JAUES

Journal of Al Azhar University Engineering Sector
Vol. 3, No. 6, January 2008, 150-160

DIRECTORY AND MIGRATION SERVICES FOR MOBILE AGENT FRAMEWORK

Mohamed Khamis, Ibrahim El Bedwi and Ihab Sroogy.

ABSTRACT

Distributed programming based on mobile agent technology plays an important role of overcoming the limitations of connectivity and saving the bandwidth. Agent framework is needed to manage the different agent aspects during agent lifetime and to allow applications to create and access agents in transparent manner. Agent framework is a middleware which includes many services that allow agent to roam around network in order to achieve its task. These services include mainly the Directory, migration, management and security. In this paper algorithms for directory and migration services are described. The agent' name is made compatible with MASIF specification for sake of interoperability between different frameworks. The algorithms of these services are aimed to improve the performance as result of decreasing the message required to lookup for an agent and also to maintain the different databases when an agent is migrated.

KEYWORDS: Distributed System, Middleware, Location Protocol, Mobile Entities.

1. INTRODUCTION

Distributed systems based on client server paradigm are facing a real challenge especially for applications which, need to establish frequent connections and transferring a huge amount of data. To overcome these problems another paradigm that allows code to be moved closer to the data, to be processed at its location, is considered a good alternative. This emerging paradigm is called mobile agent [1]. Most of the existing mobile agent platforms [2][3][4][5][6][7] are implemented as middleware layer over the application layer of TCP protocol. These platforms present different services such as communication, naming, registration, migration, localization, management and security. The platforms should present these services for the applications in a transparent manner. Another merit for mobile agent paradigm it allows dynamic software update by disseminating agents to perform new functionality in the distributed applications. Many applications have found in the mob ile agent systems an attractive solution such as network management [8], distance learning [9], electronic commerce [10], distributed database and multimedia communication. This paper presents and describes modified algorithms for lookup and migration services. Naming is method that allows applications to locate the agents, naming has to satisfy certain specifications in order to be easily integrated with other platforms. Global naming schemes can be achieved by variety of ways [11] such as human friendly name, identifiers, and addresses. Whatever the used naming technique, the platform has to map the name to an

address for the current location of the agent. The chosen naming technique will affect the transparency and the efficiency of the system. The names are the starting point for locating agents. The good naming technique will simplify the process of locating agents. The name for an agent must be unique in its domain. The agent name has to encode information that makes reaching for the agent is possible. Agents could communicate directly by sending massages to each other or indirectly by allocating shared spaces where agents can meet and interact to benefit of the locality. In general there different trends to choose the agent name, one might chooses the name to reflect the functionality of the service presented by the agent, this is the way the internet servers are used. The different naming techniques are discussed regarding number of properties [12] they are:

- transparency: name is transparent if the agent' name does not contain information about the agent' site.
- Location independent which mean the agent name does not encode any information about agent current location.
- Selectability: is the freedom of the programmer to choose the names of the created agents.

2. RELATED WORKS

2. 1. Search-by-Path-Chase

In Search-by-Path-Chase technique [13][14], a human-readable naming scheme, which is compatible with MASIF specification [15], is proposed. The system is divided into regions, where each region name must be unique to impose that there are no two agents to take the same name. The agent name contains the agent name in addition to the region of birth. The region of birth provides the starting point for the location protocol. According to the proposed mechanism, in each region there are one or more sites acting as Agent Name Server (ANS); each one maintain agent location information into database denoted as Region Agent Register (RAR). On each site a database which known as Site Agent Register or SAR is found to maintain information about all transited agents.

The proposed naming and locating service involves three phases: registration, migration, and location. The registration phase starts when agent is created by registering the agent name in RAR of the region of birth. The migration phase starts when an agent moves from one site to another. When such movement is occurred both source and destination SAR must be updated to reflect the real agent location. In addition, the RAR of both source and destination region have to be updated. If the agent moves within the same region this mean that the source and the destination region are the same and only one update operation is required. The location phase starts when an agent needs to be located. The agent's region of birth has to be extracted from the agent name and then contact the relevant ANS. After that, check if the agents still in the region of birth otherwise the current region for the agent is stored.

2. 2. Cluster-Base Tracking Technique

Cluster-Base Tracking Technique [16] provides a way to track mobile agents. It is a combination of two general tracking mechanisms: Home-Proxy and Forward-Proxy. In the former mechanism, the mobile agent sends back a message directly to its home-site, while in the latter mechanism agent will leave a trace that indicates to its new location along the path. This technique tries to minimize the network traffic and keep bandwidth by reporting the update messages inside clusters, but leaving a trace when the agent traverse a long distance or migrate to another cluster. The number of hops will be used to measure the distance between sites and establish the clusters.

To explain how this mechanism works, suppose a mobile agent is created, the home-sit for this agent is assigned as a proxy for it. Whenever the agent moves within the same cluster, it

will report the home-site proxy about the new location. But, if the agent moves to a new cluster (i.e. moves for long distance across many networks), it will consider the first entered node as a proxy node in that cluster, and any movement inside the new cluster will notify the new proxy belonged to the new cluster. Any transmission to new cluster will cause changing the proxy for the agent. To bind all proxies with each other along the journey, the mobile agent leaves trace whenever it transfers to new cluster. The lookup operations start from the agent birthplace, where it uses the forwarding link at each proxy node to track the cluster along the journey and stop at proxy node that has no forwarding pointer. Eventually, it will find the mobile agent location by looking up last proxy node.

2. 3. Agent Shadow Tracing

Agent shadow tracing technique [17] introduces naming and localization service for mobile agent, where the functionality of naming service is distributed over the entire system. According to this technique, each host in which agents are generated is responsible to maintain the location information for those agents. Each host assign shadows for its agents; these shadows maintain the agent location information when the agents went away. If an agent migrates to another host, it sends an update message to its home host in order to update the location information. So, all communication with the agent is achieved through its shadow at its home-site.

The introduced naming scheme is made up of three parts; the name of its home-host, the name of current host, and the time of birth. So, this name is categorized as location dependent name, and it is unique in the system. Each agent has a human-readable alias, which describes its functionality, and all hosts in the system maintain a table that links the agent names with their aliases.

2. 4. A Scalable Hash-Based Mobile Agent Location Mechanism

The Scalable Hash-Based Mobile Agent Location [18] is a mechanism for tracking mobile agent, and based on hashing technique. According to this mechanism, there are three sort of agents constitutes the system: Information Agents (IAgents), Local Hash Agents (LHAgents), and Hash Agent (HAgent). The IAgents are mobile agents that keep the current location information of all mobile agents assigned to them. The location of such kind of agent changes over the time depending on the agents they serve. The LHAgents reside at each host in the system; each one maintains a local copy of the hash function. The hash function is represented by using a binary tree that is called hash tree. Finally, HAgent is mobile or static agent that maintains the primary copy of the hash function; also it may be mobile or static. Mobile agents are assigned to IAgent using a hash function, which take the binary representation of a mobile agent's id as input and returns the id and location of the IAgent that is responsible for this agent. The location information for all agents in the system is distributed among all IAgents in the system.

Once a mobile agent moves, it will communicate with LHAgent to find out the current location of the IAgent that is responsible for maintaining its location, and inform it to update the current location of the mobile agent. Later, in order to locate this agent the relevant IAgent is determined by using a hash function on LHAgent on the current site which uses the id of the agent as input. Eventually, the hash function will return the current location for the agent.

A dynamic rehashing approach is used to make this approach efficient and scalable. Dynamic rehashing means that the hash function changes when the number of IAgents changes whatever increased or decreased due to splitting or merging operations. An IAgent splits when the load on it is increased and exceeds a maximum threshold. The load is measured by the number of requests either for locating or updating an agent current location. On the other hand, two IAgents are merged when the number of requests for them fall bellow a minimum threshold. Such changes in the hash tree must be changed in the primary copy of the hash

function which is found in HAgent. If the lookup operation leads to wrong location due to the differences between the hash function in LHAgent and HAgent, LHAgent will update its hash function by contacting with HAgent on demand to save the network bandwidth and retry to lookup the agent again.

2. 5. dU-SSM Scheme

DU-SSM scheme [19] is an approach that allows mobile agents to send a location update message to its central management server once every d movements along its journey, which in turn enables the server to search for the agent from latest record sequentially. In traditional naming mechanism based on location update, when an agent starts its journey it sends update messages frequently to a central management server. The server updates the location database to accommodate the new location changes. At any time, if the central management server is asked for the location of the mobile agent, it will search the database and reply with the current location of the mobile agent. Indeed, using location update message with every transmission produces high update cost, however, it cause less searching message cost.

DU-SSM scheme is based on location update and optimizes it to minimize the bandwidth overhead due to the update messages. This means that, every d-number of movements, a mobile agent sends one message to update its location. The optimal threshold d makes balancing point between updating and searching. Eventually, the cost of location update and the cost of searching are formulated to compute the optimal threshold value of d.

3. DIRECTORY AND MIGRATION SERVICES

The directory and Naming Services are responsible for providing the naming and Lookup for agents on the local machine. It may consult a local name service or may be set up to pass requests to other existing name servers.

Distributed agents require a coordination mechanism to provide programmers with a set of coordination primitives. This requires a proper addressing scheme to identify agents (naming). But in distributed systems, such as internet, it is rather better to handle agents with names rather than identifiers. In the next section we will describe a way for naming and locating agent in the system. This requires distributed database to make the system scalable. These databases need to be continually modified as the agents move (migrate). Down here also an algorithm for updating these databases, as the agents migrate.

3. 1. Agent name Scheme

Agent name, to be globally identified, is made to compose of two parts. They are the agent_local name which is assigned by the agent owner and home_zone name separated by a period as follow:

Agent global name: Agent_local name . Home _zone name

This scheme is chosen to make the naming in the system compatible with MASIF specifications for mobile agent systems.

Here agent_local name is assigned to consists of a symbolic name assigned by the agent' owner and in order to guarantees uniqueness of the name, the name is compiled by the framework to concatenate machine address, time and date of creation to this symbolic name.

The directory service, adopted in this framework, is based on a hybrid of two of techniques. These two techniques are forward tracing and dedicated servers. In case of forward tracing, there is no update message is required since whenever agent to be located we start from its home and if the agent doesn't resides there it leave a reference to follow until the agent is found. As it might be clear, in most cases a long chain of references has to be followed until an agent is located, it means it will be slow to localize an agent. Also, as consequence all machines along this chain have to be operational and reliable all the time. Performance is affected by how much long the chain is. In the second method, a dedicated servers are used to

accept update message from agents when they are moved, hence one lookup message is needed to know the new agent' location. With each move for the agent an update message has to be sent to server to tell him about its new location.

The proposed solution for this problem is merging these two techniques to make trade-off between lookup and update messages, and to get most of the advantages of the above two techniques, this techniques is called Check-point Trace Mechanism (CTM). In this approach, the network is divided into zones. Zone is determined by a set of hosts which are geographically close to each other. Each zone takes a unique name (identifier) and each host in the zone will assign a unique name in its zone. Concatenating zone and host names results a unique host name on the entire network. A dedicated server is used for each zone (Check-point Server). This server is to update location information for the agents in its zone, and hence is used to locate agents according to algorithm explained down here.

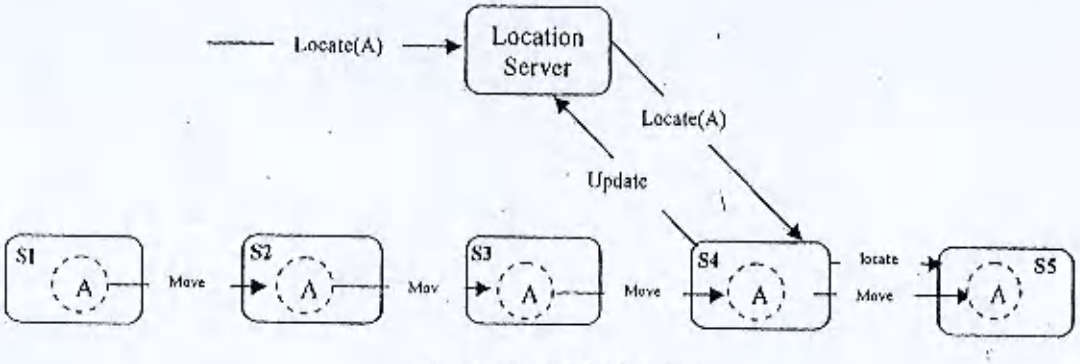

Fig. 1. The CTM Concept.

The location server for each zone will be registered at well known location. Figure 1, shows the concept of CTM approach, when an agent A moves from site S1 to site S2 it leaves a reference for its new location in S1. After a predefined number of moves it sends an update message with its current location to the location server, so its previous chain of references will be obsolete. When a client looks for agent, it starts at checkpoint server of the zone to find the reference for its location when, the last update message is sent. Lookup message to the host referred by this reference is needed to communicate with that agent if the agent is still there, otherwise the agent will leave behind a chain of references which, has to be followed to locate the agent.

To explain the idea, consider the agent A in figure 1, if the agent moved from its home' site, say S1, to site S5. According to the forward tracing approach, 0 update message and 4 lookup messages are needed to locate the agent. If checkpoint-Trace is used and we have agreed upon sending update message by the agent to the location server every 3 moves, then 1 update message will be needed and 2 lookup messages are required to locate that agent as shown in figure1.

Dividing a network of hosts into zones is to facilitate the management, to minimize the communication overhead, enhancing the overall performance and to make the system scalable one by distributed the services rather than making it central. The distance between hosts is measured by the number of hops required to send update message when the agent migrate from site to other, or to send lookup message to locate particular agent. The number of hops is equal to the number of routers the message encounters, i.e. the distance between two hosts in the same LAN is 0 hop while the distance between two hosts in adjacent LAN is 1 hop … etc.

In General the distance is measured by the number of router crossed by the sending messages. The idea beside this is the processing overhead taken by the routers, which is considered the most significant factor for the time delay. In each zone there is at least one server dedicated to maintain the current location for the agents initiated in this zone called Zone Location Server (ZLS), which plays two significant roles, first it maintains the agent location information and the second it accumulates undesired traces for all agents in the same zone to help the garbage collector. ZLS contains a record for each agent initiated in its zone called Agent Location Record (ALR). ALR hosts different information such as agent name, host of birth, current zone and the last checkpoint.

Beside the database of the server there are another two databases on each host in the zone the first called Home Agent Record (HAR) and the second called the Transitive Agent Record (TAR). The HAR holds information about all agents that are created at the current site. The TAR contains information about agent migrated across the host under concern. This later database includes information such as agent name, zone, time stamp and the host that holds the useless trace. Useless_trace is the_redundant part in the agent trace chain which might occurs as result of cycles (i.e. agent moves from a specific site and return to the same site afterward, which in turn will result a redundant part in this agent, chain which is not necessary to locate this agent).

When an agent leaves a site it leaves its reference in TAR at this site. This information is necessary to locate an agent during the lookup stage. Normally to refer to an agent we need a global name, in this paper a global agent name is taken as composition of the local agent name and agent home zone separated by a period. Zones name is unique in the whole network.

The agent framework in this paper presents services of location update and agent lookup algorithms. Explanations for these services are explained down here.

3. 2. Agent Location Update

When agent moves, it sends an update message to ZLS. The performance improving is measured by decreasing the number of update messages. The complexity of these messages is not measured only by their number but also by the distance measured by number of hops as mentioned above. Two types of agent moves are recognized. The first is intra-zone migration when agent moves from site to another inside the same zone and the second is called inter-zone migration which occurs when the agent move from one site to another in different zone. The update message for the second type is more costly. Consequently it is necessary to decrease the no number of inter-zone messages.

Agent Location Update Algorithm (ALUA) is proceeding as follow:

1. When the agent is created, it considers the host created at, is its home site and the current zone its home-zone.
2. The home host sends a message to ZLS of current zone to notify it about the new agent and create a record for this agent in ALR database.
3. When the message is received by ZLS, it creates new record for this agent in ALR database. This record will be used later for locating the agent according to the lookup algorithm explained later.
4. when the agent migrates to other site, the migration has two possibilities, they are:
    a. Migration to another host in the same zone. In this case the sender host sends the agent and maintain a trace record for this agent in TAR database.
    b. Migration to another host in the different zone. In this case the receiver host sends an update message to ZLS of the agent's home-zone of this agent to update the ALR there.

Database will inflate unless reclamations for these databases take place periodically. As might be noticed, all hosts maintain records for all agents passing through them, and they keeping those records even after agents are being destroyed. Of course, this situation will affect the scalability and performance of the whole system. Hence, the following modifications have to be considered to solve the previous problems:

1. Creating new database called Useless Trace Record (UTR) at ZLS to record all superfluous references for all hosts in its zone. To use this database the agent has to carry with it a chain of the ID's for all of the visited sites after the previous location update message for this agent. If the current host of agent sends other location update message to update the agent location according to predefined location update policy, it sends in the same time Useless Trace Message (UTM) consists of all previous visited sites by this agent, in addition to the agent name itself. ZLS keeps a record for each host consists of linked list of all superfluous references at this specific host. The erase for superfluous references at specific host takes place after its linked list of references at ZLS reaching a predefined threshold. This is to save communication cost by sending single message with multiple references to be deleted instead of sending separate message for each reference, hence saving communication band width.

2. Removing cyclic references: this occurs when an agent visits specific site twice during its journey. In this regard one of two situation occurs:

   a. Agent A_1 visits site S_1 and continues its journey to other sites, then afterward it returns to S_1 again and all visited sites since the previous visit to S_1 are in the same zone. In such case, all sites references for S_1 and all the references in between have to be removed and replaced by single reference, which is the reference of S_1. Otherwise, when the location update takes place, useless trace message will be send to ZLS contains all of these references (supposed to be removed) hence, many useless trace messages to different hosts have to be send for all of these hosts by ZLS to remove the reference for this agent at those hosts, i.e. update will have a high cost.

   b. Agent A_1 visits site S_1 and continues its journey to other sites, then afterward it returns to S_1 but in between the agent visited host/hosts in other zone. In such case a new reference for visited site S_1 is added to the list of references, which is accompanied with the agent.

3.3. Agent Migration Scenario

Figure 2 shows two zones and agent created at site A of zone Z1. In this example the agent migrates between 8 sites distributed in 2 zones.

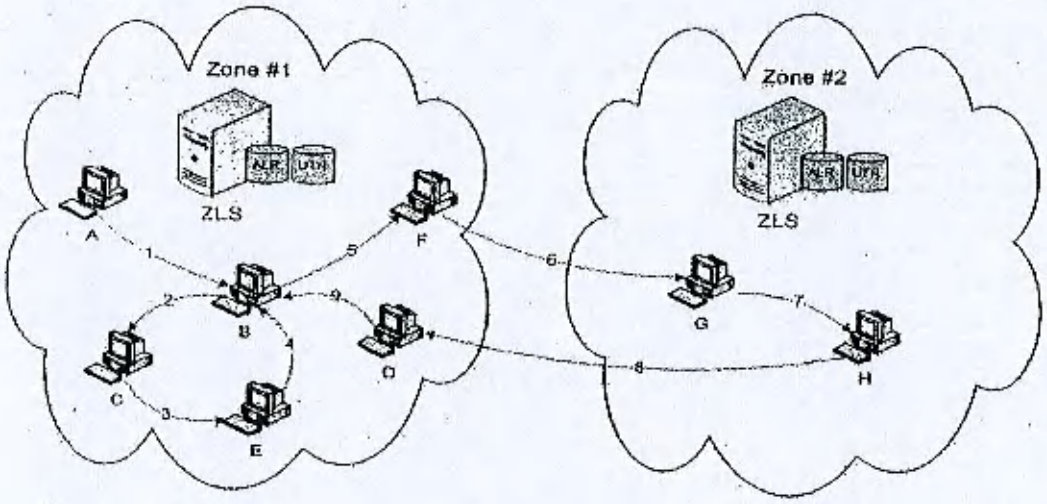

Fig. 2. Agent migration scenario.

Initially, the agent is created at site A, so it considers site A is its home-site and Z_1 is its home-zone. After that, site A sends a message to the ZLS of Z_1 in order to create a location record in ALR for the agent. After step 4, the agent visits site B for the next time without leaving the current zone, so site B consider all visited sites between the first and the second visit for site B have a superfluous traces (i.e. sites C and E in this example) and consequently site B sends a useless trace message, which contains the sites that have the superfluous traces for the agent to the ZLS of the current zone.

The ZLS adds the information in the received message to UTR. At step 6, the agent leaves the current zone Z_1 going to Z_2 reaching site G. at this point the site F sends useless trace massage to the ZLS of Z_1, this message includes site A and B. At receiver side, site G sends location update message to the ZLS of Z_1 in order to update ALR with the agent new location. This message is considered as inter-zone message that pass long distance and cause more delay. At step 8, the agent transmits to site D leaving Z_2 and reaching Z_1 again.

At sender side, host H sends a useless trace message to the ZLS of Z_2, this message updates the information in UTR by adding the agent to the queue of site G. at receiver side, host D sends location update message to ZLS of the agent home-zone to update the agent's last checkpoint to host D. after step 9, the agent visits host B for the third time, but in this time the agent transmit to another zone, so if the agent left host B going to other host, it leave another trace at site B with the new T_Z to be different from the previous trace (if it is still exists at ZLS). The messages yielded, by this scenario, is depicted using interaction diagram in Figure 3.

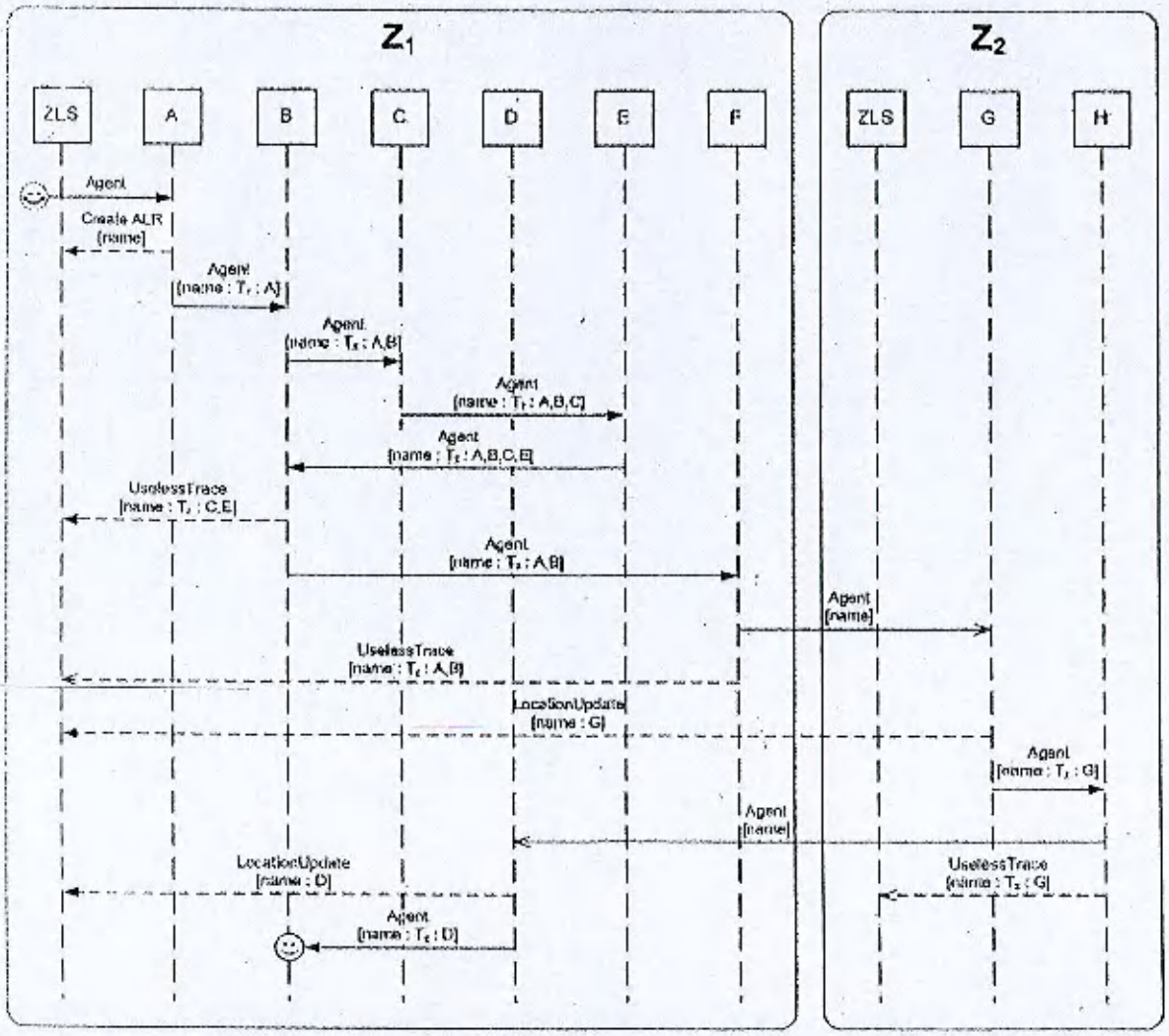

Fig. 3. Interaction Diagram.

In this figure the squares represent location server and hosts, the arrows represent the direction of message passing in the system, solid arrow refers to message in which the agent migrates and the dotted arrows represent message produced according to the location update algorithm and useless trace.

3. 4. Agent Lookup Algorithm (ALA)

Is the way of determining the location of agents after they have been launched across network, when the owner of agent needs to know its location to communication for reasons such as coordination of the agent with other agents, terminate the agent or to monitor its state etc… .

Figure 4 explains the steps used to locate an agent for application running at host A. The algorithm goes as follow:

Initially host A extract the agent home zone from the agent' name and is used to inquire the agent location at ZLS (this is the first message), when the message is received at ZLS it checks its ALR database to get last checkpoint of that agent, then it send message (second message) to the host of the registered checkpoint. If the check point is the current agent location, then a reply message (3-rd message) is sent back to host A to inform it about agent current location. Otherwise a trace for chain of references has to be followed until the agent current location is found.

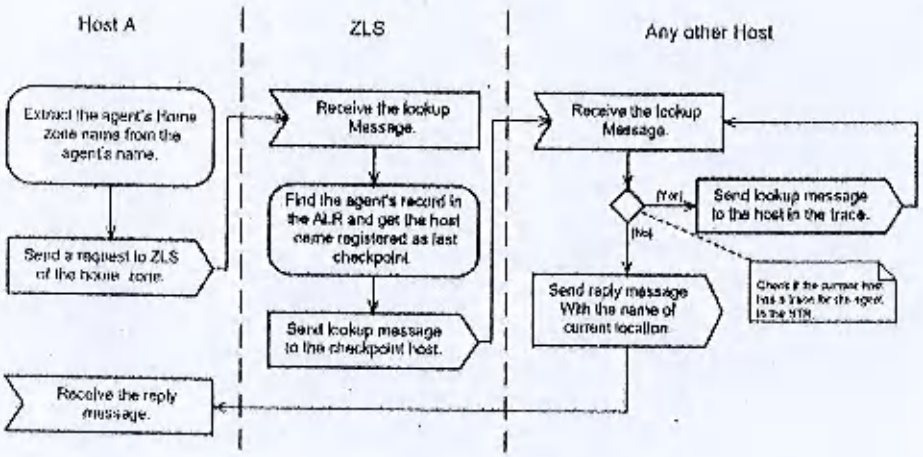

Fig. 4. Agent Lookup Algorithm.

4. Conclusion

Mobile agent technology is an alternative for building distributed applications. This technology is suitable for applications, which need transmission of data of size bigger than the size of the code necessary for processing the data at its location. This paper presented and described two algorithms for two of the basic services of agent platform. These services are the naming and migration services. The naming service introduced a mechanism which is compatible with MASIF specification. This technique is a combination of two well-known mechanisms, which are forward trace and dedicated name server. A trade-off is made to optimize the number of messages needed to look up for an agent in the system. Also the paper consider the locality of agents and divide the network into regions and assign name server for each region in order to make the system a scalable one. Also detecting and removing the cycle in the references (if happened) as result of visiting an agent the same site twice according to its itinerary. This improves and speedup the process of locating the agent in the system. The process of reclaiming the database is done according to a predefined threshold in order to reduce the number of messages exchanged between the ALS and the agent home so it save the bandwidth.

REFERENCES

1. Vu Anh Pham, Ahmed Karmouch, "Mobile Software Agents: An Overview", IEEE Communication magazine, July, 1998.
2. http://www.trl.ibm.com/aglets.
3. http://www.cs.dartmouth.edu/~agent.
4. http://www.merl.com/projects/concordia
5. http://www.cs.uit.no/DOS/Tacoma/index.html/index.html.
6. F. Hohl, M. Straßer, K. Rothermel, 1997, "Mole - Concepts of a Mobile Agent System", University of Stuttgart, Institute for Parallel and Distributed High-Performance Computers.
7. C. Baumer, M. Breugst, S. Choy, T. Magedanz, 2000, "GRASSHOPPER – A UNIVERSAL AGENT PLATFORM BASED ON OMG MASIF AND FIPA STANDARDS", Proceedings of the first international workshop of Mobile Agents for Telecommunication Applications (MATA).
8. Huawen Luo, "Agent-based Network Management System", Master Thesis-Department of Computer Science, The University of British Columbia, 2002.
9. Yugyung Lee, Quddus Chong, "Multi-agent systems support for community-based Learning", Elsvier - Interacting with Computer 15(2003) 33-55.

10. S. Han, E. Chang, and T. Dillon, 2005, "Secure e-Transactions Using Mobile Agents with Agent Broker", International Conferences on Services Systems and Services Management (ICSSSM), IEEE Press.
11. Andrew S. Tanenbaum, Marten van Steen "Distributed Systems: Principles and Paradigms," Prentice Hall; 1-st edition (January 15, 2002).
12. Antonella Di Stefano, Corrado Santoro, "Locating Mobile Agents In awide Distributed Environment" IEEE Transaction on Parallel and Distributed Systems, Vol. 13, No. 8, August 2002, pp 844-864.
13. A. Di Stefano, L. Lo Bello, and C. Santoro, 1999, "Naming and Locating Mobile Agents in an Internet Environment", Proc. Tird int'l Conf. Enterprise Distributed Objects (EDOC '99).
14. A. Di Stefano, and C. Santoro, 2002, "Locating Mobile Agents in a Wide Distributed Environment", IEEE Transaction on Parallel and Distributed Systems.
15. http://www.omg.org
16. Ting-Yuan Yeh, Tzone I. Wang, 2005, "A Mechanism for Tracking Mobile Agents in a Cluster Topology", IEEE Proceedings of the 2005 11th International Conference on Parallel and Distributed Systems (ICPADS'05).
17. Zhong Shi-qiang, Shi Zhong-zhi, Tian Qi-jia, 2001, "AST – a Method of Agent Naming and Localization", IEEE.
18. G. Kastidou, E. Pitoura, and G. Samaras, 2003, "A Scalable Hash-Based Mobile Agent Location Mechanism", IEEE Proceedings of the 23 rd International Conference on Distributed Computing Systems Workshops (ICDCSW'03).
19. Tie-Yan Li, Kwok-Yan Lam 2002, "An Optimal Location Update and Searching Algorithm for Tracking Mobile Agent", ACM.