A GROUP-BASED METHOD FOR CONTEXT-AWARE SERVICE DISCOVERY IN PERVASIVE COMPUTING ENVIRONMENT

Marzieh Ilka[1], Mahdi Niamanesh[2], and Ahmad Faraahi[3]

[1]Payam Noor University, Tehran, Iran
marzieh.ilka@gmail.com

[2]Imam Reza University, Mashhad, Iran
niamanesh@mehr.imamreza.ac.ir

[3]Payam Noor University, Tehran, Iran
afaraahi@pnu.ac.ir

ABSTRACT

A Pervasive Computing Environment includes a variety of communication networks and service systems and has places suitable for providing variety of services to client. One of the most important problems in such environments is service discovery, i.e. how a client can use his service of interest. In a Pervasive Computing Environment Context, information plays an important role for providing more suitable services to the client. So in this paper, we present a method context aware service discovery with use of context source in the form of an advertisement board for services. A context source with an information register of new entry services and, in addition, forwarding packets, acts like a billboard advertising service. In this context source method, it provides help in registering and advertising services but is not a directory and, if the context source fails, our method can undertake service discovery with other nodes in environment. To evaluate this method a simulation has been carried out. Its statistics and outputs show our suggested method, with its dynamic environment, client movement, directory less, and use of a context source is a suitable method for service discovery which has better results and costs less money and time.

KEYWORDS
Service discovery; Context; Context source; Pervasive Computing Environment; Group-based.

1. INTRODUCTION

Pervasive computing environment includes a variety of mobile nodes (such as cell phone or PDA) and different types of services without any infrastructure, suitable for providing variety of services to clients. The main goal of a pervasive computing environment is to provide an easier service for users. A client is an entity that needs using services and a service is any tangible or intangible facility a device provides that can be useful for any other device [1]. Any client or service in the network is called a node. Properties of any service that show its operation are registered in the directory [2]. The discovery service is simply the recognition of available services in a network and user requests to use them [3].Service discovering in pervasive computing environment is requiring mechanisms for more effective.

In the general service discovery, the protocol classified into two categories: directory-based and directory-less. In directory-based protocols a request packet sends to the directory to discover if

DOI:10.5121/iju.2012.3201

the requested service is there or not. But in directory-less protocols, a request packet is sent to nodes around it, so that by using these nodes and the information that they have from their environment, service discovery is done [4]. Hence, service discovery with a directory-based protocol is easier than with a directory-less one, but it faces the problems of bottleneck and single point of failure. Directory-less architecture is more suited to a pervasive computing environment, because there is no need for any infrastructure [5].

Any information is associated with the interaction between the user and the application of that information in order to describe it and the status if an attribute is called context [6]. Context information contains dynamic properties and this information differs according to entity, activity, and service provided [7]. A context source is a service that provides access to context information, such as the location of a user. A context source has different facilities depending on the service provided, such as memory and ability to record information [8].

In dynamic environment such as pervasive computing environment context awareness has the positive effect of presenting a better service for the client. The context information in service discovery improves the result, and directory-less protocols are better for dynamic nature of pervasive computing environment. But most methods that are context-aware are directory-base and not suitable for pervasive computing environment, while directory-less methods do not pay attention to context information. Combining context-aware service discovery with directory-less protocols has led to methods that have quality and are suitable for pervasive computing environment.

The rest of the paper is organized as follows. We survey related work in Section II, and then present our method for service discovery and its advantages in Section III. We present the evaluation and results of simulation in Section IV. Finally, we give a conclusion in Section V.

2. RELATED WORK

In this section, we review some related work on service discovery methods in pervasive environments. The methods are directory-less and context-aware.

In the Pawar and Tokmakoff service discovery method [9] the main focus is on using of context information and ontology. The model is based on service oriented architecture, in which a group of services using ontology to resolve service discovery. In this model, when a context changes or a new service is added then clients are notified. This model is suitable for fix network but not for pervasive environments.

CASSD (Context Aware Semantic Service Discovery) model [10] uses semantic web technology to describe services. This model is based on a central directory and includes Service Discovery, Semantic Server, Service Requester, Service Provider, and Context Provider components. Each service belongs to one category and used it to filter service discovery. This method has directory and is not suitable for dynamic environment.

In Steller's and his colleagues service discovery [11] the service discovery program may find several services according to a client request. In this case, better service should be offered to the client and, for this, precision and recall metrics are used. Recall measures means how a proper architecture discovery finds all services; and precision, how well the architecture retrieves only the relevant services. This model brings together semantics and context. Context information is represented semantically and attributes are used based on meaning rather than symbols. This model is directory-based and suitable for a fixed network but not for pervasive environments.

GSD (group-based services discovery) [12] is one of the suitable models for service discovery in pervasive environments. It is directory-less but, by using information from nodes which are in the environment register, discovery services are conducted. Service descriptions obtained from advertisement services or other nodes. In this model, services are grouped according to their functionality and it helps by intelligently routing service requests and denying broadcast service requests. Authors claim their model have some advantages such as scalability, self-starting, and adaptability. GSD is a directory less model which is based on grouping techniques for services.GSD does not include context in its discovery method.

Pohare and his colleague's service discovery [13] have used the GSD's model to classify services into groups. This model is context aware. In this model environment has various domains and all the nodes divided in to ultra-peers (UPs) and leaves (LFs). Every LF selects an ultra-peer and sends its information to that. Ups are done service discovery by register information of LFs, its group information, and group information of other Ups. The number of packages in this way in comparison with GSD is reduced and the quality of service is 79%. Service discovery in this method is dependent on Ups.

The summary of analysis the related work is shown in table I.

TABLE I. ANALYSIS RELATED WORK

	Network	**Property**	**Directory**	**Context-Aware**
Pawar	Fix-Mobile	Static-Dynamic	Directory-based	Yes
CASSD	Fix-Mobile	Static-Dynamic	Directory-based	Yes
Steller	Fix	Static-Dynamic	Directory-based	Yes
GSD	Pervasive	Static	Directory-less	No
Pohare	Pervasive	Static-Dynamic	Distributed- Directory	Yes

3. GROUP-BASED METHOD FOR CONTEXT-AWARE SERVICE DISCOVERY IN PERVASIVE COMPUTING ENVIRONMENT

The GSD method is the nearest method relating to a pervasive environment, but GSD does not pay attention to context information. Whereas, considering context information in results an increase in the quality of service discovery.

The Pohare's method is a suitable service discovery method related to pervasive computing environments, but Pohare's method depends on ultra-peers and without them service discovery stops. Whereas, not depending on any special node, is suitable for nature of dynamic environment.

In our method moreover, as mentioned before, we provide the context source by adding new information services in the sending packets without creating overload, which advertises them and acts like an advertising billboard.

In our method, the environment is divided into clusters, and each cluster has a context source. Each cluster recognizes its neighbour. Also each node recognizes our context source cluster. All nodes and context sources have memories for saving information. Our method will be explained.

3.1. Announcing a new service to context source and Advertising it by context source

For announcing a new service, first, the service sends an introduction packet to its context source. This packet includes static properties (with their values) and a list of context properties related to the service. On receiving each introduction packet, the context source reviews its memory. If it has enough memory, it records information received. Otherwise it ignores older information and replaces it with new information. Also, by considering the memory that the new service has the context source sends it new information services which are recorded in it. By doing this, a newcomer service gets information from the environment around it. Likewise, each context sources keep group information of its cluster services and, at the time of changing, sends this information to a neighbour context source; group information from the entry of a service by a new group to the cluster or the exit of a single service belonging to a group from the cluster, will change. After this action, each context source gets a general view of the group information existing in its neighbours.

In a context-aware environment, in order to get its value of context properties, each node should refer to the context source; as a result, the context source is part of nodes which refer to it much more than other nodes. In our suggested method, in the reply packet that context source sends to the requesting node, in addition to the requested information, information of new comer services are also situate. Thus, these services are advertised and the context source is used as an advertising board for services.

3.2. Producing request packet

If a node needs any service, it creates a request packet. Information in this packet includes the service properties, service group, and context properties of requested service. Also, the address of the requester exists in this packet. Using this information is explained in the routing request packet.

Routing request packet has two cases which depend on the status of the context source. If context source is active and available, a request packet is send to the requester, context source, and neighbours based chronologically on a defined radius. If the context source is inactive, the request packet is send to the requester and services what information of it is in the memory of the requester.

- **Routing request packet if context source is available**

At first, the program of service discovery is to check the memory of the requester. If the information of the requested service is in its memory, basic service is discovered. Then, if the requested service has context properties, packets for computed context properties are sent to the context source and, after computing them, discovered service with the value of the context properties is send to the client.

When the information from the requester is not enough, a request packet is sent to the context source of its cluster. The context source reviews the request packet by using information which it derives from advertising its cluster and, if it has the requested service information, it prepares a reply packet which it sends to the client. Of course, if the requested service has context properties these properties are computed too.

If information of the context source is not enough either, group information is used from neighbours and a request packet is send to those neighbours which have the same group of requested service. At first, a request packet is sent to adjacent neighbours. In this step, the action of the neighbours receiving the request packet is similar to review of the request packet by the context source. If this service does not discover, the sending of a request packet to neighbours continues, until service is discovered or the defined radius of neighbours finished.

In all of these steps if context source receives a request packet, it adds information of new services in the reply packet (discover reply or context value reply) too. The receiver node adds this information to its memory. Thus, the context source without creating any overheads advertises services which are stored in it.

- **Routing request packet if the context source is not available**

The first step of service discovery without attention to the status of the context source is to check the memory of the requester. In this step, if a suitable service is discovered the action will be finished. Otherwise, if that context source is not available, a request packet is sent to services of which information is in the requester's memory. Services that receive a request packet check their memories and, if they have information of a requested service, basic services are discovered and the reply packet is prepared and send to the requester.

Thus changing the context source to directory is prevented and nodes which are in any cluster by keeping and recording information of the environment, are helping other nodes during inactive periods of the context source.

4. EXPERIMENTAL EVALUATION

To evaluate our method, metrics such as "number of produced packets" and "performance" are calculated and examined. In this section, we explain how to calculate the number of packets produced.

4.1. Describe Attribute of Simulation

To evaluate the proposed method a simulation tools is developed. Facilities of this program are "define environment information" and "service discovery", as shown in Figure 1. When an entity is defined, its memory is allocated. The memory is used for registering information which is obtained from the environment. Basic information such as number of requests for service discovery, number of running programs, probability of context source availability, probability of user movement, and number of neighbours that reviewed in different states are initialling in service discovery part. Service discovery action base on this information and suggested method starts to work.

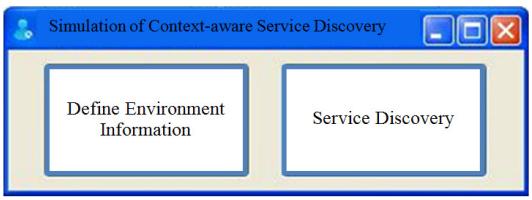

Figure 1 : Running Simulation Environment

Results of service discovery in different states of "context source", "user movement", and "number of neighbour" are reviewed and effect of each on the evaluation metrics are expressed. An active context source means this node is available and also user movement means users move in defined locations. In running simulation basic information of users and services are similar. Table II represents the primitive simulator parameters used for the experiment.

TABLE II. SIMULATIONS PARAMETERS

Parameters	Value
Number of users	100
Number of services	30
Probability active context source	70% to 100%
Probability user movement	0% to 40%
Number of request for service discovery	100
Number of running program	10
Number of neighbours that reviewed	3

4.2. Calculate Number of Packets Produced

The number of produced packets in the proposed method is depend on the status of the context source, place of discovery (requester, context source, neighbours, or other services), and type of context properties. The formula for calculating the number of packets is shown in Table III. We show number of services that reviewed with "S".

TABLE III. FORMULA TO CALCULATE NUMBER OF PRODUCED PACKET

Place of Discovery	Do not have context property	Have context property
Requester	2	4
Context Source	4	6
Other Services	S* 2+2	

Paying attention to "status of context source", "number of neighbours", and "user movement" number of packets produced were calculated and examined.

4.2.1. Number of Packets Produced and Status of Context Source

When service discovery processing, context source status may available or not available. By attention to this point, number of packets produced is different. We illustrate the result in Figure 2. As the Figure 2 shows if activity of the context source decreasing, producing number of request packets are increases. Request packets should be sent to other nodes instead of context source, this point is correct.

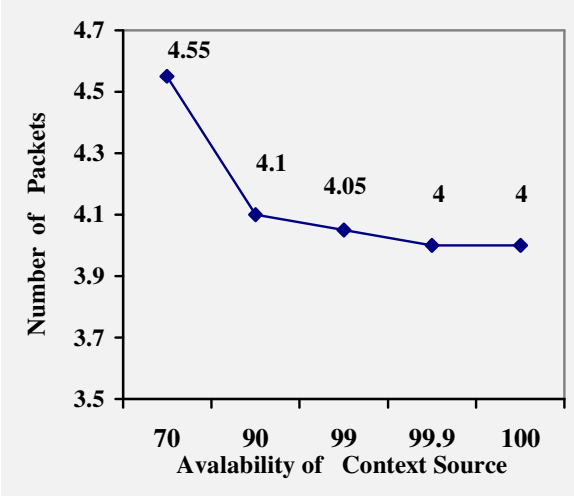

Figure 2 : Number of Packets and Avalability of Context Source

As the result of Figure 2 demonstrates, if the availability of the context source is 99% then the average of produced packets increases to 0.05.

4.2.2. Number of Packets Produced and Number of Neighbours

When a request packet does not discover in client or its cluster, it has to send to neighbours. From the result of Figure 3, it can be concluded that if a request packet send to neighbours, number of packets produced will be more.

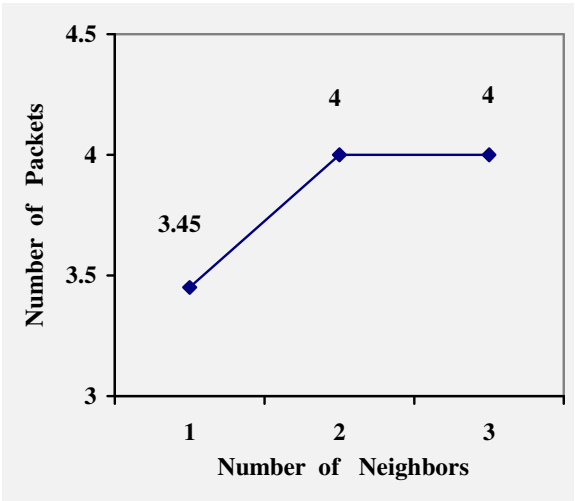

Figure 3 : Number of Packets and Number of Neighbour

4.2.3. Number of Packets Produced and User Movement

One of the properties of pervasive computing environment is changing and moving. By attention to this point, clients' movements possibility is considered differently. When a user moves from one cluster to another cluster, his information from the old cluster is not suitable for service discovery in the new cluster and using old information may even cause incorrect service discovery. So the use of older information is not useful in user movement from one cluster to another cluster.

By checking Figure 4 we could conclude that increasing user movement possibility leads to an increase number of packets produced. For instance, a 5% increase in user mobility causes a 1.3% increase in number of packets produced.

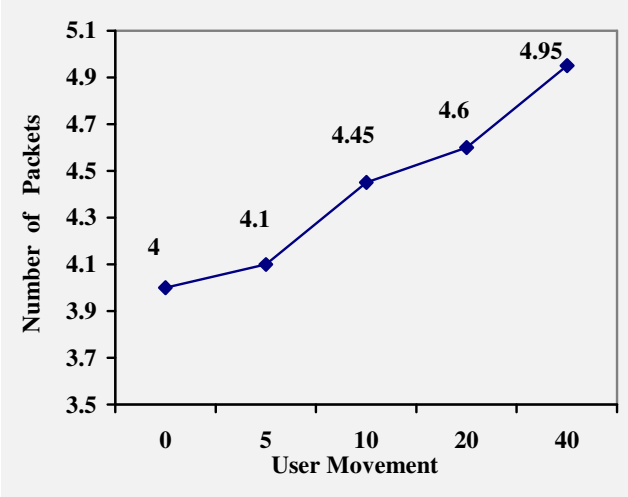

Figure 4 : Number of Packets and User Movement

4.3. Performance

In our method performance is calculated by the number of successful discoveries. By attention to "Status of Context Source", "Number of Neighbours", and "User Movement" performance were calculated and examined.

4.3.1. Performance and Status of Context Source

From the result of Figure 5, it can be concluded that if a context source is more available, the percentage success of service discovery will be more. The cause of this result is the information that each context source has gained in its cluster.

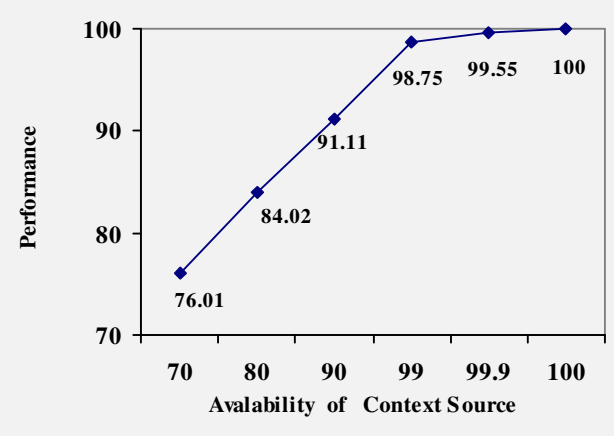

Figure 5 : Performance and Avalability of Context Source

By reviewing the Figure 2 and Figure 5, it can be concluded that if the context source is more often enabled, the results of service discovery will be more favourable. Of course, service discovery does not stop with decreasing availability of the context source, and by producing more packets service discovery will succeed; however, the probability of being successful decreases. Another point is, if the context source has 99% availability then the average of packets produced is increased by 0.05 and discovery would decrease by about 1.25%.

4.3.2. Performance and Number of Neighbours

As stated when a request packet does not discover in its cluster and have permission to send request to neighbour, service discovery is done more successful. By checking Figure 6 we could conclude that radius of neighbour that reviewed increasing leads to an increase in performance.

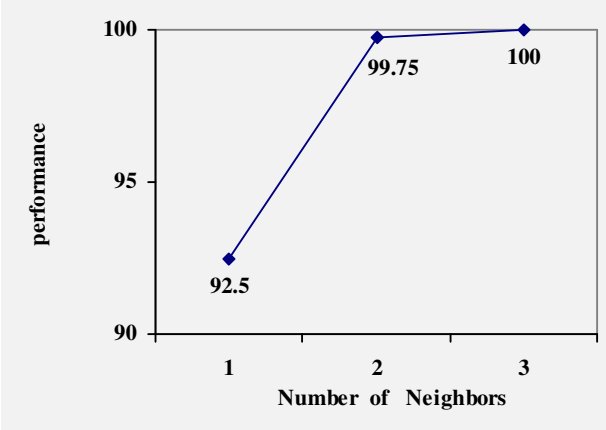

Figure 6 : Performance and Neighbour

4.3.3. Performance and User Movement

Given the information from the simulation environment definition, namely that each cluster has neighbours within a radius of 3 and in a simulation exercise (Table II) it has permission to

examine neighbours within a radius of 3, so it is concluded that, if a user moves, a request packet can be sent to other neighbours and so percentage of 100% successful service discovery remains.

5. CONCLUSIONS

In this paper, we have proposed a context aware group based service discovery method in pervasive computing environments that uses context sources as advertisement boards for services. By keeping information of the latest services in each cluster, each context source helps service discovery method. Also, by putting information of new services in transmission packets, context sources proceed to advertise them. Besides, our proposed method is able to discover services without the existence of any context source through their nodes and information. To evaluate the proposed method, we have developed a simulation tools to measure some parameters including average number of exchanged packets and performance. In the simulator, users have the ability to move among different clusters and request any services and if a request packet does not discover in its cluster, it could send to neighbours. The results of the simulation indicate that using context sources, our proposed approach has acceptable performance and number of exchanged packets. In comparison with directory-based and directory-less methods, we do hybrid and work even without any context source.

In the future work we would like to, by attention to user's interests is forwarded additional information. Also considering the importance of each property in request would increase quality of service discovery that will be considered in future work.

REFERENCES

[1] Mian. A. N., Baldoni. R., Beraldi. R., (2009) **"A Survey of Service Discovery Protocols in Multihop Mobile Ad Hoc Networks"**, Pervasive Computing IEEE, Vol. 8, pp. 66–74.

[2] Mian. A. N, Baldoni. R, Beraldi. R, (2009) **"A Survey of Service Discovery Protocols in Multihop Mobile Ad Hoc Networks"**, Pervasive Computing IEEE, Vol. 8, pp. 66–74.

[3] R. Marin-Perianu, P. Hartel, and H. Scholten, (2005) **"A Classification of Service Discovery Protocols"**, Electrical Eng. Mathematics and Computer Science (EEMCS), pp. 1–22.

[4] Cho. C., Lee. D., (2005) **"Survey of Service Discovery Architectures for Mobile Ad hoc Networks"**, Department of Computer and Information Science and Engineering (CICE), Vol. 5531, pp. 1-10.

[5] Thangam. S, Kirubhakaran, E, (2010) **"Analysis of Various Service Discovery Protocols for Infrastructure-less Networks"**, International Journal of Computer Applications (0975 – 8887) Volume 12– No.8.

[6] Hong-Linh Truong, Schahram Dustdar, (2009) **"A survey on context-aware web service systems"**, International Journal of Web Information Systems, Vol. 5, pp.5 – 31.

[7] Baldauf. M., Dustdar. S, Rosenberg. F., (2007) **"A survey on context-aware systems"**, International Journal of Ad Hoc and Ubiquitous Computing, Vol. 2, pp. 263-277.

[8] Hesselman. C., Tokmakoff. A., Pawar. P., Iacob. S., (2006) **"Discovery and Composition of Services for Context-Aware Systems"**, 1st IEEE European Conference on Smart Sensing and Context, Vol. 4272, pp. 67-81.

[9] Pawar. P, Tokmakoff. A, (2006) **"Ontology-Based Context-Aware Service Discovery for Pervasive Environments"**, 1st IEEE International Workshop on Services Integration in Pervasive Environment, PP: 1-6.

[10] Patel. P., Chaudhary. S., (2009) **"Context Aware Semantic Service Discovery"**, World Conference on Services – II, pp.1-8.

[11] Steller. L, Krishnaswamy. S, Newmarch. J, (2006) **"Discovering Relevant Services in Pervasive Environments Using Semantics and Context"**, International Workshop on Ubiquitous Computing - IWUC, pp. 3-12.

[12] Chakraborty. D, Joshi. A, Yesha. Y, Finin. F, (2006) **"Toward Distributed Service Discovery in Pervasive Computing Environments"**, IEEE Transactions on Mobile Computing, Vol. 5, No. 2, pp. 97-112.

[13] Yau S, Pohare. G, (2009) **"An Efficient Approach to Situation-Aware Service Discovery in Pervasive Service Computing Environments"**, 6th International Conference on Ubiquitous Intelligence and Computing, Vol. 5585, pp. 68-82.