Inter-Cluster Service Lookup Based on Jini

Wen-Hsien Tseng and Hsing Mei
Web Computing Lab., Computer Science and Information Engineering Department,
Fu Jen Catholic University
tony88@csie.fju.edu.tw and mei@csie.fju.edu.tw

Abstract

Jini's discovery protocols allow people to access requested services spontaneously in their own federations. However, to access services in remote federations, people have to know where the remote lookup services of them are located. This prerequisite is owing to Jini's discovery protocols relying on multicasting support, which becomes an issue when people want to access services across non-multicastable federations. Jini system, moreover, cannot handle the mobility. When mobile clients join into a new federation, they have to relocate services, which were installed in the initiating federations. All these problems are consequent on lack of communication between federations in the current Jini system.

In this paper, we propose two new schemes to extend the Jini lookup service: one is called Inter-Federation Communication; and the other is called Proxy-Object Forwarding. In the first scheme, we construct an extended lookup service in charge of the inter-federation service lookup. In the second one, we predict the federations that mobile clients will move to, and forward the requested proxy-objects to the predicted federations. With the effort of this paper, the performance of the Jini system is enhanced. Our experiments show the significant improvement in response time with Inter-Federation Communication and Proxy-Object Forwarding.

1. Introduction

Today people can get the information and services as long as they know where the resources are located on the network. Furthermore, people can access network resources wherever they are through wireless devices such as PDAs, mobile phones… etc. A sentence which best grasps this situation is the slogan used by Sun, "the network is the computer". In order to achieve its ideal, Sun developed the Jini [1] technology to construct a flexible and spontaneous network environment. Although Jini technology allows people obtain the resources easily, there are still problems of inter-cluster communication [2, 3] in the Jini system. The word "federation", the Jini terminology, will be used to represent the word "cluster" in the rest of this paper. Hence, we propose the Inter-Federation Communication Scheme for the extended Jini lookup service to achieve the resource-sharing between federations.

Moreover, people can change their federation when they are using these services. Current Jini specification does not thoroughly consider the mobility issue. Hence, when a client joins into a new federation, it has to find services again by means of the original Jini discovery protocols. This wastes lots of network bandwidth and the client must know when it joins into a new federation, so this means a heavy workload on the client side. Hence, we propose the Proxy Object Forwarding Scheme, an object caching mechanism [4, 5, 6], allowing a mobile client to dynamically find available services through the nearest lookup services, and not increase the client's workload.

The rest of this paper is structured as follows: In Section 2, we introduce Sun's Jini technology and problems in the Jini environment, and list possible solutions that can solve the resource-sharing problem. In Section 3, we introduce the Inter-Federation Communication Scheme, and then bring up the Proxy Object Forwarding Scheme. In Section 4, we describe the architecture of our system with using two proposed schemes, and then analyze our system. In Section 5, we show our experiment results. Finally, Section 6 puts forward our conclusions and ideas for future development.

2. Background

The Jini Technology is a simple set of distributed network protocols proposed by Sun Microsystems, Inc. The goal of Jini is to construct a spontaneous and self-healing distributed system based on the idea of federating groups of users, resources, and devices and software components, required by those users. The Jini technology provides simple mechanisms that enable services and users join into and detach from a federation without any prepared planning or human intervention. Jini federations are far more dynamic than any currently available networked group mechanism where configuring a network is a centralized function done by hand.

In the Jini system, members communicate with each other by using Java Remote Method Invocation (RMI) [7] mechanism. RMI allows not only data but also entire objects to be passed from node to node around the Jini system. When clients (even services themselves) require some service, they need only the proxy object of the service to communicate with the service.

2.1 Problems to be solved in Jini

Many researches focus on using the Jini technology to solve some specific distributed application or solving the Jini surrogate problems. However, the focus of our paper is on the inter-federation communication mechanism of Jini system. In this subsection, we describe two main problems in the Jini environment.

1) *Lack of Communication between Federations* [8, 9]:

In the Jini discovery protocols, clients can obtain resources from nearby lookup services or remotely through known lookup services only. Sun's original Jini infrastructure does not provide the communication mechanism between lookup services so that lookup services are unable to share the resource information in their federation with others. Hence, people can not have access to potentially useful resources even these resources are in a nearby federation. Fig. 1 illustrates this problem.

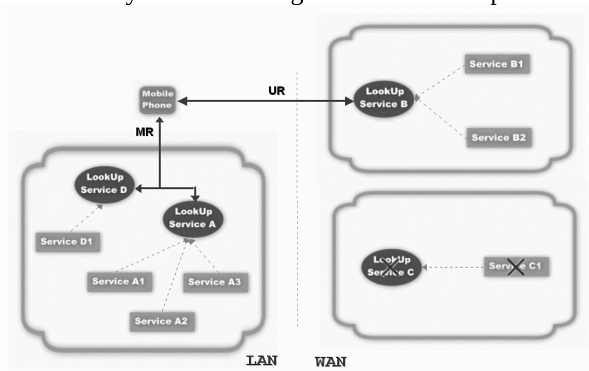

Fig. 1. -- Lack of Communication between Federations.

2) *Unsuitable for the Mobile Environment*:

Consider the following situation: A client joins into a federation and uses the Jini discovery protocols to find some available services. After a while, the client moves into a new federation. Can the response of the previous request in the original federation be sent to the new federation where the client is now? The answer from current Jini system is 'no', because the discovery protocols of the current Jini implementation do not consider the mobile issues yet. The client needs to use the Jini discovery protocols to find services again, wasting considerable network bandwidth and increasing the workload of the client.

2.2 Possible Solutions

Below, we list some possible solutions that enhance Jini federations with the ability to communicate with others.

1) *The multicast packet can be sent everywhere*:

The Jini multicast request protocol only allows a client to find available services within its own federation. This is due to the IP multicast radius. By adjusting this parameter, multicast packets can be sent everywhere and a Jini network can extend its reach. Although this method is simple and straightforward, it wastes the network bandwidth and needs support from the underlying IP network.

2) *Peer Lookup*:

The Jini peer lookup protocol allows clients to access the Jini network without lookup services. This manner not only increases the workload and the storage of clients, but also wastes the network bandwidth because this manner still needs the multicast announcement packets to be sent worldwide even though clients know where the requested services are located.

3) *Redesign the Jini discovery protocols*:

Redesigning the Jini discovery protocols seems to be a good manner in witch to help federations communicate with each other. However, Jini technology is stress on providing a set of simple network protocols; it need not solve such complicated problems in the infrastructure layer.

4) *Hierarchical Federation Architecture*:

The "Hierarchical Federation Architecture" [10] manner is a centralized management system like the architecture of DNS [11, 12, 13]. This centralized mechanism operates as following: A lookup service in the local federation registering with another lookup service in the remote federation makes the local lookup service available to the remote federation. This mechanism fits in with the policy "dealing with such problems in the application layer", but it is not flexible and needs lots of administration.

3. Inter-Federation Lookup Service

In this section, we will show two of our proposed schemes: one is "Inter-Federation Communication Scheme" that provides a communication scheme between federations to solve the above-mentioned problem; the other is "Proxy Object Forwarding" that provides an efficient way to find services in the mobile environment to solve the "Unsuitable for the Mobile Environment" problem mentioned in Section 2.

3.1 Inter-Federation Communication Scheme

In order to achieve the resource sharing between federations, we proposed the Inter-Federation Communication Scheme for the Jini lookup service. The extended-Jini lookup services communicate with others by this scheme, but not all lookup services in a federation only some specific lookup service called the extended-lookup services need the communication ability. Hence, lookup services in one federation can be divided into two parts: one is the original lookup service that deals with the original Jini discovery process; the other is the extended-lookup service that deals with requesting available services from the neighbor federations. Fig. 2 shows the architecture of this scheme.

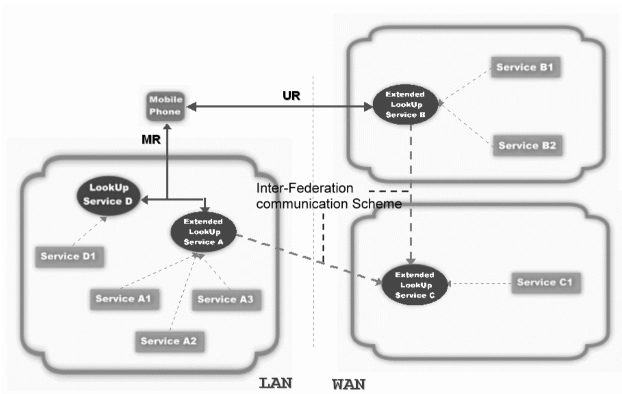

Fig. 2. -- Architecture of Inter-Federation Communication Scheme.

Because the Jini system is distributed, we use the neighbor list [14, 15] in which the extended-lookup service records the IP address of its neighbor to construct our system instead of using the centralized manner. Through the neighbor list, we can make our system flexible and easily administrated.

The operations of the Inter-Federation Communication Scheme are as follows:

The client side: Use the original Jini discovery protocols to find available service. If the client does not receive any available service form nearby or remote original lookup services, it sends its request to the extended-lookup service in its federation then wait for a response.

The original lookup service side: Use the original Jini protocols to deal with requests.

The extended-lookup service side: When the extended-lookup service receives requests from clients, firstly it portrays the original Jini lookup service to deal with these requests. If it can provide available services, it sends the proxy objects of services that clients want back to them or the initiating extended-lookup service. Otherwise, it changes its role into the extended-lookup service, and then resends these requests to neighbor extended-lookup services from its neighbor list in a time period to await a response.

The service side: Use the original Jini discovery protocols to register its proxy object to local lookup services or remote ones and interact with clients using the proxy object.

The time period of re-sending the inter-federation communication request is a key factor of our system. If the time period is too short, it will waste the network bandwidth and impact on the performance and throughput of extended-lookup services. If the time period is too long, it will increase the response time of clients. In Section 4, we will illustrate why the value of this parameter influences the performance and throughput of our system and how to define the optimumvalue of this parameter.

3.2 Proxy Object Forwarding Scheme

The purpose of the "Proxy Object Forwarding" mechanism is to make a mobile client can dynamically find available services through their nearest lookup services at that time and not increase the workload of the mobile client. The operations of this mechanism are listed below:

The extended-lookup service side: The initiating extended-lookup service forwards the proxy object to the predicted federation [16, 17] where the mobile client will likely be located.

The mobile client side: If the mobile client does not change its federation, it waits for responses. If the mobile client changes its federation to our predicted federation, it can obtain the proxy object from the nearby extended-lookup service by the "Proxy Object Forwarding" mechanism (shown in Fig. 3). Otherwise, it uses the re-sending requests mechanism in the "Inter-Federation Communication Scheme" to obtain its requested services.

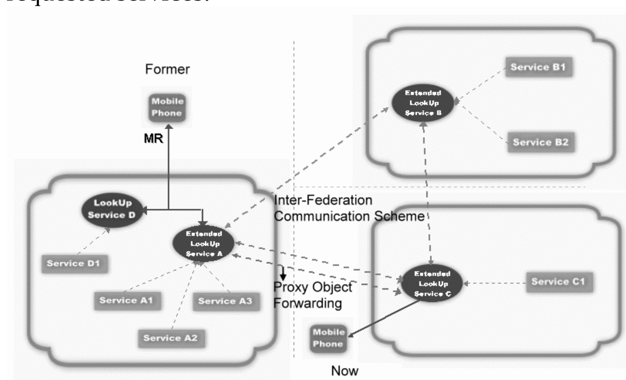

Fig. 3. -- The Mobile Client Changes his Federation.

4. Extended-Jini System

This section presents an implementation of our extended Jini system and the algebraic analysis for this system. We begin with the illustration of the architecture in detail. The operation flow of the extended-Jini system can be divided into two parts:

1) *Intra-Federation Operations*:

- Clients use the original Jini multicast request protocol to find their requested services from original lookup services or extended-Jini lookup service locally. If the services are found, then send back the proxy object to the initiator of the process, and the process is complete. Otherwise clients send requests to the local extended-Jini lookup service.
- The local extended-Jini lookup services also use the original Jini multicast request protocol to find services that initiating clients want from original lookup services locally.

2) *Inter-Federation Operations*:

- The extended-Jini lookup services use the Inter-Federation Communication Scheme to request from remotely extended-Jini lookup services in their neighbor list.

- If the remote extended-Jini lookup services have the requested service, then send back the proxy object back to the initiator and end this process.
- The process of our system performs circularly every time period and end when the number of loop iterations equals to 7.
- When the initiating extended-Jini lookup service receives the proxy objects that clients want, it forwards these proxy objects to their neighbor extended-Jini lookup services by the Proxy Object Forwarding Scheme.

In the following, we provide algebraic expressions for this implementation system. Firstly we define some influence factors of our system (Table. 1).

Table. 1. -- The Parameter List.

CT_{EiEj}	Communication time between the i_{th} depth extended-lookup service and the j_{th} depth extended-lookup service
CT_{EiO}	Communication time between the i_{th} depth extended-lookup service and the original lookup service in the same federation
CT_{EE}	Communication time between two extended-lookup services
CT_{EO}	Communication time between the extended-lookup service and the original lookup service
ST_E	Searching time that the extended-lookup service spends on searching its registered services
ST_O	Searching time that the original lookup service spends on searching its registered services
RT_C	Response time that a client spends between sending a request and receiving a response
T_{Period}	The time period of resending request
D	The depth of the federations that the required service exists
N_S	Average number of registered services in the lookup service

The analysis can be divided into two parts:

System with Static Clients Only

We focus on a static client environment, in which a client requests available services and obtains proxy objects of these services in the same federation. The tree structure can be used to represent our system (shown in Fig. 4). The root node is the initiative federation from where a client sends a request. The nodes in the first depth are the neighbors of the initiative federation, and so on.

The response time of a client can be divided into 4 parts:

1) The communication time that a client sends a request to the extended-Jini lookup service in his federation. It equals to CT_{CE}

2) The total sum of the communication time between two extended-Jini lookup services, searching time of each extended-Jini lookup service, the communication time between the extended-Jini lookup service and the original Jini lookup service and the searching time of the original Jini lookup service from the first depth to the $(n-1)_{th}$ depth. It simplifies to $(D-1)(CT_{EE}+ST_E+CT_{EO}+ST_O)$

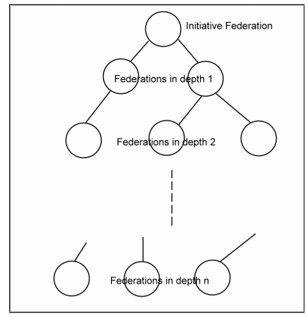

Fig. 4. -- The Topology of the Extended-Jini System.

3) The third part can be counted in two cases

Case 1: the requested service which is found in the extended-Jini lookup service. It simplifies to $CT_{EE}+ST_E$

Case 2: the requested service which is found the the original Jini lookup service. It simplifies to $CT_{EE}+ST_E+CT_{EO}+ST_O$

4) The proxy object of the requested service is sent back to the initiating federation and the client. It simplifies to $D\times CT_{EE}+CT_{CE}$

Eventually, we can obtain the response time of a client from the following expressions

Case 1 (The requested service in the extended-Jini lookup service)

$$RT_C = CT_{CE}+(D-1)(CT_{EE}+ST_E+CT_{EO}+ST_O)+CT_{EE}+ST_E + D\times CT_{EE}+CT_{CE}$$

Case 2 (The requested service in the original Jini lookup service)

$$RT_C = CT_{CE}+(D-1)(CT_{EE}+ST_E+CT_{EO}+ST_O)+CT_{EE}+ ST_E+CT_{EO}+ST_O+D\times CT_{EE}+CT_{CE}$$

Now we define an optimal value of the time period of resending request (T_{Period}). If T_{Period} is smaller than RT_C, then it will increase the frequency of resending request. It doesn't only waste the network bandwidth but also increase the workload of the both Jini lookup services moreover the system will be crash. If T_{Period} is bigger than RT_C, then it will increase the response time of clients and reduce the system throughput. So the optimal value of T_{Period} is defined in $T_{Period}\approx RT_C$. However, predicting the value of RT_C is not such simple job, we only give the optimal definition of RT_C in this paper. How to dynamically adjust the value of RT_C will be done in the future work.

System with Mobile Clients

Next, we focus on the mobile client environment that is when a client requests available services in the initiating federation, then moves to a new federation and obtains proxy objects of these services in the current federation.

The response time of a client can be counted in two ways

- *Proxy Object Forwarding*: The proxy-object forwarding scheme defines that when the initiative federation receives proxy objects of demanded services, it will forward these proxy objects to its neighbors, recorded in the neighbor list. The response time of a client is the sum of the RT_C value in the static client environment and the communication time between two extended-Jini lookup services. $RT_C = RT_{Cstatic} + CT_{EE}$
- *Non-Proxy Object Forwarding*: If the client moves to the new federation beyond the scope of the neighbor definition, we use the strategy that a client resends his request. However, when a client resends his request in the current federation, the depth of the federation that the required service located maybe increases one level or reduces one level. In order to simplify our problem, the response time of a client can be given in the following:

$$RT_C = T_{Period} + RT_{Cnew} \approx T_{Period} + RT_{Cstatic}$$

5. Evaluation

In this section, we evaluate our implementation of the extended Jini system and show some experimental results.

Experimental environment description: There are 5 federations. Each federation includes the following: one extended-Jini lookup service, 10 original lookup services. Each lookup service records 10 different types of services. In the first two experiments, the 30 clients send their requests each 30 seconds in one hour by using the non-proxy object forwarding. In the third and fourth experiments, the 30 clients only send their requests once at the same time.

- *Experiment 1*

Purpose: Measure the influence of different values of T_{period} on the system throughput in the mobile environment.

The analysis of result: From the Fig. 5, we observe that when the value of T_{period} is smaller than RT_c, the system throughput will be reduced. It is due to re-sending request packets. Our system has to handle a great deal of requests resulting in the performance of our system going down. When the value of T_{period} is close to RT_c, it brings our system into full play. However, if the value of T_{period} is much bigger than RT_c, the performance of our system goes down again. It is caused by the interval between requests.

- *Experiment 2*

Purpose: Measure the influence of different values of T_{period} on the client's response time in the mobile environment.

The analysis of result: From the Fig. 6, we consider the overhead of lookup services and the network bandwidth of our system caused by re-sending the request packets. We can observe that when the value of T_{period} is smaller than RT_c, the clients will send plenty of request packets in a short time. The performance of our system will go down and the client's response time will be increased. When the value of T_{period} is close to RT_c, it brings our system into full play and the client's response time will be close to the optimal value. However, if the value of T_{period} is much bigger than RT_c, the client's response time will be increased again.

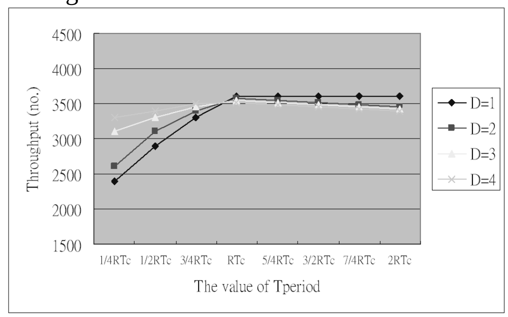

Fig. 5. – Experiment 1: System Throughput. (In Different Values of T_{period})

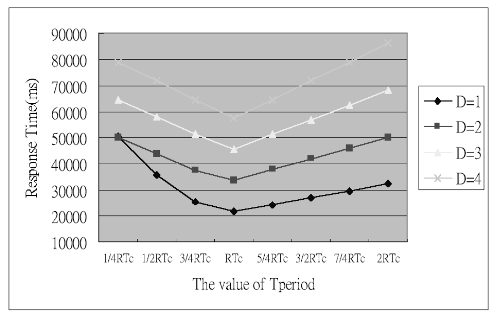

Fig. 6. – Experiment 2: Client's Response Time. (In Different Values of T_{period})

- *Experiment 3*

Purpose: Compare the client's response time of proxy-object forwarding with non- proxy object forwarding in the mobile environment.

The analysis of result: From the Fig. 7, we can observe that if the proxy-object forwarding scheme is adopted; it will reduce the client's response time.

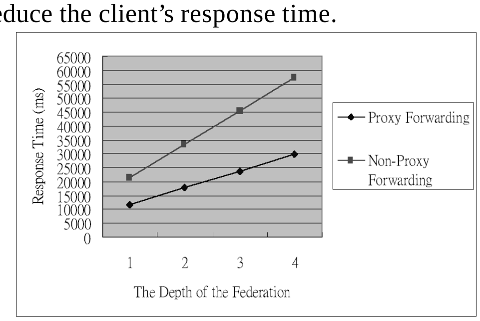

Fig. 7. – Experiment 3: Client's Response Time.

- *Experiment 4*

Purpose: Compare the system throughput of proxy-object forwarding with non- proxy object forwarding in the mobile environment.

The analysis of results: From the Fig. 8, we can observe that if the proxy-object forwarding scheme is adopted; it will improve the performance on the system throughput. However when the value of the depth of federations is close to or larger than four, the response time of clients is larger than the interval of each request. Hence the system performance will be reduced.

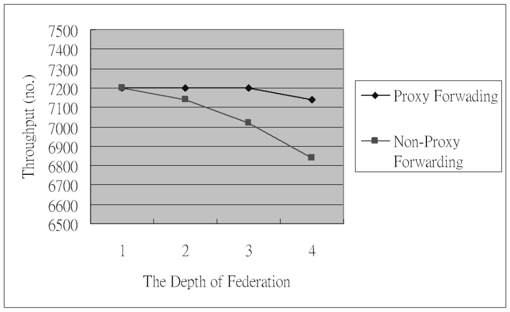

Fig. 8. – Experiment 4: System Throughput.

6. Conclusion and Future Work

In this paper, we propose two new schemes: the "Inter-Federation Communication" and the "Proxy-Object Forwarding" based on Jini. The first scheme solves the problem that the Jini system lacks the inter-federation service lookup. In order to build a flexible distributed system, we use the neighbor list data structure to keep the topological information of our system.

The Proxy-Object Forwarding Scheme solves the problem that the Jini system is not suitable for the mobile environment. Clients could efficiently obtain their requested services with this scheme if they were in the predicted scope, i.e. neighbors of the initiating federation. Additionally, we suggest an optimal value of T_{period} used in non-proxy object forwarding environment. Clients, who move to the new federation beyond the scope of the neighbor definition, could also obtain their requested services as soon as possible.

We also implement an extended-Jini system with these two schemes. From the analysis of the experimental results, we observe that the response time and throughput are significantly improved.

In the future, we can enhance the extended-Jini system by providing a mechanism, through which we can dynamically adjust the value of T_{period} at runtime. In current system we do not consider the "mobile service" issues; hence it is possible that we would add mechanisms to make our extend-Jini system suitable for both of mobile client and mobile service environments. Besides, we could provide SDK to help developers construct more flexible inter-clustered software systems.

References

[1]. The Jini Architecture Specification, available at http://www.sun.com/jini/specs/jini1_1.pdf.

[2]. Stefano Basagni, "Distributed Clustering for Ad Hoc Networks", Proc. of the Fourth International Symposium on Parallel Architectures, Algorithms, and Networks.

[3]. M.A. El-Gendy, H. Baraka and A.H. Fahmy, "Migrating Group Communication Protocols to Networks with Mobile Hosts", Proc. of the 1998 Midwest Symposium on Systems and Circuits.

[4]. Hujic, D.; Zak, G.; Croft, E.; Fenton, R.G.; Mills, J.K.; Benhabib, B., "An active prediction, planning and execution system for interception of moving objects", Proc. of the 1995 IEEE International Symposium on Assembly and Task Planning.

[5]. Gregory V. Chockler et al, "Implementing a caching service a distributed COBRA objects", IFIP/ACM International Conference on Distributed systems platform, 2000.

[6]. Hiroaki Higaki, Naokazu Nemoto, Katsuya Tanaka and Makoto Takizawa, "Protocol for Groups of Pseudo-Active Replicated Objects", Proc. of the Fifth International Workshop on Object-Oriented Real-Time Dependable Systems.

[7]. The Java Remote Method Invocation Specification, available at http://java.sun.com/j2se/1.3/docs/guide/rmi/spec/rmiTOC.html.

[8]. Ramón Cáceres , Venkata N. Padmanabhan, "Fast and scalable handoffs for wireless internetworks", Proc. of the second annual international conference on Mobile computing and networking, p.56-66, November 1996, Rye, New York, United States.

[9]. Jon Chung-Shien Wu , Chieh-Wen Cheng , Gin-Kou Ma , Nen-Fu Huang, "Intelligent handoff for mobile wireless internet", Mobile Networks and Applications Volume 6 Issue 1 Jan./Feb.2001.

[10]. Sing Li et al., Professional Jini, Wrox, Reading, August 2000.

[11]. William Adjie-Winoto , Elliot Schwartz , Hari Balakrishnan , Jeremy Lilley, The design and implementation of an intentional naming system, ACM SIGOPS Operating Systems Review, v.33 n.5, p.186-201, Dec. 1999.

[12]. P. Mockapetris and K. Dunlap, "Development of the Domain, Name System," in Proc. ACM SIGCOMM, Stanford, CA, 1988, pp. 123-133.

[13]. Jaeyeon Jung , Emil Sit , Hari Balakrishnan , Robert Morris, "DNS Performance and the Effectiveness of Caching", Proc. of the First ACM SIGCOMM Workshop on Internet Measurement Workshop November 2001.

[14]. E. Guttman, C. Perkins, J.Veizades, and M. Day, "Service Location Protocol, Version 2", IETF, RFC 2608, June 1999; available at http://www.ietf.org/rfc/rfc2608.txt.

[15]. J. Kempf, and E. Guttman, "An API for Service Location", IETF, RFC 2609, June 1999; available at http://www.ietf.org/rfc/rfc2609.txt.

[16]. G. Liu and G. Maguire Jr., "A Class of Mobile Motion Prediction Algorithms for Wireless Mobile Computing and Communications," ACM/Baltzer MONET, 1(2), 1996, pp.113-121.

[17]. William Su, Sung-Ju Lee, Mario Gerla, "Mobility prediction and routing in ad hoc wireless networks", International Journal of Network Management Volume 11, Issue 1 Jan./Feb. 2001.