An Agent-mediated, Secure and Efficient Internet Marketplace

Yan Wang, Kian-Lee Tan, Jian Ren, Xiaolin Pang
Department of Computer Science
School of Computing
National University of Singapore
3 Science Drive 2, Singapore 117543,
Republic of Singapore
{ywang, tankl, renjian, pangxiao}@comp.nus.edu.sg

Abstract

In this paper, we present a framework of secure Internet marketplaces on the basis of software agents. 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. Based on the security mechanism and commercial credit assessment mechanism of the framework, a 2-phase model is also briefly discussed in this paper on how to evaluate e-shops before and after dispatching parallel mobile agents to them for the purpose of controlling the scale of mobile agents and satisfying the best-buy strategy of consumers. In addition, by using cryptographic technology, a parallel dispatch model of mobile agents with secure structure is also introduced to achieve both security and efficiency.

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 secure 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. In this framework, based on public-key encryption technique [4, 5] and X.509 authentication framework [6], a security mechanism is also set up and the commercial credit issue is also used to provide a healthy environment for consumers from the stage of asking quote prices. In addition, a parallel dispatch model of secure mobile agents is presented to achieve both security and efficiency when searching a large number of e-shops.

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.

Procs. of the 4th International Conference on Electronic Commerce Research (ICECR-4), Dallas, TX, USA, November 8 - 11, 2001, pp 649-661

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 [7]. 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 [7] gives a brief description on how to utilize the mobile agent technology.

Lange briefly introduced a mobile agent based marketplace architecture in [8] and showed that the Aglet Software Developing Kit (ASDK) system [9, 10] 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 [11].

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, Panayiotou and Wang all showed the advantages of applying the mobile agent approach to parallel processing over distributed databases or data sources [3, 12, 13, and 14]. 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, customers 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 [15]. 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 [16]. Therefore, building electronic marketplaces on the basis of mobile agents with a uniform framework for the market community is expected to be beneficial to customers to provide them with the best-buy trades more efficiently over lots of electronic shops.

In addition, when a mobile agent arrives at a host, which is a server for mobile agents, for execution, the code and data will be exposed to the host and the resources at the host may also be exposed to the mobile agent. Thus, security mechanisms should be set up to protect mobile agents from malicious hosts as well as to protect hosts from malicious agents. Some works have been done to protect the hosts, e.g., the access privilege protocol [17, 18] and the role based mechanism [19] restrict an agent's access to resources of a host to certain privileges assigned by the sender and confirmed by the host. Attempts to access other resources are regarded as attacks by the host. Protecting the agent is also a difficult task. In particular, in e-commerce environment, since e-shops are competitive, it is important to protect the routes and data of a mobile agent if it should visit a list of hosts (e-shops) or if it should dispatch other mobile agents to other hosts to accomplish the task. If a malicious host knows the route information, it may tamper with it so that its competitors that may offer better prices or services will not be visited. This calls for novel methods to be designed.

3 The Infrastructure for Internet Marketplaces

3.1 Overview

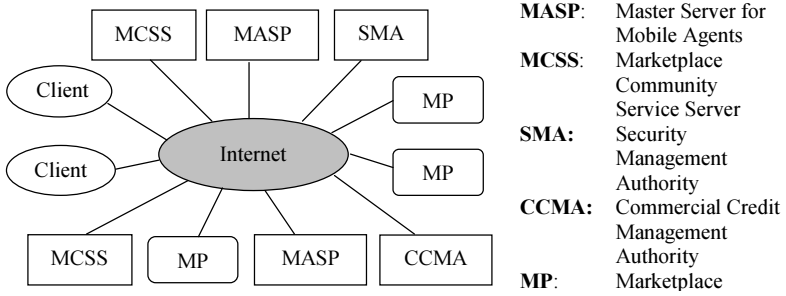

Figure 1 The Overview of the Marketplaces

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

3.2.1 MCSS (Marketplace Community Service Server)

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. 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.

3.2.2 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 consumers 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 3.4.

3.2.3 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.

3.2.4 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.

3.2.5 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 and 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.

3.2.6 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

- 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
- broadcasting the certificates of e-shops to other relevant sites, such as MASPs, and
- 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 MASP so that mobile agents can be dispatched from registered MASPs.

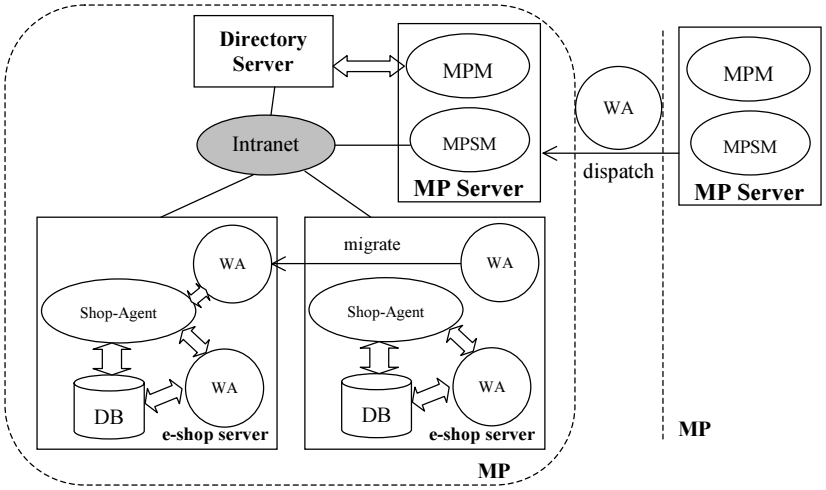

Figure 2 MP components

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

- maintaining the shop information and goods information stored in the shop database;
- periodically sending updated information to MPM for modifying the goods-catalogue of the e-shop maintained in the MPDS (MP Directory Server);
- communicating with incoming consumer-agents providing the goods information they required;
- monitoring the execution of foreign consumer-agents and protecting the local resources of the e-shop;
- registering the e-shop to the MPM and through it registering to the MCSS when the e-shop is set up;
- 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.

3.3 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 MCS when any e-shop withdraws or the whole MP withdraws.

3.4 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 chooses 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 agents, which are termed as worker agents (WA) and Primary Worker Agents (PWA), 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 in parallel following the secure parallel dispatch model which is presented in section 4 that can achieve both security and efficiency when a large number of mobile agents should be dispatched.

(3) Second phase evaluation

After the results are returned by all mobile agents, 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.

(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 [20]. 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.

In addition, during the whole process, based on our secure parallel dispatch model, when any attacks from malicious hosts are detected, the master agent at MASP will report to SMA and the security rank of corresponding e-shop will be degraded. This will cause the decrease of the number of mobile agents dispatched to the e-shop there since the first phase evaluation will be taken before any searching-agents to e-shops are dispatched.

3.5 2-Phase Evaluation

In our proposed framework, we can easily deploy mobile agents for parallel processing. When the master consumer-agent is created and running, it can get a list of e-shops from the MCSS that offer the goods that its client needs to buy. For the phase of searching and negotiating the consumer-agent can act as a master agent and dispatch multiple worker agents to these e-shops for querying goods' information, such as the stock status and the price. Each work agent is responsible for visiting one e-shop. Once it fulfils its task, it sends the results to the master consumer-agent.

If there are a huge number of e-shops (i.e. in thousands) providing the needed goods, to reduce the network load, we conduct a *two-phase evaluation process* at the side of master consumer-agent before and after dispatching worker agents. This process uses the principle of utility theory and fuzzy-set rules. The evaluation function of goods x is given over a set of domain specific attributes x_i as follows:

$$U(x) = \sum_i w_i * V_i(F_i(x_i))$$

where F_i is the grading function that calculates the firing level for attributes x_i. This grading function is attributes dependent, i.e. for price, the category of levels can be very good, good, moderate, poor and very poor. V_i is the score function that maps the attribute grading levels into interval [1,10] and w_i is the weight of attribute x_i. We assume the weights are normalized, i.e. $\sum w_i = 1$.

Before dispatching worker agents, the first phase evaluation is done only over two attributes. They are commercial credit and security rank. The commercial credit of an e-shop is set by CCMA based on the number of its previous successful transactions. The security raking is done by SMA according to the security history of an e-shop. The client should input the weight for each attribute and the selection criteria such as the number of e-shops to be searched or the lower limit for evaluation value. Those attributes for the first phase evaluation are stored in the MCSS with the goods types for all e-shops. After the first phase evaluation, worker agents will be dispatched to those qualified e-shops for searching in parallel.

On the second phase, when all worker agents send back addition details of the goods, such as the price, stock status, warranty service and expense, and delivery/shipment service and expense, the evaluation process will be taken again over all attributes and the sorted results will be presented to the end user. With his/her selection, the master agent can send new worker agents to a small set of visited e-shops to negotiate for lower price and/or more convenient services. According to the results, the end user will choose one e-shop for booking and payment.

More details can be found in paper [25].

4 Parallel Dispatch of Secure Mobile Agents

4.1 Binary Dispatch Model

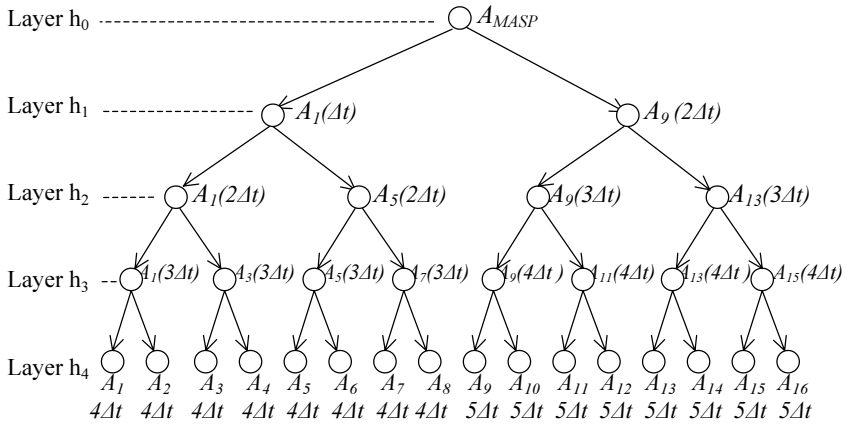

Figure 3 Dispatch Tree with 16 Mobile Agents

In this section, we introduce the proposed parallel dispatch model. For simplicity, we restrict our discussion to a *binary* dispatch model where an agent can dispatch two other agents resulting in a binary tree structure. Clearly, the model can be easily generalized to dispatch multiple (more than 2) agents. As shown in Figure 3, A_{MASP} is responsible for dispatching PWAs (Primary Worker Agent) and distributing tasks to them. A PWA is the special WA (Worker Agent) that should dispatch other mobile agents. A WA is only responsible for locally asking prices, accessing data and returning the result. A PWA can also have a task of performing data access depending on the application. Suppose A_{MASP} has to dispatch 16 agents to different hosts. Now, they can be divided into 2 groups led by two PWAs, say A_1 and A_9. When agents A_1 and A_9 are dispatched, each of them has 8 members including itself. For A_1, it will dispatch A_5 and distribute 4 members to it. Then A_1 will transit to the same layer (i.e., h_2) as A_5, which is called a virtual dispatch. But now A_1 has 4 members only. Following the same process, A_1 will dispatch A_3 and A_2. At last, after all dispatch tasks have been completed, A_1 will become a WA and can start its data-accessing task if it has. As a whole, since all PWAs are dispatched to different hosts, the dispatch process can be preformed in parallel. When there are $n=2^h$ mobile agents and Δt is the average time for dispatching a mobile agent, $(h+1)\Delta t$ will be the time for dispatching n mobile agents in the binary way. So, the dispatch time complexity will be $O(\log n)$. Thus, the proposed model is efficient.

There are three alternative implementations for a PWA to create and dispatch a child agent in the IBM Aglet system [10]. The first approach is that the MASP passes the child agent to the PWA who creates the child agent and encapsulates arguments such as the route and tasks and then dispatches it. This method is expected to be inefficient in a WAN environment. The second is to compress the framework of child agents to a .jar file and attach it to the PWA when it is dispatched. The child agent is created from the compressed file for being dispatched. The third one is to adopt the clone-like strategy. If some mobile agents have the same type of tasks, they can be put to the same group where a PWA can easily create a child agent by locally making a copy and modifying the static data. After encapsulating the route to the copy, the PWA can dispatch it to a remote host. The common feature for three alternatives is that initiation data can be encapsulated to the framework of an agent when it is created. A secure clone environment that provides security mechanisms to detect illegally forged agents is also an important issue that is a bit out of the scope of this paper. However, with our proposed EID structure and corresponding dispatch models, illegally forged agents and tamper attacks can be detected at the destination hosts. Here we address general-purpose secure dispatch models and do not restrict it to any implementation system.

4.2 Structure of Mobile Agent

In this part, we present the structure of searching agent that is dispatched from the master agent AMSAP.

The meaning of its components is explained as follows:

1. Agent Passport: It is the certificate of the sending MASP including its public key.
2. Encrypted Initiation Data (EID): EID is the data encapsulated to the agent when it is created for being dispatched. EID comprises the description of its tasks including dispatching tasks or data-accessing task or both. For a PWA, the EID includes the IP address of for its right child if it is a PWA and EIDs for the right child and left child (itself). The signature in EID signed by MASP can be used for checking integrity and preventing forgery and attacks.
3. Code: For a PWA and WA, the part of code is different. The code is also included in the signature of EID so that at any stage, forgery and tamper on the code will be detected.
4. Result: In different dispatch model, the structures of results are different. If a WA visits only one e-shop, the data is encrypted and sent to AMASP directly. For a serial migrating WA, it should put the results obtained from different e-shops together in good structure of cipher text.

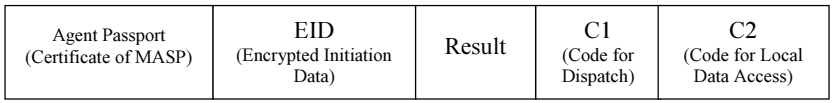

Figure 4 Structure of a Mobile Agent

The structure of mobile agent is similar to that proposed by [17, 18] which including Identifier (Agent Identifier, Creator-Certificate, Timestamp), Privilege_Token, Agent_code, Data_Store and Security_Tags. But we put the agent passport (i.e., the certificate of sending MASP) separately and the security_tag and Agent_ID are all put into the EID that is generated by AMASP. Agent privileges are put to the task encapsulated in the EID. The EID adopts the nested structure that is available for decryption at any dispatch layer and only necessary information is exposed to the host when EID is decrypted there

Several parallel dispatch models with different extensions and EID structures are presented in following sections.

4.3 Parallel Inter-MP Dispatch and Parallel Intra-Mp Dispatch (PD-PD Model)

In this model, all dispatch tasks are carried out in a parallel way, such as the binary dispatch. In this way, all PWAs are dispatched between any two MP-Servers without reaching any e-shops directly before reaching a MP-Server. To simplify, we restrict that a PWA has only dispatch tasks and will not visit any e-shop for asking prices. When a parent PWA arrives at the MP-Server, it can dispatch its child PWAs to other MP-Servers. If the inter-MP dispatch tasks have been completed, the PWA can dispatch 2 WAs to e-shops in current MP whereby the left dispatch is not a virtual dispatch any more. In this model, a WA is used for visiting one e-shop, asking prices only and sending back the result to the master agent at original MASP.

The EID structure for binary dispatch of this model is:

EID Structure (I):

(1) For a PWA at CH, $EID(CH)=P_{CH}$[PWA, Token, ip(RH), EID_L, EID_R, S_{MASP}[H(PWA, Token, ip(PH), ip(CH), ip(RH), EID_L, EID_R, Passport, Code, t)]], where Token equals to PWA

(2) For a PWA at CH, $EID(CH)=P_{CH}$[PWA, Token, ip(LH), ip(RH), EID_L, EID_R, S_{MASP}[H(PWA, Token, ip(PH), ip(CH), ip(LH), ip(RH), EID_L, EID_R, Passport, Code, t)]], where Token equals to WA

(3) For a WA at CH, $EID(CH)=P_{CH}$[WA, ip(PH), Task(CH), ip(MASP), S_{MASP}[H(WA, ip(PH), ip(CH), ip(MASP), Task(CH), Passport, Code, t)]]

In EID structure (I), EID(CH) denotes the Initiation Data of the agent at current MP-Server, CH, where the agent should go; ip(H) denotes the IP address of host H; RH and PH denote the right child's host and the parent host respectively; Token denotes the child agent is PWA or WA; EID_L and EID_R denote the encrypted EID for the left and right children respectively; P_{CH}[M] denotes the message M is encrypted by the public key, P_{CH}, of the current host CH; and S_{MASP}[H(D)] denotes the signature signed on the hash of document D by host MASP using hash function H and its secret key S_{MASP}. Passport is the one encapsulated in this agent. Code is the code of the agent for dispatch task or data access task. Task(CH) is the task description for this agent including the description of specified goods and its data access privileges that matches its code. And *t* is the timestamp at which the signature is generated. *t* is unique for all routes within a dispatch tree. The addresses of PH and CH only appear in the signature for verification. Token in the EID of a PWA shows the characteristics of its child agents. If Token equals to PWA, it shows the right child is a PWA and hereby the left dispatch is a virtual one. Otherwise, it should dispatch WAs to e-shops.

Starting the binary dispatch process, the agent A_{MASP} dispatches two PWAs to different hosts, each being encapsulated with an EID for future dispatch task. We call them the first left PWA (PWA_{1L}) and the first right PWA (PWA_{1R}). When an agent has successfully arrived at the current host CH, the

carried route EID(CH) can be decrypted with the secret key of CH so that the agent can know:

(1) it is a PWA or WA;
(2) the signature signed at host MASP that can be used for integrity verification

For a PWA, it will also know:

(1) the address ip(RH) of the right child host RH and whether it is a PWA or a WA;
(2) the encrypted initiation data EID_R for the right child agent, which can only be decrypted by the right child host;
(3) the encrypted initiation data EID_L for the left dispatch.

For a WA, it will know:

(1) the address of MASP, ip(MASP), the home host where A_{MASP} is residing. With this address, the WA can send its result to A_{MASP};
(2) task Task(CH) that should be completed at host CH;

Clearly, under this model, at any layer, only the address of the right child agent is exposed to the current host so that the right dispatch can be completed. For a PWA, if it has at most $m=2^k$ members, only k addresses of its members are exposed to the host.

In PD-PD model, for a PWA, the last dispatch is to dispatch two WAs to corresponding e-shops. When a WA is dispatched to an e-shop server CH, it will ask the information of the goods specified by T(CH) and then migrate to the MP-Server, PH, the host where it is dispatched from, and send back the result to agent A_{MASP}.

The algorithm of PD-PD model for dispatching agents in binary way is described as follows:

Algorithm 1: Binary dispatch with secure routes

Step 1: when an agent A is successfully dispatched to a host CH, it will use the secret key of CH, S_{CH}, to decrypt the carried initiation data EID(CH);

$$ID=S_{CH}[EID(CH)]$$

Step 2: if Token equals to WA, go to step 6, otherwise, A will dispatch another PWA to ip(RH), encapsulating the route EID_R to it;

Step 3: if the dispatch is successful, host RH will send a message including its signature to CH;

$$msg=S_{RH}[H(Entity_{RS}, ip(RH), t)] \quad (1)$$

where $Entity_{RS}$ is the full entity of the dispatched agent RS including its passport, code, and data. t is the timestamp when RH receives the agent successfully.

Once getting such a message, host CH will keep $S_{RH}[H(Entity_{RS}, ip(RH), t)]$ in its database as a successful dispatch record.

Step 4: Now A should try to complete its virtual left dispatch. Let $ID=S_{CH}[EID_L]$

Step 5: if Token still equals to PWA, go to step 2, otherwise go to step 6

Step 6: A dispatches a WA to ip(RH) encapsulating EID_R to it and dispatch another WA to ip(LH) encapsulating EID_L to it.

When a WA is dispatched to e-shop server CH, it will start its task Task(CH). Then it should send its result to A_{MASP}:

$$msg=P_{MASP}[ip(CH), Result(CH), SIG_{MASP}, t_2, S_{CH}[ip(CH), Result(CH), SIG_{MASP}, t_2]] \quad (2)$$

where SIG_{MASP} is the signature signed by MSAP and included in the EID of the WA. Here it is used for showing the identification of the agent. $SIG_{MASP}=S_{MASP}[H(WA, ip(PH), ip(CH), ip(MASP), Task(CH), Passport, Code, t_1)]$. Result(CH) is the result obtained at e-shop server CH and t_2 is the time when the result is obtained, $t_2>t_1$.

4.4 Parallel Inter-MP Dispatch and Serial Intra-MP Migration (PD-SM Model)

In the PD-PD model, if all dispatches are carried out in the binary way, 10PWAs will be dispatched to the same MP-Server if there 20 e-shops that should be visited and each PWA dispatches 2 WAs. This will overload the network traffic. An improvement can be done to let the PWA dispatch several WAs to e-shops within a MP. An alternative way is to complete all inter-MP dispatches in parallel and only one PWA is dispatched to a MP. When the PWA has completed its dispatch tasks, it migrates in the MP for visiting several e-shops and sends the result set to the master agent at MASP.

4.5 Serial Intra-MP Migration among the Threshold Number of e-Shops (PD-SM+ Model)

However, if the number of e-shops in a MP that should be visited is very limited (say 3 to 4), serial visit by migration is acceptable. Otherwise, several PWAs should be dispatched to the MP and each one visits a set of e-shops and the number of e-shops visited by one agent is not greater than a threshold. This will help to reduce the inter-MP network load caused by full parallel dispatch and will not significantly affect the whole efficiency.

The EID structure is described as follows:

EID Structure (II):

(1) For a PWA at CH, EID(CH)=P_{CH}[PWA, ip(RH), EID_L, EID_R, S_{MASP}[H(PWA, ip(PH), ip(CH), ip(RH), EID_L, EID_R, Passport, Code, t)]]

(2) For a WA, assuming the threshold is 3 and the e-shop servers are H1, H2 and H3 in sequence, EID(H_1)=P_{CH}[WA, ip(H_1), MIG, S_{MASP}[H(WA, ip(CH), ip(H_1), Mig, Passport, Code, t)], P_{H1}[WA, ip(H_2), T(H_1), S_{MASP}[H(WA, ip(H_1), ip(H_2), T(H_1), Passport, Code, t)], P_{H2}[WA, ip(H_3), T(H_2), S_{MASP}[H(WA, ip(H_2), ip(H_3), T(H_2), Passport, Code, t)], P_{H3}[WA, NULL, T(H_3), S_{MASP}[H(WA, NULL, T(H_3), Passport, Code, t)]]]]]

In EID structure (II), the EID structure for a PWA is similar to that of PD-PD model but it has no Token information since a PWA always dispatches others PWAs. For a WA's EID, the task at CH is to migrate to H1 which is shown by task token MIG. When a PWA has completed all its dispatch tasks, it will become a WA and migrate within the MP following the predefined sequence as H1, H2 and H3. When having obtained the information from H3, it will send the whole results back to agent A_{MASP}.

$$msg=P_{MASP}[Result(H_3), ER(H_2), SIG_{MASP}(H_3), t_3, S_{H3}[Result(H_3), ER(H_2), SIG_{MASP}(H_3), t_3] \quad (3)$$

where $ER(H_2)$ is the encrypted result obtained H_2, $ER(H_2)=P_{MASP}[Result(H_2), ER(H_1), SIG_{MASP}(H_2), t_2, S_{H2}[Result(H_2), ER(H_1), SIG_{MASP}(H_2), t_2]$, and $ER(H_1)=P_{MASP}[Result(H_1), SIG_{MASP}(H_1), t_3, S_{H1}[Result(H_1), SIG_{MASP}(H_1), t_1]$

For the above 3 models, the common principle of dispatch or migration is that a WA cannot be dispatched to or migrate to an e-shop directly from a host out of the MP. For PD-PD model, a WA is dispatched by a PWA from the MP-Server. For model PD-SM or PD-SM^+ model, a WA can migrate to an e-shop from the MP-Server or another e-shop in the same MP. At any e-shop server, a searching-agent can only ask the information of goods through the shop-agent or read data directly controlled by the privileges that are included in the task and confirmed by the shop-agent.

Here we only introduced the EID structures that at any host only one child host is known to dispatch a child agent to deploy the right branch. In [21], we presented the robust structures and mechanisms for the full binary dispatch model where several substitute host addresses can be obtained at any layer when the predefined child host is not reachable and hereby efficiency, robustness and security are all achieved.

4. 6 Resolving Security Threats

4.6.1 Protecting Agents from Malicious Hosts

In this section, we will examine several security issues that will be encountered when dispatching mobile agents and show how our PD-PD model resolves them. The rest two model can do the same.

(1) Preventing a PWA from Dispatching a Child Agent

During the period of dispatching a child agent, a malicious host may peek the code of the agent and make it skip the dispatch process in certain layer after the route is decrypted. Note that skipping a host would mean skipping all other addresses that may be triggered by that host. In the worst case, assuming host H_1 is the malicious one, as shown in Figure 1, if the dispatch of A_5 from H_1 is not in fact performed, those agents in the group including A_5 to A_8 will not be activated. This means the successful interception to the dispatch of a PWA will affect all members included in the aborted PWA. However this attack can be detected in this model.

Taking the case in Figure 3 as an example, if H_1 makes A_1 skip the process of dispatching agent A_5, agent A_{MASP} cannot receive any messages from each agent of A_5, A_6, A_7 or A_8. If this happens, since the four agents belong to the same group led by agent A_5, A_{MASP} will suspect first that A_5 may have not been dispatched. A_{MASP} will ask hosts H_1 and H_5 to show whether the predefined dispatch has been performed. Apparently, if the dispatch has been carried out, H_1 will receive the confirmation message with the signature $S_{H5}[(Entity_{A5}, ip(H_5), t)]$ from H_5. H_1 cannot forge this signature without H_5's secret key. So, no matter what H_1 claims, the attack can be detected.

If the skipped dispatch is for a WA, such as A_7 doesn't dispatch A_8, it can also be detected since H_7 cannot show a correct signature from H_8 to show the dispatch is successful.

(2) Route Skip Attack

There is yet another case that can be handled in this model. Consider a partial dispatch route: PWA A_i at host H_i dispatches A_j to H_j and A_j dispatches A_k to H_k, or there are more PWAs between A_i and A_k. In this model, the EID encapsulated to a PWA includes the encrypted route for its right child agent, which can only be decrypted at the child's host. That means when a PWA is dispatching an agent, it does not know how many members the agent has. So the case described above that A_i directly dispatches A_k is not likely to take place without the involvement of A_j. That is why the EID is in a nested structure. In the worst case, even if H_i can successfully predict that H_k is its descendent in the dispatch route and makes A_i dispatch a forged agent to H_k, the attack will not be successful either since the signature by MSAP encapsulated in EID clearly shows where the agent should come from and which host should be its destination. So, the forged agent will be detected by the destination host. Furthermore, the signature is also required to be included in the returned result

for the verification by A_{MASP}. So since forging the signature is impossible, this kind of attack cannot success.

(3) Tampering a PWA to Dispatch an Agent to a Wrong Host

Since the hosts are in a competitive situation, if a malicious host knows a host where an agent will be dispatched from it, and the remote host may probably offer a better service than itself, it may tamper the address so that the agent can be dispatched to another host which is known not to be able to provide a competitive offer. The tamper can be done just after the encrypted route is decrypted. However, when an agent is dispatched to a wrong host, its encrypted route will not be correctly decrypted there. Without the correct route, the verification process cannot be undertaken. Alternatively, even if the destination host can get the correctly decrypted route, the route will show that is a wrong destination since the address of the destination host is included in the signature in the route generated by MASP that cannot be tampered with. Thus, in both situations, the attack can be detected by the destination host and the agent will be returned to the sender. Meanwhile, this error will be recorded by the destination host for future investigation.

(4) Sending the result of a WA to A_{MASP} Directly or Not

In the PD-PD model, when a WA has fulfilled its data access task, it will send a message to A_{MASP} directly by encrypting the result, the signature by the host as well as the signature by the MASP originally included in the agent's EID. The structure is shown as message (2) in section 4.2. The whole message is encrypted with the public key of MASP so that it can only be decrypted by agent A_{MASP}. We choose this way in this model with regard to both security and performance issues. An alternative is that a PWA should be responsible for dispatching agents and collecting data from them. If PWA A_i dispatched PWA A_j which dispatched WA A_k and A_k encrypted its result with the public key of MASP and sent it to A_j where H_j cannot decrypt. To send the whole result set to A_i, A_j should encrypt its own result together with the encrypted result from A_k. If they are put as two separate encrypted results, deletion or tamper attacks may easily occur in the returning path especially when a large number of results are sent to a PWA. Meanwhile, this will increase the burden of a PWA and the performance will definitely become worse.

A possible solution preventing the results from being tampered or deleted that may take place at any host where a PWA resides is for the receiving side to send a reply to the sending side, just like the process for dispatching. The reply should be a signature generated on the received message by the secret key of the receiving side. In this way, deletion and tampering can be detected by the verification among the MASP, sending side and receiving side. However, the performance will become inferior.

In comparison, in our model, since a WA only visit one host, the host would not delete the result or prevent its offer from being returned once the agent has been successfully dispatched there. In case the attack occurs, based on the detection of successful dispatch, the problem should be with the side of the host where the agent has arrived. In terms of performance, since each WA has different starting time and ending time for the data accessing task and each offer will be in small size, the returned results can hardly cause the A_{MASP} to become a bottleneck.

(5) Replay Attack

In a malicious host, the replay attack may occur. Consider the following scenario, that a malicious H_i who has a PWA residing in it and it dispatched agent A_j to host H_j. After the normal process has been completed, H_i may replay the dispatch with a forged agent so that on one hand it can get the offer information from H_j constantly and periodically if H_i tampers the agent so that it sends the result to H_i, and on the other hand, excessive agents may jam H_j. However, when an agent is dispatched from H_i to H_j as a replay attack, the timestamp included in the signature from MASP cannot be tampered with. By verifying the signature, H_j can easily detect the replay attack and H_i will face the risk to be reported.

Similarly, another type of replay attack is for a host, which a WA had earlier resided, to repeatedly counterfeit the WA and send messages to the agent A_{MASP}. With the check of the timestamp in the signature, if A_{MASP} repeatedly receives offers from the same host, it will close the communication channel and start an investigation.

(6) Collusion Attack

If in a normal sequence, host H_a should dispatch an agent to H_b. Assuming H_a and H_c are in a collusion tie, the agent is dispatched to H_c. In this way H_a and H_c make an attempt to skip the visit to H_b who is their competitor and send their own offers instead. However H_c can hardly forge the signature by H_b that should be included in the message returned to A_{MASP}. In such a case, the counterfeited message can be detected when it is returned and this will cause the investigation against H_c and H_a. Since H_b will report that no such agent has ever been dispatched to it and H_a cannot show the correct dispatch record which should include the signature by H_b, the attack can be identified. The attack can be successful only when H_a, H_b and H_c make a collusion attack sending a result from H_b encapsulating the price from H_c. However, in a healthy competitive environment, the probability is fairly low. Even if it can take place, the future negotiation or buying agents will visit H_b not H_c and if H_b cannot offer the goods with the provided price, it will result in a commercial cheating, which is the same as a merchant's giving a nominal price and causing the abortion of the

purchase. This will cause the deduction of the merchant's commercial credit and little agents will be dispatched later to such merchants.

In addition, since the code information is included in the signature at any stage, attacks of tampering or forging an agent's code can be detected.

In the above discussion, we have already considered the situation that a PWA can dispatch other PWAs or WAs from an e-shop server directly. In the models proposed in this paper, we have restricted that a WA that visits an e-shop should be dispatched to an e-shop server from the MP-Server only. This provides a better framework for assuring security.

4.6.2 Protecting Hosts

In the framework, through the use of MPSM all foreign agents should undergo the security measurement. The MPSM should also be responsible for checking all messages passing through the MP and transfer them by security channels via cryptographic technology.

With the privilege mechanism proposed by [18], the resources of e-shops are protected and through the check on EID, forged agents and others attacks discussed above can be detected.

As a matter of fact, with the discussion in section 4.5.1, when forged agents or mis-dispatched agents are detected and these kinds of attacks are reported, those destination hosts have been protected.

4.7 Performance

[22] presented the methods for protecting the routes of a mobile agent that visits a number of e-shops by migration. Since the visit is serial, the migration time complexity is *O(n).* In comparison, as we have mentioned in section 4.1, the dispatch time complexity of binary dispatch model is *O(logn).* When the EID structure is used in each mobile agent, the time complexity is the same. For PD-SM+ model, though serial migration is used, the whole performance cannot be inferior since the serial migration is only restricted to a very small scope and it is helpful to reduce the code complexity.

5 Conclusions

This paper first 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 support consumers' commercial activities. The Java based mobile agent technology enables the features of light-weighted, portable and platform-independent clients. The architecture is suitable for both mobile and stationary users. 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, IBM Aglets 1.0.3 [9] and the SAX APIs (the Simple API for XML) from Sun Microsystems [23].

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 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.

With the cryptographic technique and EID structure, the security of both mobile agents and hosts are ensured without sacrificing the efficiency and most attacks can be detected at the destination host. Due the commercial cheating, it can be reported to CCMA by consumers. Combining the security ranking and commercial credit systems, and 2-phase evaluation model, consumers can be provided a healthier environment.

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 [24]. 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 access in the traditional way. Based on this, a new e-shop can be assigned an initial commercial credit value. Though at the beginning, its low commercial credit value may affect it to be included by some first phase evaluations, it can upgrade it through successful transactions with individual users.

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. And agent-oriented modeling and programming can give interesting solutions for some suitable application 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, 359-364 (1998).
[2] Rodrigo, T. D. and Stanski, A., "The Evolving Future of Agent-based Electronic Commerce", in **Electronic Commerce: Opportunity and Challenges**, eds. Rahman S. M. and Raisinghani M. S., Idea Group Publishing, Hershey, USA, 337-351 (2000).
[3] Panayionou, C., Samaras, G., Pitoura, E. and Evripidou, P., "Parallel Computing Using Java Mobile Agents", *Proceedings of 25th EUROMICRO Conference*, Milan, Italy, Volume 2, pp. 430-437 (1999).
[4] W. Diffie, W. and Hellman, M.E., "New direction in cryptography", *IEEE Trans. Information Theory,* vol. IT-22, no.6, 644-654 (1976).
[5] Wayner, P., **Digital Copyright Protection**, SP Professional, Boston, USA, (1997).
[6] CCITT, "Recommendation X. 509-1989. The Directory-Authentication Framework", Consultation Committee, International Telephone and Telegraph, International Telecommunication Union, Geneva, (1989).
[7] 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, 116-123 (1999).
[8] Lange, D., and Oshima, M. "Mobile Agents with java: The Aglet API", in **Mobility: Process, Computers, and Agents**, eds. Milojicic, D., Douglis, F. and Wheeler, R., Addison-Wesley Press, Reading, Massachusetts, USA, 495-512 (1999).
[9] http://www.trl.ibm.co.jp/aglets/.
[10] Lange, D., and Oshima, M. "Programming and Deploying Java Mobile Agents with Aglets", Addison-Wesley Press, Massachusetts, USA, (1998).
[11] Dasgupta, P., Narasimhan, N., Moser, L.E. and Melliar-Smith, P.M., "MAgNET: Mobile Agents for Networked Electronic Trading", *IEEE Transactions on Knowledge and Data Engineering,* vol: 11, no: 4, 509 –525 (1999).
[12] Silva, L. M., Batista, V., Martins, P. and Soares, G., "Using Mobile Agents for Parallel Processing", *Proceedings of International Symposium on Distributed Objects and Applications (DOA'99),* Edinburgh, Scotland, 34-42 (1999).
[13] 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, 228 –237 (1999).
[14] Wang Y., Law K. C. K. and Tan K. L. "A Mobile Agent based Protocol for Distributed Databases Access", *Proceedings of 2000 IEEE International Conference on Systems, Man, and Cybernetics (SMC'2000),* Nashville, Tennessee, USA, 2028 –2033 (2000).
[15] 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, (1999).
[16] 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, 223-228 (1998).
[17] Karjoth, G., Lange D. B., and Oshima M., "A Security Model for Aglets", *IEEE Internet Computing,* 68-77 (1997).
[18] Varadharajan, V., "Security enhanced mobile agents", *Proceedings of the 7th ACM conference on Computer and Communications Security*, Athens, Greece, 200 – 209 (2000).
[19] Ubayashi, N., Tamai, T., "RoleEP: Role based Evolutionary Programming for Cooperative Mobile Agent Applications", *Proceedings of International Symposium on Principles of Software Evolution,* 232 –240 (2000).
[20] 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, 70-90 (1998).

[21] Wang Y. and Tan K. L., "A Secure Model for the Parallel Dispatch of Mobile Agents", *Proc. of ICICS2001*, Springer-Verlag, LNCS Vol. 2229, (2001).
[22] Westhoff, D., "Methods for Protecting a Mobile Agent's Route", *Proceedings of the Second International Information Security Workshop (ISW'99),* Springer-Verlag, LNCS 1729, 57-71 (1999).
[23] http://developer.java.sun.com/developer/products/xml/docs/api/overview-summary.html.
[24] http://www.w3schools.com/site/site_intro.asp.
[25] Jian Ren, Yan Wang, Xiaolin Pang, Kian-Lee Tan, A 2-Phase Evaluation Model for Agent-mediated Internet Marketplaces, submitted to WISE2001, (2001).

Biography:

Yan Wang: Dr Wang received his Doctorate Degree of Engineering in computer science and technology in 1996 from Harbin Institute of Technology, China. He is currently a Postdoctoral Fellow/Research Fellow of the Department of Computer Science, School of Computing, National University of Singapore. His research interests cover mobile agent, electronic commerce and computer security.

Kian-Lee Tan: Dr Tan received his Ph.D. in computer science in 1994. He is currently an Associate Professor in the Department of Computer Science, School of Computing, National University of Singapore. His major research interests include multimedia information retrieval, wireless computing, query processing and optimization in multiprocessor and distributed systems, and database performance. He has published numerous papers in conferences such as SIGMOD, VLDB, ICDE and EDBT.

Jian Ren: Mr. Ren received his Bachelor Degree of Engineering in management information system from Tsinghua University, China in 1998. He is now a Master student of the Department of Computer Science, School of Computing, National University of Singapore.

Pang Xiaolin: Miss Pang received her Bachelor Degree of Engineering in computer science from Taiyuan University of Technology, China in 1997. She is now a Master student of the Department of Computer Science, School of Computing, National University of Singapore.