Proc. of 3rd International Conference on Mobile Data Management (MDM2002), IEEE Computer Society Press, Jan. 8-11 2002, Singapore, pp 61-68

Building Internet Marketplaces on the Basis of Mobile Agents for Parallel Processing

Yan Wang, Jian Ren
Department of Computer Science
National University of Singapore
3 Science Drive 2, Singapore 117543
{ywang, renjian}@comp.nus.edu.sg

Abstract

In this paper, we propose a framework of Internet marketplaces on the basis of mobile agent. It not only simulates real commercial activities by consumers, agents and merchants, but also provides an environment for parallel processing. The latter is particularly important as more shops (sites) can be searched in real time to provide consumers with better choices. The conducted experiments show that, in comparison with several serial mobile agent models, parallel mobile agent model can improve the performance significantly.

1. Introduction

The advances of web technologies such as the Internet, HTML, Java and XML have greatly pushed the development of Electronic Commerce (EC). Today, many electronic shops (e-shops) publish their product catalogue on the Internet, offering a wide variety of goods. More importantly, consumers are turning to the Internet for such information as well as to purchase their goods online.

However, the wide variety of choices to the consumers has also introduced the problem of information overloading. Moreover, there are so many e-shops and goods for the consumers that it has become too time-consuming, if not impossible, to find the best (cheapest) deal. Hence, further research and development are necessary to gain experiences to provide consumers with a more convenient and user-friendly environment. One promising direction is to exploit mobile agents for e-commerce [1].

As pointed out by Rodrigo [2], future e-commerce models will enhance current models by using mobile agents. On one hand, in our real life, people can turn to a few agents or agencies for buying something such as an air ticket, or renting a house. They can choose a satisfied one from multiple provided plans. On the other hand, the mobile agent scenario offers us more flexibility to apply the consumer/agent/merchant model of real commercial activities to the building of electronic marketplaces and also provides an environment for parallel processing over distributed site [3].

In this paper, we propose a framework for Internet marketplaces that exploits mobile agent technology extensively. It not only supports activities of consumers and merchants, it but also facilitates parallel computation. The latter is especially important as more sites/shops can be searched in a shorter time to provide consumers with better choices in their decision-making. The mobile agent based framework can inherit and extend the conventional client/server architecture of the HTML and applet technologies, which are widely adopted by existing e-shops and logically it is transparent to the end users.

Based on the proposed framework, we have implemented a prototype system and explored several serial mobile agent models and one parallel mobile agent model that can be adopted in such an environment. All models are implemented as 100% Java codes using Aglets system [4]. The system was set up on top of 17 PCs connected in a LAN. The results of conducted experiments show that the parallel mobile agent model outperforms other models by a wide margin.

2. Related work

There has been an increasing amount of research activities to exploit mobile agents to support electronic markets or enterprises.

In [1], Sohn proposed an architecture for electronic market by applying the mobile agent technology. In his work, the market is consisted of conductors and members. The conductor manages the market and members participate in electronic commerce activities. Members are providers, shops and consumers. The conductor provides the framework of the market and manages the setup of members, product ontology and member information. Sohn's work gives us a fundamental description for setting up an electronic market with mobile agents and it introduces some internal activities that these agents should do. But his work addresses only an individual market without any focus on an electronic market community on distributed sites.

Chrysanthis's work views the establishment of a virtual enterprise (VE) as a problem of dynamically expanding and integrating workflows in decentralized, autonomous and interacting workflow management systems [5]. In this framework, mobile agents are employed for advertising, negotiating and exchanging information as well as its management. Chrysanthis's contribution is the idea of using workflows to support multi-organizational processes to form a VE and [5] gives a brief description on how to utilize the mobile agent technology.

Lange briefly introduced a mobile agent based marketplace architecture in [6] and showed that the IBM Aglet Software Developing Kit (ASDK) system [4, 7] is suitable to build an electronic marketplace and the meeting pattern and communication mechanism of Aglets, which are mobile Java objects, can be adopted to meet the requirements for representing the behaviors of mobile agents. In his framework the consumer agent visits marketplaces one by one for shouting of a request and performing negotiation activities. There is, regrettably, no global control mechanism. A similar work is also introduced in [8].

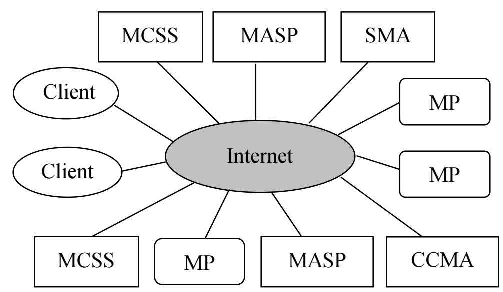
The above-mentioned works benefit much from the deployment of mobile agents, such as good mobility, high autonomy as well as the role simulation and role specification that present the realistic simulation to the real commercial activities. But they simply put mobile agents in a serial working pattern and their global control mechanisms are not clear.

The work of Silva, Papastavrou and Panayiotou all showed the advantages of applying the mobile agent approach to parallel processing over distributed databases or data sources [3, 9 and 10]. A mobile agent can decompose its tasks to multiple sub-mobile agents and dispatch them to distributed sites simultaneously in order to let them work in parallel. Hence, the mobile agent technology is naturally suitable for deploying parallel and distributed computation. The performance is comparable to, and in some sense outperforms the current approach via expensive network and slow network, such as the wireless network or dial-up network.

The performance issue is another important consideration for adopting the mobile agent approach when building electronic marketplaces. Particularly, consumers need to know 'fresh' goods' prices. Those cache strategies adopted by web search engines are not suitable. In addition, the mobile agent approach is suitable for supporting mobile clients since it does not require permanent network connections [11]. Furthermore, the Java based mobile agents inherit the computer-independent feature from Java programs and hence provide the platform-independent integration of heterogeneous databases and data sources [12]. Therefore, building electronic marketplaces on the basis of mobile agents with a uniform framework for the market community is expected to be beneficial to consumers to provide them with the best-buy trades more efficiently over lots of electronic shops.

3. The infrastructure for Internet marketplaces

3.1 Overview



MASP: Mobile Agent Service Provider
MCSS: Marketplace Community Service Server
SMA: Security Management Authority
CCMA: Commercial Credit Management Authority
MP: Marketplace

Figure 1. Overview of the marketplaces

In our proposed framework, there exists a set of marketplaces (see Figure 1). They are connected to the Internet. The Mobile Agent Service Provider (MASP) is an execution environment for mobile agents. A consumer-agent can be created at MASP as the client's request. Such a consumer-agent can reside in the MASP, act as a master agent and dispatch its worker agents to related marketplaces (MP) to fulfill the tasks. Meanwhile, a set of MASPs should be set up and distributed globally. They are similar to today's ISPs (Internet Service Provider).

In the proposed architecture, there also exists a set of Marketplace Community Service Servers (MCSS). A MCSS is responsible for maintaining the information of MPs and e-shops in the MPs. The role and mechanism of MCSS are similar to the DNS (Domain Name Service) server, which offers the conversion between domain name and IP address. But the functions of a MCSS are more complicated. Similar to the DNS servers, all the MCSSs should be distributed in different zones.

3.2 System components

1 **MCSS** (Marketplace Community Service Server): In the proposed architecture, there also exists a set of MCSSs. A MCSS is responsible for maintaining the information of MPs. The information should include the domain names, IP addresses of MPs and e-shops, goods catalogue, and the identifications of the MPM (MP Manager) and shop-agents running at each MP. These information of related MPs and e-shops can be provided when a client tells the MCSS what kind of goods it needs. The role and mechanism of MCSS are similar to the DNS (Domain Name Service) server, which offers the conversion between domain name and IP address. Similar to the DNS server, when only a few MPs are set up, one MCSS can be set up for the serving. When more MPs are set up, a set of MCSSs should be distributed in different zones.

2 **MASP** (Mobile Agent Service Provider): MASP is a provider of the service enabling mobile agents as the response to clients' requests. It is a server provided to registered clients where a consumer-agent is created as the request from the client for searching the information of one or more specified goods. With the client's searching criteria, the consumer-agent will dispatch in parallel a pool of mobile agents to relevant e-shops, which will return the queried results. The whole process is introduced in section 3.5.

3 **SMA** (Security Management Authority): SMA is responsible for generating certificates for all MPs, e-shops and MASPs, and managing them. In addition, SMA is responsible for taking security investigations and making security assessments on those authorized hosts according to attack reports. Here a host donates the MP-Server or E-shop server where mobile agents can be dispatched. Our security dispatch model can be found in [17, 18]

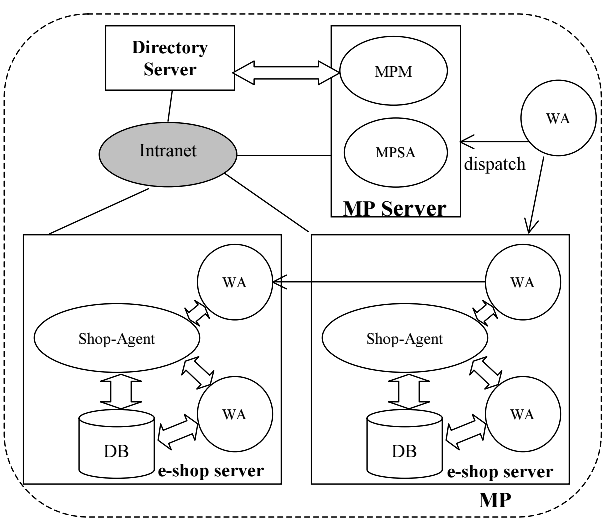


Figure 2. MP components

4 **CCMA** (Commercial Credit Management Authority): CCMA is the authority making commercial credit assessment and management over all e-shops. When merchant cheating occurs, a client can report it to CCMA. After investigation, the commercial credit of the e-shop will be downgraded. On the other hand, successful transactions will help to upgrade the commercial credit.

5 **Client:** A client should be a registered user of any MASP before utilizing the facility of the MASP based on mobile agents. When having become a registered user, a client can
(1) search specified goods through the service based on mobile agents from the MASP till making the payment.
(2) appeal to the CCMA for any merchant-cheating that may occur during the purchase that can hardly be detected before payment. If the cheating is true, the merchant's commercial credit will be deduced that will result in that fewer consumer-agents will be dispatched there.

3.3 MP components

A MP is the Internet marketplace consisting of a set of e-shops that run simultaneously on different servers. Within each MP, it has the facility to accept registrations of e-shops, maintain a directory of them, and authenticate foreign mobile agents.

The components of a MP is presented as follows:

1 **MPSM** (MP Security Manager): MPSM is similar to the firewall of an intranet. It is in charge of:
(1) maintaining the security of the whole MP, such as authenticating coming foreign mobile agents and monitoring the communication out of the MP from a mobile agent or an e-shop, and
(2) broadcasting the certificates of e-shops to other relevant sites, such as MASPs, and
(3) registering the MP to MCSS.

2 **MPM** (MP Manager): A MPM is responsible for the management of the MP, such as accepting the registration of an e-shop in the MP and the application of withdrawal, and maintaining the directory of e-shops in the MPDS directory server (MP Directory Server). The MPM is also responsible for accepting the registration of a MASP so that mobile agents can be dispatched from it.

3 **MP-Server**: It is a server where MPSM and MPM run. It is also an execution environment for incoming mobile agents. An agent dispatched by a master consumer-agent for visiting e-shops in the MP will first arrive here. Only after having passed through the security check by MPSM, it can enter any e-shops in the MP.

4 **Shop-Agent**: A Shop-Agent is an agent running at the shop server that is responsible for
(1) maintaining the shop information and goods information stored in the shop database.

Proc. of 3rd International Conference on Mobile Data Management (MDM2002), IEEE Computer Society Press, Jan. 8-11 2002, Singapore, pp 61-68

(2) periodically sending updated information to MPM for modifying the goods-catalogue of the e-shop maintained in the MPDS (MP Directory Server).
(3) communicating with incoming consumer-agents providing the goods information they required.
(4) monitoring the execution of foreign consumer-agents and protecting the local resources of the e-shop
(5) registering the e-shop to the MPM and through it registering to the MCSS when the e-shop is set up
(6) applying the certificate of the e-shop from SMA and sending it to the MPM

5 **MPDS** (MP Directory Server): Its responsibility is to store the registration information and the goods catalogue information of all e-shops in the MP. Only the MPM can update and maintain them.

6 **E-shop Server**: An e-shop server is the place where the e-shop is set up within the domain of a MP and the shop-agent runs. It is also the execution environment of incoming mobile agents.

3.4 Procedures of setting up a MP and an e-shop

When setting up a MP, the MPM should register the MP to MCSS by sending the following:

(1) MP's name, domain name and IP address
(2) MP's certificate including its public key obtained from SMA
(3) id of the MPM
(4) IP addresses, certificates, directory and goods catalogue of all e-shops in the MP
(5) identifications of corresponding shop-agents
(6) current time

When an e-shop is set up in a MP, the shop-agent should register to the MPM by sending the following:

(1) e-shop's name and IP address
(2) e-shop's certificate including its public key
(3) e-shop's goods catalogue
(4) identification of the shop-agent
(5) current time

Information (1) and (2) are put in the MPDS (MP Directory Server) by MPM. If the goods catalogue of an e-shop is changed, the shop-agent will notify the MPM and MPM will report these changes to MCSS. The MPM will also report to MCSS when any e-shop withdraws or the whole MP withdraws.

3.5 Process workflow

Based on our framework, the process enabling buying and selling can be described as follows:

(1) Input request

For a client, he/she choose a MASP where he/she has registered as a user to input the information of a good such as the name, model, type, some selection criteria for the goods such as the warrantee service and delivery/shipment service, and potential merchants, such as the security rank and commercial credit.

(2) First phase evaluation and searching e-shops

With the request of a client, a consumer-agent is created at the server of the MASP, who will act as a master agent and dispatch a pool of mobile worker agents to qualified e-shops after carrying out the first phase of the two-phase evaluation, which makes an evaluation on security rank, commercial credit of all e-shops that sell the same kind of specified goods. These attributes are obtained from MCSS, SMA and CCMA. After evaluation, a pool of WAs are dispatched to qualified e-shops.

(3) Second phase evaluation

After all mobile agents returned the results, the second phase evaluation is performed on both goods' information and e-shops' security rank and commercial credit. The sorted results are presented to the client by the master agent.

The 2-phase evaluation model uses fuzzy rules to conduct the evaluation. More details can be found in paper [18, 19].

(4) Negotiation

With the client's selection, a few e-shops will be selected for negotiation by dispatching a negotiation-agent. Some negotiation models have been proposed, such as [13]. In this paper, we will not address this issue.

(5) Book and Payment

With the success result of negotiation, one e-shop will be selected to book the goods and make an online secure payment.

(6) Purchase Feedback by Consumers

4. Experiment results

On the basis of the proposed architecture, we have partially implemented a prototype system, which is set up in a LAN consisting of PCs running Window NT, JDK 1.1.6, Aglets 1.0.3 [4] and the SAX APIs (the Simple API for XML) from Sun Microsystems [14].

In order to prove the availability of the proposed architecture and observe the performance improvements by parallel processing, we conducted experiments using a cluster of up to 17 PCs running the prototype system. All PCs have the same configuration of Pentium 200MHz CPU and 64MB RAM. They are connected to the LAN through 10Mbits/sec Ethernet cards. One of them is taken as the MASP while others are e-shops. All mobile agent programs were implemented by using the Aglet system and the Tahiti server for Aglets was set up in each site. In the experiments we compared the efficiency of one parallel mobile agent model with some serial models.

4.1 Models

Table 1. Five models in the experiments

Name	Execution Pattern
Itinerary Agent	Serial
Shuttle Agent	Serial
Serial Agent	Serial
VP Agent	Virtual Parallel
SParallel Agent	Serial Dispatch, Parallel Data Access

We implemented five mobile agent models of execution in the experiments. Three are fully serial mobile agent models, one is a virtually parallel model (VP Agent model) and the rest is parallel mobile agent model. All models except the VP Agent model adopt the master/worker pattern. Each of them has one master consumer-agent. It dispatches one or multiple worker agents and collects the results from them. These serial mobile agent models resemble the typical operations of fully serial searching activities on the web by consumers. In the VP Agent model, the consumer-agent executes locally to access all remote data through HTTP connections. The parallel models used multiple worker agents for parallel data access (see Table 1).

In the first model, an itinerary agent is created by the master consumer-agent with the addresses of a list of e-shops. It migrates and visits all e-shops in its list one by one. When migrating to the next e-shop, it carries the data accumulated from all previous e-shops including the current one. After it has visited all the e-shops, it returns to the MASP and carries back the whole data. This model corresponds to the approach proposed by [7].

In the Shuttle Agent model, there are also one master consumer-agent and only one worker agent. The worker agent is dispatched to an e-shop in the list maintained by the master consumer-agent. When it has read the data, it returns to the MASP carrying the results. After sending the data to the master consumer-agent, it migrates to the next e-shop and repeats the process until it has visited all e-shops and returns all the data.

In the third model, similar to the second one, the master consumer-agent dispatches a worker agent to an e-shop. When the worker agent has obtained the data, it sends it back by a message and then disposes. Having gotten the result, the master consumer-agent will dispatch a new worker agent to the next e-shop until all e-shops are visited.

In the VP Agent model, the master consumer-agent is created at the client end and it starts multiple threads. Each thread locally reads data located in a remote HTTP server. The thread stops when it has finished reading the data and thus gets final results. This model resembles in a more efficient way with virtually parallel searching activities. Obviously, in this model, no worker agent is dispatched.

The fifth model benefits from the fully parallel data access. In this model, the master consumer-agent dispatches multiple worker consumer-agents one by one to all e-shops that should be visited. These worker agents can access local data in parallel and send their results to the master consumer-agent in succession but the dispatch process is fully serial.

For all models, the execution time is calculated for the period from the creation of the master consumer-agent till the moment when all results are collected. In each model with dispatched mobile agent the dispatched worker agent is designed to read data from the XML files of an e-shop directly and locally. In the implementations of all models the master consumer-agent is used to count the execution time with the uniform clock at the MASP side.

The common feature of the implementations of all models with dispatched mobile agents is that the complexity for parsing XML documents is hidden at the e-shop side where work agents, which are dispatched to e-shops from the MASP, finish the parsing tasks. For the VP Agent model, the parsing task is done at the side where the master agent resides. Furthermore, in the implementations of all models, they share the same Java codes for querying data from XML files. And certainly, they are compared with the same environment since they all run on Tahiti servers.

4.2 Performance analysis

The results presented in Figure 3 to Figure 8 are the average obtained from 4 independent executions. We reboot all Tahiti servers for every execution to avoid the impact of the class cache on the execution time of repeated experiments. The variance across different executions of the same model is not significant.

For each model, the dispatched worker agent visiting an e-shop obtains 1Kbytes or 100Kbytes data from the local XML file. The length of the XML file is set to 1 Mbytes, 10 Mbytes and 100 Mbytes respectively. The read data can finally be obtained only after the whole XML file is parsed. The number of e-shops is set to 2, 4, 8 and 16. These variations aim to observe the performance differences among experiments and illustrate how these variations impact the performance.

Figure 3 presents the results from the case of reading 1Kbytes from the 1Mbytes XML file of each e-shop. It shows that the SParallel Agent model is better than serial agent models in most cases. It is the same with Figure 4 to 8. The results in Figure 3 also show that the performances of serial agent models are very close to each other while the parallel processing can gain improvements more significantly.

Similar conclusions can be drawn from the results showed in Figure 4 and 5 where the file size is set as

Proc. of 3rd International Conference on Mobile Data Management (MDM2002), IEEE Computer Society Press,
Jan. 8-11 2002, Singapore, pp 61-68

10Mbytes and 100Mbytes respectively. The difference is that the performance of the SParallel Agent model is greatly improved and it can outperform each serial model.

With respect to the VP Agent model, it can still outperform the three serial mobile agent models in Figure 3. Nevertheless, it becomes as bad as them with the increase of the size of read files in Figure 4 and Figure 5. It shows that when the data set size is small (e.g. 1Mbytes in the experiments), the VP Agent model can outperform other serial mobile agent models because in this case, these mobile agent models would incur more overheads from creating and dispatching worker agents while the time spent on reading a small file is short. However, with the increase of the file size as well as the number of visited e-shops, other serial mobile agent models improve since they can benefit from the worker agent's reading local files and in comparison the VP Agent model should read remote files and it evidently gets more overheads from it (see Figure 4 and 5).

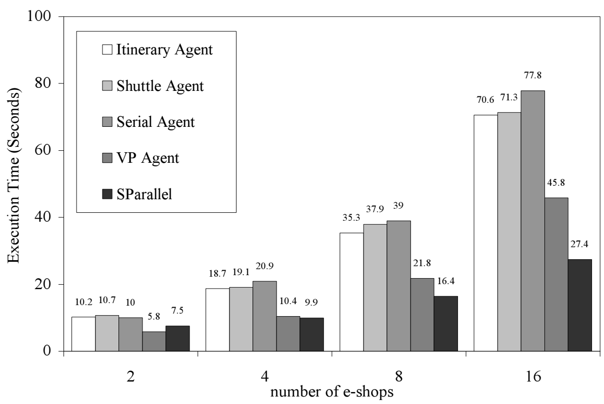


Figure 3. Results for reading 1Kbytes data from the 1Mbytes XML file of every MP

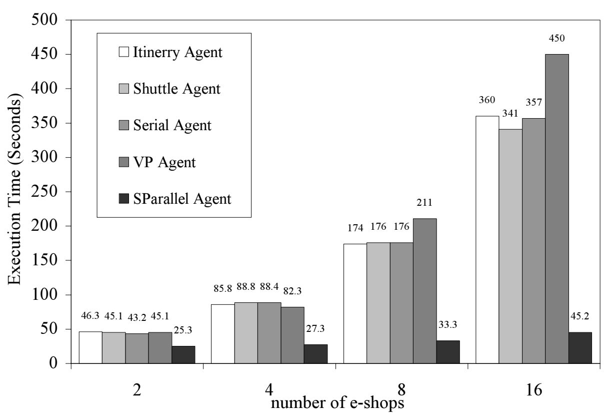


Figure 4. Results for reading 1 Kbytes data from the 10Mbytes XML file of every MP

When the size of data set obtained from each e-shop is set to 100Kbytes, some new changes can be observed. These results are presented in Figure 6, 7 and 8. The most significant change is that the Itinerary Agent model becomes extremely worse than other serial models when the result data sets are obtained from 1MBytes XML files and 10Mbytes XML files as presented in Figure 6 and 7. This shows with the increase of the size of data set it carries, the time for a mobile agent to migrate from one site to another increases significantly. This observation is clearly shown when the experiments are being done.

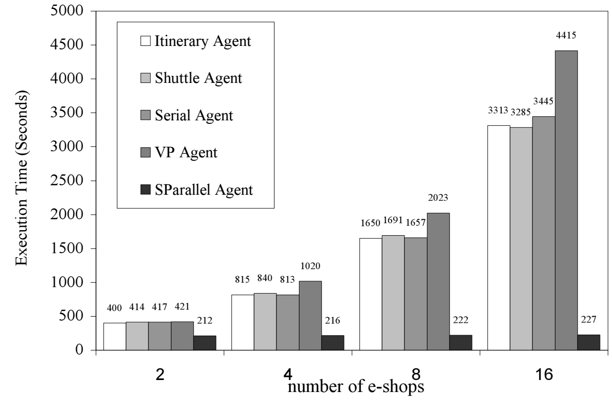


Figure 5. Results for reading 1Kbytes data from the 100Mbytes XML file of every MP

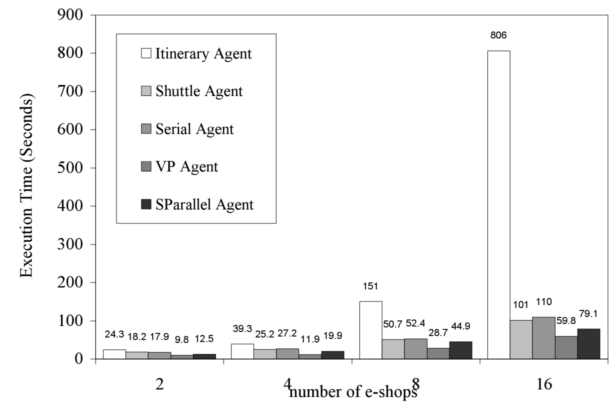


Figure 6. Results for reading 100 Kbytes data from the 1Mbytes XML file of every MP

However, when the result data is obtained from the 100Mbytes XML file, as shown in Figure 8, the difference between the Itinerary Agent model and other serial models is not so significant since most time should be spent on the data accessing and hence the delay from migrations for the Itinerary Agent model is not a critical factor any more.

In comparison with other serial models, the VP Agent model is good when accessing the 1Mbytes files (see Figure 6). However, when the size of file is set to 10Mbytes and 100Mbytes, it becomes worse. As shown in Figure 8, it becomes the worst serial agent model. This shows that when the data size is large, the network delay can significantly impact the performance. This situation becomes worse with the increase of the number of e-shops.

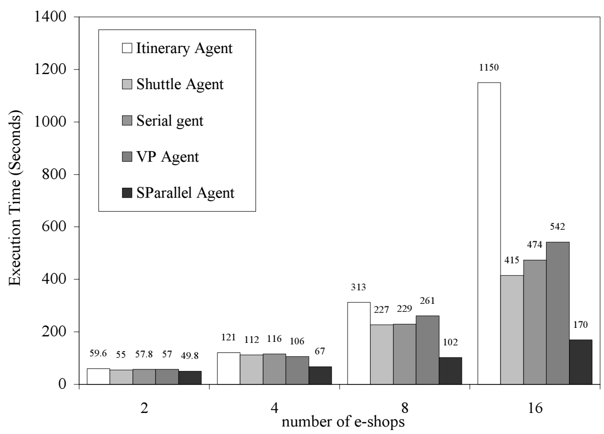


Figure 7. **Results for reading 100 Kbytes data from the 10Mbytes XML file of every MP**

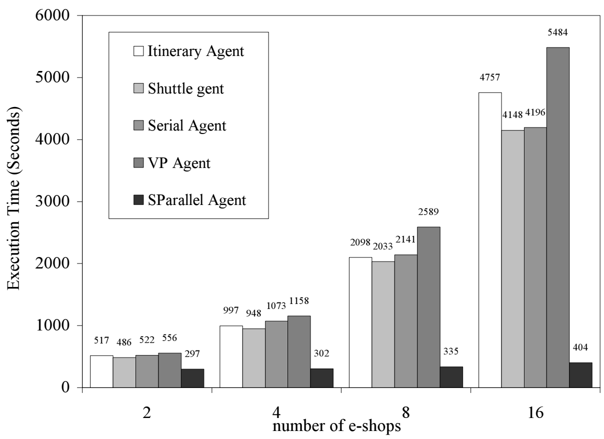


Figure 8. **Results for reading 100 Kbytes data from the 100Mbytes XML file of every MP**

With respect to the SParallel Agent model, it is good in most cases since it can benefit from the parallel execution of multiple dispatched agents. However, if the size of accessed file is 1Mbytes when the time for accessing the file is very short, as shown in Figure 3 and 6, it is not the best all the time because in this case the whole dispatch time becomes the main overhead and the serial dispatch process impacts the whole performance. Its disadvantage can be covered up when the accessed file is large so that the dispatch process is not the main overhead. As a whole, the results show that SParallel mobile agent model can sufficiently improve the performance when the accessed data sets are large and they are distributed over a large number of sites. Since all computers have the same configuration, the saving of execution time is totally obtained from the parallel execution on the basis of mobile agents. Especially, the execution time is not significantly increased with the increase of the number of visited e-shops. This is extremely feasible and meaningful for e-commerce applications and consumers. With respect to the dispatch process, the experiment in [15] shows that binary dispatch model can further improve it with a good margin.

In each experiment, we compared the results of the MT Parallel Agent model with the average of four serial mobile agent models. The result shows that when reading data from each 100Mbytes XML file of 16 e-shops, the SParallel Agent model can obtain over 93% saving of the execution time.

Due to the condition limitations, it is difficult to execute all the experiments to the Internet. However, in the Internet environment, the dispatch time of a mobile agent will become longer and it is more dependent on the current state of network traffic. For the execution of local data accessing, it is dependent on the performance of the e-shop server and how many agents are running there at the same time. No matter in what condition it is, serial agent models will be more dependent on the network traffic and e-shop servers. In contrast, the parallel model will continue to benefit from the parallel executions of worker agents and hereby it will outperform undoubtedly.

5. Conclusions

This paper presents a framework of Internet marketplaces built on the basis of mobile agent technology to support parallel processing over distributed sites. By using the mobile agent model, it provides the possibility to realistically simulate consumers' commercial activities. The Java based mobile agent technology enables the features of light-weighted, portable and platform-independent agents. The architecture is suitable for both mobile and stationary users.

The evident benefit is that the proposed architecture can easily support both serial processing and parallel processing by mobile agents. The searching and negotiating models can be designed and implemented in the control strategy of the master consumer-agent, which uses a pool of worker agents to fulfill the consumer's input tasks in parallel. Some serial processing and the migration of mobile agents offer more flexibility and make the model closer to people's real activities. However, exploiting parallel processing provides the most important benefit and it can help to set up a system with higher efficiency and provide better supports for the consumers' best-buy strategy. The conducted experiments showed the significant performance improvements gained by parallel processing while providing 'fresh' information on goods. Especially, these improvements are gained with the platform-independent feature and the scenario hides the complexity at the e-shop side for parsing XML data, which is practical for current client/server models. In addition, the framework is generic and it is transparent to end-users.

At last, the architecture is based on the Internet and the setting up process is similar to that of the Internet. Obviously, XML is not a replacement of HTML. The XML is used to describe and carry data [16]. The HTML is used to display the data. Therefore, within our proposed architecture, each shop can still set up web pages for individual user to browse and access in the traditional way.

What we should address is that mobile agent approach may not be a replacement of client/server model. But it extends it to be more flexible enabling agent-oriented modeling and programming, and can give interesting solutions for some suitable applications such as e-commerce.

Both mobile agent and XML technologies are very new in comparison with other web technologies. When we implemented this system using the Aglets system, it is not too complicated for the communication between mobile agents with the message mechanism of the Aglets system. Nevertheless, as there exist many Java based mobile agent systems and each mobile agent should run automatically, in the long run, standards and protocols should address some issues, such as the standard communication interface between any mobile or stationary agents, and the description of goods catalogue by XML. With the support of these standards and protocols, the mobile agent and XML technologies can bring the e-commerce into a more efficient stage.

Acknowledgement

This work is partially supported by the NSTB/MOE funded project on Strategic Program on Computer Security (R-252-000-015-112/303).

References

[1] Sohn, S. and Yoo, K. J. "An Architecture of Electronic Market Applying Mobile Agent technology", *Proceedings of 3rd IEEE Symposium on Computers and Communications (ISCC '98)*, Athens, Greece, June 30-July 2, 1998, pp. 359-364.

[2] Rodrigo, T. D. and Stanski, A. "The Evolving Future of Agent-based Electronic Commerce", in *Electronic Commerce: Opportunity and Challenges* (Edited by Rahman S. M. and Raisinghani M. S.), Idea Group Publishing, Hershey, USA, 2000, pp. 337-351.

[3] Panayionou, C., Samaras, G., Pitoura, E. and Evripidou, P. "Parallel Computing Using Java Mobile Agents", *Proceedings of 25th EUROMICRO Conference*, Milan, Italy, September 8-10, 1999, Volume 2, pp. 430-437.

[4] URL: http://www.trl.ibm.co.jp/aglets/

[5] Chrysanthis, P., Znati, T., Banerjee, S. and Chang, S. K. "Establishing Virtual Enterprises by means of Mobile Agents", *Proceedings of Ninth International Workshop on Research Issues on Data Engineering: Information Technology for Virtual Enterprises (RIDE-VE '99),* Sydney, Australia, March 23-24, 1999, pp. 116-123.

[6] Lange, D., and Oshima, M. "Mobile Agents with java: The Aglet API", appears in *Mobility: Process, Computers, and Agents* (edited by Milojicic, D., Douglis, F. and Wheeler, R.), Addison-Wesley Press, Reading, Massachusetts, USA, 1999, pp. 495-512.

[7] Lange, D., and Oshima, M. *Programming and Deploying Java Mobile Agents with Aglets*, Addison-Wesley Press, Massachusetts, USA, 1998.

[8] Dasgupta, P.; Narasimhan, N.; Moser, L.E.; Melliar-Smith, P.M., MAgNET: mobile agents for networked electronic trading, *IEEE Transactions on Knowledge and Data Engineering*, Volume: 11 Issue: 4, July-Aug. 1999, pp. 509 –525.

[9] Silva, L. M., Batista, V., Martins, P. and Soares, G. "Using Mobile Agents for Parallel Processing", *Proceeding of International Symposium on Distributed Objects and Applications (DOA'99),* Edinburgh, Scotland, September 1999, pp. 34-42.

[10] Papastavrou S., Samaras G. and Pitoura E. "Mobile Agents for WWW Distributed Database Access", *Proceedings of 15th International Conference on Data Engineering (ICDE'99),* Sydney, Australia, March 23-26, 1999, pp. 228 –237.

[11] Law, K.C.K. and Wang, Y., "An Agent based Approach for Distributed Database Access in a Mobile Environment", *Electronic Proceedings of the 2nd International Conference on Information, Communications & Signal Processing*, Singapore, December 7-10, 1999.

[12] Stefano A. D, Bello L. L. and Santoro C. "A Distributed Heterogeneous Database System based on Mobile Agents", *Proceedings of 7th IEEE International Workshop on Enabling Technologies: Infrastructure for collaborative Enterprises*, Stanford, California, USA, pp.223-228, June 17-19, 1998.

[13] Guttman, R. H., and Maes, P., Agent-mediated integrative negotiation for retail electronic commerce, *Proceedings of Workshop on Agent Mediated Electronic Trading*, Minneapolis, Minnesota, USA, May 1998, pp. 70-90.

[14] URL: http://www.w3.org/

[15] Wang Y., Dispatching Multiple Mobile Agents in Parallel for Visiting E-Shops, *Proceedings of International Conference on Mobile Data Management (MDM2002),* IEEE Computer Society Press, Jan. 8-11, 2002, Singapore.

[16] URL: http://www.w3schools.com/site/site_intro.asp

[17] Wang Y., Tan K.L., A Secure Model for the Parallel Dispatch of Mobile Agents, *Proceedings of International Conference on Information and Communications Security (ICICS2001*), Xian, China, November 13-16, 2001, Springer-Verlag, LNCS Vol. 2229.

[18] Wang Y., Tan K. L., Ren J. and Pang X., An Agent-mediated, Secure and Efficient Internet Marketplace, *Proceedings of the 4th International Conference on Electronic Commerce Research (ICECR-4)*, Dallas, TX, USA, November 8 - 11, 2001.

[19] Ren J., Wang Y., Pang X. and Tan K. L., An Evaluation Model based on Agent-mediated Internet Marketplaces Enabling Parallel Processing, *Proceedings of the 2nd International Conference on Web Information Systems Engineering (WISE2001)*, Kyoto, Japan, Dec. 3-6, 2001