# Service Discovery Technology in Autonomous Decentralized Community System

Yuji Horikoshi Khaled Ragab Naohiro Kaji Kinji Mori
Department of Computer Science, Tokyo Institute of Technology
2-12-1 Ookayama Meguro, Tokyo, 152-8552, Japan.
Tel: +81-3-5734-2664, Fax: +81-3-5734-2510
E-mail: {horikoshi@mori., ragab@mori., nkaji@mori., mori@}cs.titech.ac.jp

## Abstract

In recent years, information services in accordance with users' preference, utilization place and time are required more and more. Autonomous Decentralized Community System (ADCS) has been proposed to provide services for a Local Majority, which consists of users who have similar preferences or are in a similar situation. ADCS network consists of many autonomous stationary nodes that support wireless communication to users' mobile terminals and the network covers lands. Cooperation among nodes achieves flexibility of the system and allows users in a Local Majority to cooperate mutually and share information.

In ADCS various services, which are created by users or service providers (SP), are allocated on the network and their positions are unpredictable. To utilize services, the distributive service discovery technology, which limits request forwarding with TTL, is available. However, it has problem on response time. This paper proposes the service discovery technology that does not limit query forwards but make reply messages as cancel massages that stop request forwarding follow and catch up with request messages by cooperation among nodes based on Autonomous Decentralized Community System concept. The effectiveness of proposal on response time is shown by simulation.

## 1. Introduction

The advancement of information and communication technology has made global and various information services available in anytime and anywhere for anyone. Mobile information service terminals have been pervasive. Users can access many services or the Internet with mobile phone services like *i-mode* in Japan.

Users, however, have been provided with too many services to know which service is appropriate for them. Service providers (SP) also have been troubled in service for users by various needs of increasing users. Under this background, the requirement for mobile information services with situation-awareness has increased. The information services, which consider users' situations, time and place, and preferences, are required.

We have proposed Community Service, where services are provided through mutual cooperation between SP and a Local Majority, which consists of users who have a similar preference or are in a similar situation, and mutual cooperation among a Local Majority.

A system with flexibility and timeliness is required to provide Community Service, because a service area changes according to a state of existence of a Local Majority and properties of services, and users would like to utilize at the moment.

Some location-aware systems using mobile terminals for providing services based on locations of users have been reported.[1][2] These technologies have described a basic concept of service mediation platform. They has assumed that each service is provided to static area and considered no dynamic area. Systems using a service accelerator system and an autonomous decentralized service system to provide personal service have been reported.[3][4] These systems has mediated between SP and a user and provided an individual user with services based on their profiles. They have not considered a group of users.

Autonomous Decentralized Community System (ADCS) has been proposed to provide services for a Local Majority.[5][6] ADCS consists of many autonomous nodes forming a network and cooperation among nodes achieves flexibility of the system and allows users in a Local Majority to cooperate mutually and share information.

In order to provide Community Service such as the information sharing in ADCS, a node corresponded to a user requesting information needs to discover information created by other users, which is defined as *service*, i.e. service discovery is required. A naïve method to discover services is available. However, it has problem on the response time to discover.

This paper reports the proposition of Autonomous Service Discovery Technology for timeliness in ADCS.

This paper is structured as follows. Next section presents the application and system requirements. Section 3 discusses ADCS. Section 4 exposes Autonomous Service Discovery Technology. Section 5 shows the effectiveness of this technology. Last section concludes this paper.

## 2. Application and Requirements

### 2.1. Community Service

User would like to utilize appropriate services in accordance with users' situations and preferences. Community Service to satisfy this need has been proposed. Community Service is the service for a Local Majority and it is achieved through mutual cooperation among a Local Majority and cooperation between SP and a Local Majority.

The image of Community Service is as follows. In lunchtime, hungry users in a city area want to know restaurant information. They are in a Local Majority for a restaurant as a SP. The SP disseminates information of lunchtime service for accessible users to the SP. Users in the Local Majority want to exchange information on restaurants by word of mouth with their mobile terminals.

### 2.2. System Requirements

In order to provide Community Service, a system with flexibility and timeliness is required.

Users would like to utilize the appropriate service in accordance with their preferences and situation like their location and time to SP in each user' s accessible area. The service areas differ according to each service property. Therefore, flexibility is required.

Users would like to utilize appropriate services at the moment. Therefore, timeliness is required.

## 3. Autonomous Decentralized Community System

Autonomous Decentralized Community System (ADCS) has been proposed, which achieves flexibility described previously. The System is structured based on Autonomous Decentralized System.[7] ADCS consists of autonomous nodes connecting each other. In order to provide services, nodes forms autonomously a group, which is called *community*, in accordance with service property. In a community, all messages are broadcasted within the community. Figure 1 shows the architecture of this system.

The architecture is Data Field architecture and each node communicates and processes autonomously with content codes. Each node consists Autonomous Control Processor (ACP) that judges and processes based on local information, storage that stores users' requests and

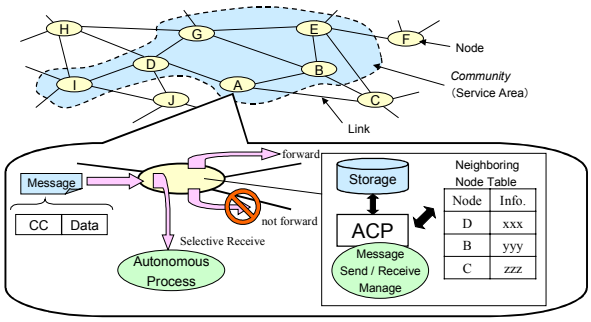


**Figure1: System Architecture**

services and Neighboring Nodes Table (NNT).

Each node autonomously judges whether a received message is forwarded to its neighboring node by service property in messages and NNT. As a result of this judgment of each node, the group that consists nodes receiving the message is a community.

Nodes are base stations for wireless communications and connect SP and users with mobile terminals. The network covers lands. They transmit data between physically neighboring nodes and broadcast messages to the users within the sphere of each cell. SPs and users communicate with the physically nearest node and have information such as users' preferences and service information.

The area covered by community of nodes is service area. Users in the area can exchange information.

## 4. Autonomous Service Discovery Technology

### 4.1. Service Discovery

In ADCS, when services a user demands are not provided the node the user stays in, discovering services to create a community is needed. In order to provide Community Service for a user, discovering of information that other users or SPs provide is required. Such information is defined as *service*.

The architecture of ADCS is based on the network that consists of many nodes. Services provided in ADCS are massive numbers and positions of services change by user's moving. Thus, their positions are unpredictable.

Because many services are provided and change every moment, a centralized mediation of services is difficult. Therefore, distributive discovering is required. It is realized that a node which request services send request messages to the network and nodes on the network relay these messages one by one.

### 4.2. Problem in Service Discovery

In Autonomous Decentralized Community System, to provide services in accordance with users' situations too wide a search area for service discovery is unsuitable for

user's situations as well as produces wasteful traffics. Therefore, to stop a propagation of requests is required.

A naive method using messages with Time To Live (TTL), which decreases every a forward and makes a message vanish when it becomes zero is available. Such a method is used in some P2P systems.[8]

However, to design an appropriate TTL in advance is difficult because services' positions are unpredictable and physical density of nodes is not uniform. Under this situation, requests with a certain TTL may fail to discover services. In this case, a node with the naive method waits for receiving a reply during an adequate term using a timer: the service discovery timer, and the node detects failure of it the timer expires. A node sends a request with bigger TTL if fails, and repeats this process until a service is discovered.

Thus, under unpredictable situation, the naive method cannot achieve timeliness on service discovery because of waiting time in the case of failure. A technology which assures timeliness on service discovery even if the unpredictable situation is required.

## 4.3. Overview of Autonomous Service Discovery Technology

In order to prevent the response time of service discovery from worsening in unpredictable situations, Autonomous Service Discovery Technology is proposed. This technology does not limit an area request messages reach in advance, but makes reply messages, which stop forwards of request messages, pursue and catch up with request messages. Thus service discoveries can be done with only one sending of requests to a network and the response time is prevented from worsening caused by waiting time for requesting again when a request fails.

The concept of the proposed technology regards propagations of messages as ripples on a network. This technology makes a propagation of requests be a ripple with lower speed and that of replies be a ripple with higher speed. Moreover, it makes the propagation of reply massages cancel out that of request messages when both meet.

When a requester emits the ripple of request and it reaches a replier, the ripple of reply is also emitted. Although not only the ripple of requests but also that of replies propagate on a network, the ripple of replies is faster than the other and catch up with it sooner or later. Then propagations of both of messages stop and an area messages reach is a community. Services are provided to users in this community. Figure 2 shows this process.

## 4.4. Message Formats

The formats of two message types used in proposed technology are described as follows.

### 4.4.1. Service Discovery Request Message

Figure 3 shows message format of request message for service discovery. CC indicates the message is request message for service discovery. Requester Node ID means an identifier of the node that sends the message. Message ID is a unique identifier for each message given by the requester node. Requester Node ID and Message ID identify each message. Request Content contains how services a user wants to discovery and utilize. Each node judges whether it can reply for the request message with this field.

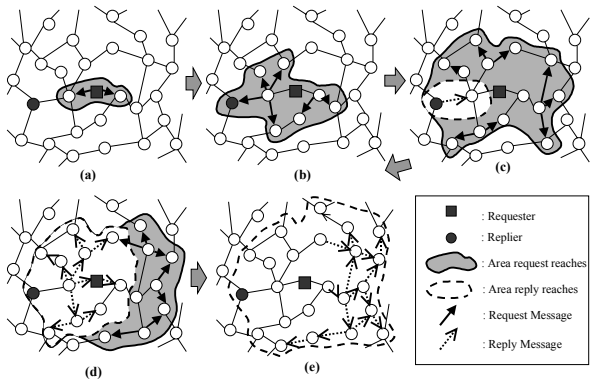


**Figure 2: A Service Discovery Process**

| Content Code (CC) | Requester Node ID | Message ID | Request Content |
|---|---|---|---|

**Figure 3: Format of Request Message**

### 4.4.2. Service Discovery Reply Message

Figure 4 shows message format of reply message for service discovery. CC indicates the message is reply message for service discovery. This field indicates which node has sent the request message corresponded to this reply message. This represents which request message this reply message replies for. This means an identifier of the node that replies for the request. Nodes that receive reply messages can find which node provides the service. This contains contents of reply. Contents vary in accordance with services.

| Content Code (CC) | Requester Node ID | Request Message Message ID | Replier Node ID | Reply Content |
|---|---|---|---|---|

**Figure 4: Format of Reply Message**

## 4.5. Basic Operations of Nodes

Each node takes five states for each request message in accordance with a process of the service discovery. When nodes receive a message, nodes check CC of the received message. If CC indicates Service Request Message or Service Reply Message, nodes act and change states according to the current state and content of received message. Some states have timer and nodes do similarly when the time is up.

In order to identify each process of service discovery when the other process rises at the same time, nodes have the state table as NNT that makes the states correspond to request messages. NNT memorizes the destinations of forwards of the messages. Figure 5 shows this table.

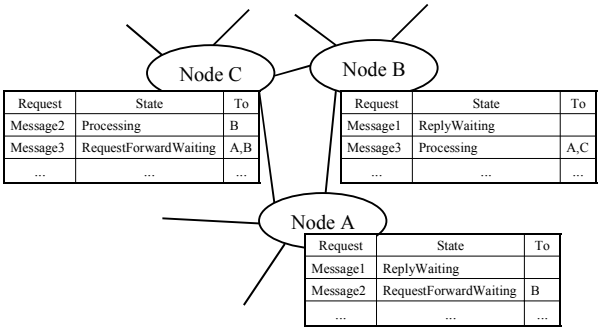


**Figure 5: Neighboring Node Table (NNT)**

## 4.6. State Transitions of Nodes

Nodes take five states that are Normal, Processing, Request Forward Waiting, Reply Waiting and Message Absorbing based on situation in the process of service discovery. Figure 6 shows the state transition diagram for this technology. The initial state is Normal.

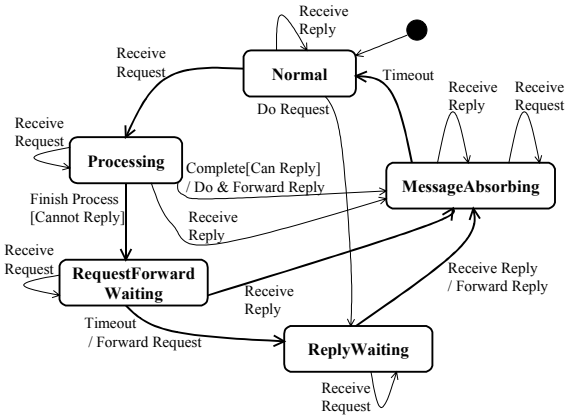


**Figure 6: State Transition Diagram of Nodes**

Figure 7 shows the correlation of the situation of the process of this technology shown in Figure 2 with the states of nodes shown in Figure 6. The state transitions are like a cycle, i.e. the cycle starts from Normal state that nodes have not received a request message yet and ends as states of all nodes involved in the process of service discovery becomes Normal.

Detail descriptions of each state including actions of nodes when an event, a receiving of a message is or expiring of a timer etc., occurs are as follows.

### 4.6.1. Normal State

This is the state that a node has not received a service-discovery request message. To avoid infinite

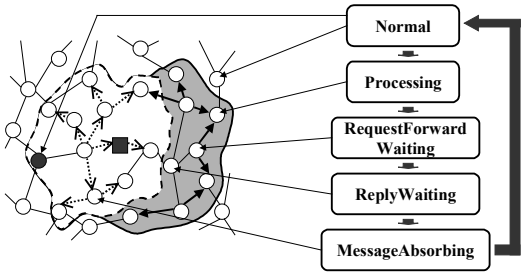


**Figure 7: Correlation Between State Transition and Service Discovery Process**

transmissions of service-discovery reply messages, nodes discard any reply messages except the reply message corresponded to the request message that they have received. Actions of nodes when an event occurs are described as follows.

**Receive service-discovery request message**

The node registers the request message in NNT and sets all neighboring nodes except the source node of forward as destinations. Then, the node goes to Processing State.

**Receive service-discovery reply message**

The node discards this reply message.

**Send new service-discovery request message**

The node creates service discovery request message and sends it to all neighboring nodes. Then, the node goes to Reply Waiting State.

### 4.6.2. Processing State

This is the state that the node processes the received request message and judges whether the node can reply to this message.

**Judge that the node can reply to the request**

The node creates the service-discovery reply message and sends it to all neighbors. Then, the node goes to Message Absorbing State.

**Judge that the node cannot reply to the request**

The node goes to Request Forward Waiting State.

**Receive service-discovery request message**

The node deletes the node that has forwarded this request message from the destination list in NNT. Then, continues processing.

**Receive service-discovery reply message**

The node discontinues processing and sends the received reply message to all neighbors except the forwarder of it. Then, the node goes to Message Absorbing State.

### 4.6.3. Request Forward Waiting State

This state uses a timer and realizes slowing of propagation of requests. The request forward waiting timer determines how long a node may wait forwards of the request message. If this timer expires, the node

forwards the request message to neighboring nodes listed in NNT. If the node receives a reply message until this timer expires, it means that the reply message catches up with the request message. The node cancels forwards of the both messages or messages propagate no more. Each node autonomously determines $T_{wait}$, the length of the request forward waiting timer.

**Receive service-discovery request message**
The node deletes the node that has forwarded this request message from destination list in NNT and continues waiting.

**Receive service-discovery reply message**
The node cancels forwards of the request message and the reply message. Then the node goes to Message Absorbing State.

**The request forward waiting timer expires**
The node forwards the request message that had been received to nodes listed in NNT. Then, the node goes to Reply Waiting State.

#### 4.6.4. Reply Waiting State

This is the state that the node waits for receiving the reply message correlated to the request message the node has received. When it is received, the node forwards it to neighboring nodes. This process makes reply messages pursue preceding request messages.

**Receive service-discovery request message**
The node discards this request message.

**Receive service-discovery reply message**
The node forwards this message to all neighboring nodes except the node that has sent it. Then, the node goes to Message Absorbing State.

#### 4.6.5. Message Absorbing State

This state is a buffer to avoid direct returning to Normal State. This prevents the node from reentering the cycle of the process of service discovery when the node, in spite of completion of the cycle, receives a request message again via another path. This state uses a timer. The message-absorbing timer determines how long the node continues to discard received messages. This state has $T_{Absorb}$, the length of the message-absorbing timer. This is given adequately long time.

**Receive service-discovery request message**
The node discards the received message.

**Receive service-discovery reply message**
The node discards the received message.

**The message-absorbing timer expires**
The node deletes the request message corresponded to this process from NNT. Then, the node goes to Normal State.

## 5. Simulation

Simulations verify the behavior and the response time of proposed technology. Response time is defined as the time from when a requester sends the request message until when the requester receives a reply message.

The simulation model is shown as follows. The simplified network topology shown in Figure 8 is similar to the topology used in actual simulations.

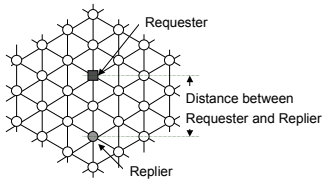


**Figure 8: Simulation Model**

There are one requester and one replier in the network. The time of transmission delay of links is 1 [ms]. The time required for process of each node is 5 [ms]. The time of the message-absorbing timer ($T_{Absorb}$) is 50 [ms].

#### 5.1.1. Verification of Proposed Technology

Many simulations have verified the proposed technology. Figure 9 shows states of a simplified simulation process. The black node at the center of the network is requester node and the black left-lower node is requester. In this simulation, the time of the request forward waiting timer ($T_{wait}$) is 8 [ms].

#### 5.1.2. Evaluation on Response Time

The response time of the proposed technology is compared with that of the technology using TTL under the condition that the distance from a request node to a replier node changes from 1 to 15.

The technology using TTL sends a request message with TTL 5 at first. If the first trial fails to discover services, the node sends a request with TTL 10 again. If the second trial fails, TTL 15 is adopted.

The result is shown in Figure 10. Figure 10 describes that the response time of the proposed technology is proportional to the distance between a requester and a replier. However, the response time of the naïve method worsens drastically as the distance becomes bigger. Because the response time with the proposed technology does not worsen drastically even though a distance to services increases, the proposed technology has the effectiveness on timeliness in unpredictable situations; even if a distance to services is unpredictable and the distance may be far, the technology always achieves the moderate response time.

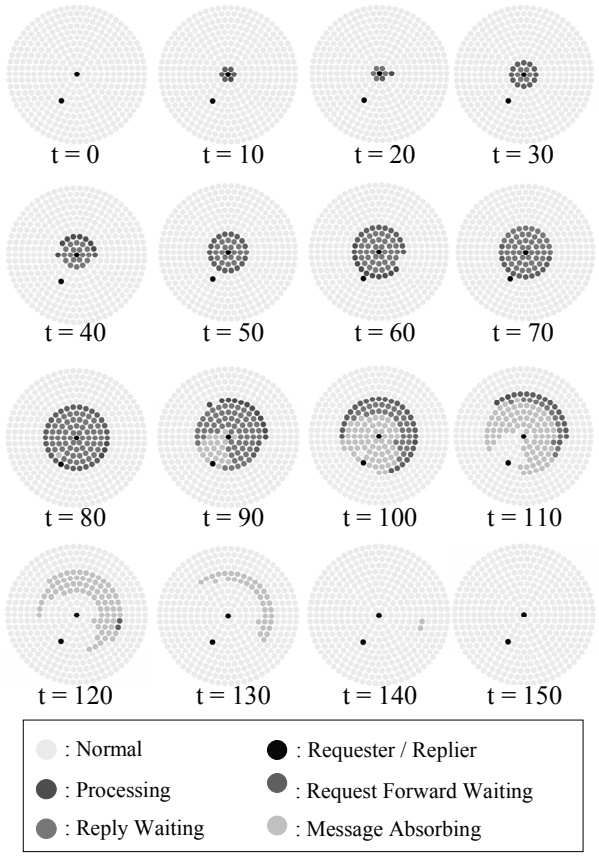


**Figure 9: Simulation Result**

## 6. Conclusion

Community Service, which is provided through cooperation among a Local Majority, is proposed to provide services in accordance with users' situations and preferences. To provide Community Service, a system with flexibility and timeliness is required. Autonomous Decentralized Community System is for providing Community Service. In ADCS, flexibility is achieved by the architecture that forms a community, which is a group of nodes. Broadcasting of information within a community allows Community Service such as information sharing among users near each other at the time.

When services which a user want is not provided, service discovery to form a community is required. Autonomous Service Discovery Technology that is able to discover services without limiting propagations of request messages in advance is proposed. This technology avoids worsening of the response time on service discover in unpredictable situations, which the naive method has. Simulations verify behavior of this technology and show effectiveness on timeliness in unpredictable situations. There is a difficulty to determine the waiting time of each node, nevertheless the response time does not worsen even if replier node is far from requester. Nodes' autonomous function to determine waiting times will be studied.

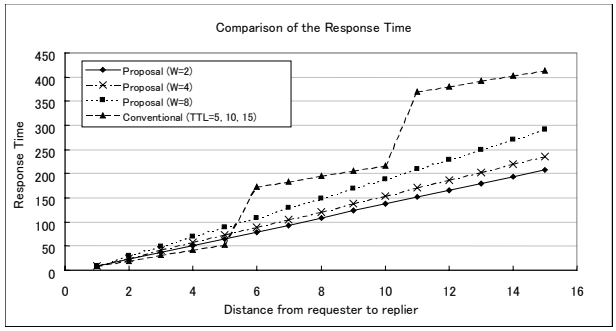


**Figure 10: Simulation Result on Response Time**

## References


[1] J. Hightower and G. Borriello, "Location Systems for Ubiquitous Computing," IEEE Computer, vol.34, no.8, pp.57-66, 2001.

[2] N. Marmasse and C. Schmandt, Location-Aware Information Delivery with ComMotion,"HUC2000, LNCS1927, pp.157-171, 2000.

[3] Kinji Mori et al., "Service Accelerator (SEA) System for Supplying Demand Oriented Information Services", Proc. of IEEE CS Workshop Future Trend of Distributed Computer Systems, pp149-161, 1994.

[4] ADSS DSIG (Autonomous Decentralized Service Systems, Domain Special Interest Group), White Paper for ADDS, ads/98-12-01, OMG, 1997, http://www.omg.org/

[5] T. Ono, K. Ragab, N. Kaji, and K. Mori, Service-oriented Communication Technology for Achieving Assurance, The 22nd International Conference on Distributed Computing Systems Workshops, pp69-74, 2002.

[6] K. Ragab, T. Ono, N. Kaji, and K. Mori, Community Communication Technology for achieving Timeliness in Autonomous Decentralized Community Systems, Proceedings of the 2nd International Workshop on Autonomous Decentralized System (IWADS 2002), pp56-60, 2002.

[7] Kinji Mori, "Autonomous Decentralized Systems: Concept, Data Field Architecture and Future Trends", Proc of ISADS93, pp150-157, March 1999.

[8] Matei Ripeanu. Peer-to-Peer Architecture Case Study: Gnutella Network. In Int'l. Conf. on Peer-to-Peer Computing (P2P2001), Linkoping, Sweden, August 2001.