processing requests in the distributed network.

The rest of this paper is organized as follows. We describe in the section 2 the performance model including the resource definition allocation of resource scenarios. In section 3 and 4 we describe an analytical model for comparing the performance benefits of both mechanisms of interaction for allowing resources, the client-server and the mobile agent paradigm with their time to completion. In section 5 we describe the results of evaluating each model. We end this paper by some concluding remarks and some of our further investigations.

2. The Model

We define a large communication network as a supernetwork composed of a collection of networks connected by boundary nodes.
A resource is an entity that provides state and functionality to be utilized by other entities. Resources are divided into two basic categories [3]: system resources and application resources. System resources are bound to specific hosts, representing hardware devises (e.g. disk) or logical system objects (e.g. socket). Application resources are software entities which are managed by an application. To use a resource, the user initiates a call by submitting a request to a server that manages and controls this resource. We investigate in this paper resource allocation scenarios in the Client/Server approach and in the mobile agent based approach.

In this paper, we consider that the performance of a re-

source allocation strategy is measured by the total time required to complete the resource requests. The completion time is the time between when the request is initiated by the user and when it is satisfied. Obviously the completion time is important to the user of the network.

In the Client/Server approach, each request comes with a destination server. We consider two scenarios. In the first one, the user specifies a destination server and submit its request to that server. In the second scenario, a destination server is chosen uniformly at random from a fixed list of servers holding the required resource and the user submit its request to the selected server.

In the mobile agent based approach, the user initiates a request without specifying any destination server. We consider also two scenarios. In the first one, the user launches a discovery agent that jumps from site to site to seek the resource and returns back the result to the user. In the second scenario, the user launches several discovery agents to seek the required resource.

3. C/S scenarios

The client-server paradigm is well-know and widely used. In this paradigm, a server is a computational component placed at a given site and makes available a set of resources. A client component located at another site requests the server the execution of a service via an interaction process. The server performs the required service and may produces a result that will be delivered back to the client[1, 8]. Typical services are the execution of basic procedures for data retrieval and storage or remote evaluations. It should be noted that the fundamental aspect of the Client-Server paradigm is that the client should know *a priori* the server holding the required resources. In the following, we will describe two C/S based scenarios.

3.1. The first C/S based scenario

In this scenario, we assume that the client knows the location of the target server. A client-server interaction includes binding to the server, marshalling, transfer, unmarshalling of the request parameters, execution of the request, and marshaling, transfer and unmarshalling of the reply[10]. For simplicity, time for marshalling and unmarshalling are omitted since they are independent of the C/S interaction process. In a real system the transfer time depends on the distance between two node and the amount of shipped data[1]. Let t_{01} be the request transfer time from the client to the server and let t_{10} be the transfer time of the result from the server back to client. We consider M/M/1 queuing system with arrival rate λ and service rate μ[9]. Denote by ℓ be the average number of client in the queuing system. The execution time of the request at the server is $\frac{1}{\mu}$ and the client average time in queueing is $\frac{\ell}{\lambda}$. The expected completion time for this scenario is given by

$$T^{(1)} = t_{01} + \frac{1}{\mu} + \frac{\ell}{\lambda} + t_{10}$$

3.2. The second C/S based scenario

In this scenario, the client knows in advance N servers that hold the required resource, N is a fixed number. The client selects one of them randomly. The probability to chose the server i is denoted by p_i. The execution time of the request at the ith server is $\frac{1}{\mu_i}$ and the client average time in queueing is $\frac{\ell_i}{\lambda_i}$, ℓ_i is the average number of client in the ith queue and λ_i is the arrival rate. Let t_{0i} be the request transfer time from the client to the ith server and let t_{i0} be the transfer time of the result from the ith server back to client(i.e. node 0). Then, the expected completion time is given by

$$\begin{aligned} T^{(2)} &= \sum_{i=1}^{N} p_i(t_{0i} + \frac{1}{\mu_i} + \frac{\ell_i}{\lambda_i} + t_{i0}) \\ &= \sum_{i=1}^{N} p_i(t_{0i} + t_{i0}) + \sum_{i=1}^{N} p_i(\frac{1}{\mu_i}) + \sum_{i=1}^{N} p_i(\frac{\ell_i}{\lambda_i}) \end{aligned}$$

$T^{(2)}$ can be rewritten as follows

$$T^{(2)} = E(X) + E(Y) + E(Z)$$

where X, Y and Z are random variables with the respectively distributions $p(X = t_{0i} + t_{i0}) = p_i$, $p(Y = \frac{1}{\mu_i}) = p_i$ and $p(Z = \frac{\ell_i}{\lambda_i}) = p_i$. $E(X)$ indicates the average time of request and reply routing, $E(Y)$ is the average request execution time and $E(Z)$ is the average client queueing time.

4. Mobile Agent scenarios

A mobile agent is an software entity which may move from location to location to meet other agents or to access resources provided at each location. The mobility of the agent is the basic difference from the client-server approach [1, 4, 10]. It should be noted that the fundamental aspect of the mobile agent paradigm is that the client does not know *a priori* the server holding the needed resources. We consider two mobile agent based scenarios.

4.1. The first mobile agent based scenario

In this scenario, one mobile agent is associated to each user request. The user launches a discovery agent that jumps from site to site to seek the resource and returns back the result to the user. The interaction process includes

marshalling, transport, unmarshalling and execution of the request by the server. With the same simplifying assumptions as above, the time of marshaling and unmarshalling are omitted. Let t_{ij} denotes the agent migration time between nodes i and j. Let t_{i0} be the reply transfer time from the node holding the useful resource to the user node (i.e., node 0). The request completion time consist of transfer time of request and reply, execution time and the time in queue. The expected time to completion is given by:

$$T_k^{(3)} = \sum_{i=1}^{k} t_{i-1i} + \frac{1}{\mu_k} + \frac{\ell_k}{\lambda_k} + t_{k0}.$$

This formula can be understood as follows. The first term is the travel time from originating site to *kth* server that holds the needed resources. The second term is the execution time at the *kth* server. The third term is the residence time at the kth queue. The last term is the reply transfer time back to the user node. The expected time to completion can be rewritten as follows:

$$T_k^{(3)} = S_{0k} + \frac{1}{\mu_k} + \frac{\ell_k}{\lambda_k} + t_{k0},$$

where $S_{0k} = t_{01} + t_{12} + t_{23} + ... + t_{(k-1)k}$ is the sum of the migration cost time between the user node and the *kth* server holding the required resource.

4.2. The second mobile agent based scenario

In this scenario, the user launches several discovery agents, duplicated into clones, to seek the required resource. More precisely, when an agent arrive to a network boundary node, it splits into clones which migrate toward the subnetworks. Recall that we have defined a large communication network as a super-network composed of a collection of networks connected by boundary nodes.

We consider that the first agent who gets back the result is the candidate of search, i.e. that one who minimize the completion time. If we consider that the client sends m agents, the completion time is the minimum aver all the agents. The corresponding time is given by :

$$T^{(4)} = \min_{i=1..m} (T_k^{(3)}(i))$$

where $T_k^{(3)}(i)$ is the corresponding time, described in the previous section, of the ith agent travelling.

5. Model evaluation

In this section, we compare the C/S and MA scenarios based on the equations given in the previous sections. In order to compare the different scenarios, we have fixed some parameters that are common to all of them.

- Request size (B_{req}): we have supposed that the request made by user is small and constant for any possible situation.
- Request execution mean time at the server ($\frac{1}{\mu}$): the exact value for this time and its distribution greatly depend on the type of request examined and on the characteristics of the server used. We have fixed this parameter as exponential and included between a minimum and a maximum value $(\frac{1}{\mu})_{\min}$ and $(\frac{1}{\mu})_{\max}$.
- Residence request mean time in the queueing ($\frac{\ell}{\lambda}$) : the exact value for this time and its distribution greatly depend on the characteristics of the server and of the inter-arrival exponential distribution. We have fixed this parameter between a minimum and a maximum value $(\frac{\ell}{\lambda})_{\min}$ and $(\frac{\ell}{\lambda})_{\max}$.
- Size of the query reply (B_{rep}): the reply to a request can realistically be considered included between a minimum and a maximum value, with a uniform distribution. We suppose that the size (in bytes) is constant for all scenario.
- Size of the agent (B_A) : the size of the agent is considered to be the sum of the size of request (B_{req}) and the size of code (B_{cod}) and does not vary during migration from one node to another.
- We assume that the available bandwidth is similar for each node pair (B_d).

In a real system the cost of communication depends on the distance between nodes and the amount of shipped data[1]. The distance might be expressed in number of hops, round trip time, available bandwidth or anything else[11]. We assume uniform networks, i.e., the cost of communication is independent of the particular node pair and is proportional to the amount of bytes that are transmitted.

5.1. Comparison of the C/S scenarios

The network load for a simple C/S interaction consists of the size of the request B_{req} sent by the client and the size of the reply B_{rep} returned back by the server. Therefore, the communication cost for a simple interaction on a network with a bandwidth B_d consists of the request and the reply transfer times. Therefore, for the first C/S scenario, $T^{(1)}$ can be rewritten as follows

$$T^{(1)} = \frac{B_{req} + B_{rep}}{B_d} + \frac{1}{\mu} + \frac{\ell}{\lambda}$$

For the second C/S scenario, the completion time as mentioned before is:

$$T^{(2)} = E(X) + E(Y) + E(Z)$$

Based on the assumption that request size (B_{req}) and the query reply (B_{rep}) are fixed and, in addition, if we consider that the network provides a large bandwidth B_d then the communication cost can be negligible. We suppose also that all the machines have the same execution speed. The comparison between $T^{(1)}$ and $T^{(2)}$ is therefore reduced to comparison of the residence time in queueing.

The size of a server queue is the number of requests submitted to that server. In the first C/S scenario, the queue of requests waiting at the single server is as large as the number of submitted requests.

Let consider that in the second C/S scenario, we have N servers. We assume that the probability to select one of them is $1/N$. We consider that M requests are submitted, where $M = O(N)$. Since the destination of each request is random, the probability that r, $r \leq M$, or more requests queue in some particular queue server is at most

$$\begin{aligned}\binom{M}{r}(\frac{1}{N})^r &< (\frac{Me}{r})^r(\frac{1}{N})^r \\ &= (\frac{Me}{Nr})^r\end{aligned}$$

Hence, most queues sizes never grow larger that $O(1)$ when $M = O(N)$. The size of a queue is a constant, independent of N and M. In other words, if each request is headed to a random server destination, then at most $O(1)$ requests are ever contained in the same queue, with probability close to 1. Therefore, the second C/S scenario is better that the first one, when $M = O(N)$.

5.2. C/S and MA scenarios comparison

The completion time for the second C/S scenario is

$$T^{(2)} = E(X) + E(Y) + E(Z)$$

where X, Y and Z are random variables that counts respectively the time of request and reply routing, the request execution time and the client queueing time.

For the first MA scenario, the completion time $T^{(3)}$ is

$$T_k^{(3)} = S_{0k} + \frac{1}{\mu_k} + \frac{\ell_k}{\lambda_k} + t_{k0}$$

We consider $\frac{B_{A(i-1,i)}}{B_{d(i-1,i)}}$ is the migration cost time between two successive nodes $(i-1)$ and i on the agent path and $\frac{B_{R(k,0)}}{B_{d(k,0)}}$ is the reply communication cost between the server holding the required resource and the originating user node. With the assumptions described above that are the size of an agent, denoted by B_A, does not vary during migration from one node to another and the available bandwidth is similar for each node pair, we have $\frac{B_{A(i-1,i)}}{B_{d(i-1,i)}} = \frac{B_A}{B_d}$, $\frac{B_{R(k,0)}}{B_{d(k,0)}} = \frac{B_R}{B_d}$ and $S_{0k} = k\frac{B_A}{B_d}$.

Thus, the time to completion $T_k^{(3)}$ can be rewritten as follows:

$$T_k^{(3)} = k\frac{B_A}{B_d} + \frac{1}{\mu_k} + \frac{\ell_k}{\lambda_k} + \frac{B_R}{B_d}$$

To simplify, we consider that all the servers have the same service rate $\mu_k = \mu$, for all k and also $B_R = B_{rep}$. Denote by $W_k = \frac{\ell_k}{\lambda_k}$ the average time spent in queue k. Hence, $T_k^{(3)}$ becomes

$$T_k^{(3)} = k\frac{B_A}{B_d} + \frac{1}{\mu} + W_k + \frac{B_{rep}}{B_d}$$

Regarding $T^{(1)}$, with the same assumptions, we have $E(X) = \frac{B_{req}+B_{rep}}{B_d}$ and $E(Y) = \frac{1}{\mu}$. Denote by $W = E(Z)$ the average time spent in a queue. Thus $T^{(2)}$ becomes

$$T^{(2)} = \frac{B_{req} + B_{rep}}{B_d} + \frac{1}{\mu} + W$$

In order to compare the C/S scenario with the MA scenario, we consider the difference between $T^{(2)}$ and $T^k(3)$ as follows

$$\begin{aligned}T_k^{(3)} - T^{(1)} =& k\frac{B_A}{B_d} + \frac{1}{\mu} + W_k + \frac{B_{rep}}{B_d} - (\frac{B_{req} + B_{rep}}{B_d}) - \frac{1}{\mu} - W \\ =& k\frac{B_A}{B_d} - \frac{B_{req}}{B_d} + (W_k - W) \\ \leq& k\frac{B_A}{B_d} + (W_k - W)\end{aligned}$$

Denote by $\delta = \frac{B_A}{B_d}$, we have

$$T_k^{(3)} - T^{(1)} \leq k\delta + (W_k - W)$$

According to this equation, the MA scenario is better than the C/S one if

$$W_k \leq W - k\delta$$

In other words, if

$$\frac{\ell_k}{\lambda_k} \leq \frac{\ell}{\lambda} - k\delta,$$

which means that the mobile agent paradigm performs better than client-server paradigm if the total time (the average time in queueing and the migration time to the suitable server) is still less than the time spent in queueing in the apriori known server in C/S paradigm.

In the second MA scenario, an initial agent starts on the requesting site and after a local step, creates replications (i.e. clones) that walks randomly to further sites. This would allow agents to cover a much wide area of servers space in a shorter time. Future investigations, will be done to analyze the impact of the second MA scenario on latency and network load.

6. Conclusion

In this paper, we have analyzed the performance of the mobile agent based approach together with the classical client/server approach for allocating resources in distributed networks. We have compared these two approaches by considering four scenarios for resource allocation. We have showed that the MA approach may be better that the C/S approach under some assumptions. Further investigations concern the extension of our framework to more general models by relaxing the assumed hypothesis.

References

[1] A. Carzaniga, G. P. Picco, and G. Vigna. Designing distributed applications with mobile code paradigms. *Proceedings of the 19th International Conference on Software Engineering, Boston, MA*, 1997.

[2] D. Chess, C. Harrison, and A. Kershenbaum. Mobile agents: Are they a good idea? *IBM T. J. Watson Research Center*, 1994.

[3] Y. Gidron, O. H. I. Ben-Shaul, and Y. Aridor. Dynamic configuration of access control for mobile components in fargo. *Concurency and Computation*, 13(1):5–21, January 2001.

[4] R. Gray, D. Kotz, G. Cybenko, and D. Rus. Mobile agents: Motivations and state-of-the-art systems. *http://citeseer.nj.nec.com/gray00mobile.html*, 2000.

[5] D. Hagimont and L. Ismail. A performance evaluation of the mobile agent paradigm. *Conference on Object-Oriented*, pages 306–313, 1999.

[6] N. D. Hoa. Distributed object computing with tina and corba. *In Proceedings of WDS'97, Week of Doctoral Students at Faculty of Mathematics and Physics, Prague*, 24-27 1997.

[7] R. Jain, F. Anjum, and A. Umar. A comparaison of mobile agent and client -server paradigms for information retrieval tasks in virtual entreprises. *http://www.argreenhouse.com/papers/fanjum/*, 2000.

[8] A. Puliafito, S. Riccobene, and M. Scarpa. Which paradigm should i use? an analytical comparison of the client-server, remote evaluation and mobile agent paradigms. *Concurency and Computation*, 13(1):71–94, 2001.

[9] A. Ruegg. *Processus Stochastiques Avec Applications Aux Phénomènes D'attente et de Fiabilité*. Processes polytechniques romandes, 1 edition, 1989. ISBN 2-88074-168-8, 1015 Lausanne Suisse.

[10] M. Straber and M. Schwehm. A performance model for mobile agent systems. *Proc.Int.Conf.on.Parallel and Distributed Processing Techniques and Application (PDPTA'97), Las Vegas*, pages 1132–1140, 1997.

[11] W. Theilman and K. Rothermel. Disseminating mobile agents for distributed information filtring. *http://www.informatik.uni-stuttgart.de/ipvr/vs/personen/theilmann/asama99Abs.html*, 1999.