PAPER *Special Section on New Technologies and their Applications of the Internet*

Autonomous Pull-Push Community Construction Technology for High-Assurance

Khalid MAHMOOD[†a)], **Xiaodong LU**[†], **Yuji HORIKOSHI**[†], ***Nonmembers***, ***and*** **Kinji MORI**[†], ***Fellow***

SUMMARY Location Based Services (LBS) are expected to become one of the major drivers of ubiquitous services due to recent inception of GPS-enabled mobile devices, the development of Web2.0 paradigm, and emergence of 3G broadband networks. Having this vision in mind, Community Context-attribute-oriented Collaborative Information Environment (CCCIE) based Autonomous Decentralized Community System (ADCS) is proposed to enable provision of services to specific users in specific place at specific time considering various context-attributes. This paper presents autonomous community construction technology that share service discovered by one member among others in flexible way to improve timeliness and reduce network cost. In order to meet crucial goal of real-time and context-aware community construction (provision of service/ service information to users with common interests), and defining flexible service area in highly dynamic operating environment of ADCS, proposed progressive ripple based service discovery technique introduces novel idea of snail's pace and steady advancing search followed by swift boundary confining mechanism; while service area construction shares the discovered service among members in defined area to further improve timeliness and reduce network cost. Analysis and empirical results verify the effectiveness of the proposed technique.

key words: *autonomous decentralized community system, community context-attribute-oriented collaborative information environment, community pervasive services, service discovery protocols, autonomous community construction, progressive ripple zone (PRZ)*

1. Introduction

Immense growth in mobile telephony, development of real-time transfer of data over 2.5G and 3G networks and introduction of WAP and GPRS technologies, parallel to tremendous growth of consumer interest and embracement of e-commerce, has resulted in proliferation of services for mobile users. Constructed from the service providers (SP)' point of view and by virtue of using static service declarative models, current information systems bring forth services to anyone, anywhere, anytime– SPs impart information regardless of the dynamic context attributes like end-users' demands and varying surrounding situations. However, recently *Location Based Services* (LBS) has enabled provision of services based on the geographical location of the ubiquitous mobile devices [6], [7], [25], but still they don't consider vital context attributes like time aspect and user's interests. Therefore, to offer services on the basis of correlated spatio-temporal (location specific, time oriented) situations and users with common interests can't be satisfied through global information services of the Internet or location-based services. Hence, *Community Context-attribute-oriented Collaborative Information Environment* (CCCIE) is proposed to enable provision of services to specific users in specific place at specific time, through collaborative processing of various surrounding dynamic contexts [4], [14], [21]. CCCIE incorporates various context-attributes, like time-awareness and user's demands, to revamp one-dimensional LBS into *n*-dimensional situation-aware *Community Pervasive Services* (CPS).

Given the architecture, Service Discovery Protocols (SDPs) in areas of both wireless and wired networks can be classified into centralized directories based, purely decentralized P2P based, and hybrid of above two-involving concept of service coordinators [5], [8], [16], [17]. Intuitively, the approach of centralized registries or service coordinators for *CPS* can't satisfy *high assurance* [2] due to performance bottlenecks, potential risk of single point of failure or inconsistency problems (if backup exists), and potential vulnerability of DoS attacks. Moreover, centralized approaches can't scale well, as it is highly unfeasible to update all services and associated distributed contexts frequently in dynamic operational environment. Therefore, to satisfy *high assurance Autonomous Decentralized Community System* (ADCS) is defined as a place of a coherent group of autonomous members (peers) having individual objectives, common but varying interests and demands at specified time and somewhere/anywhere; and it involve coordination and cooperation among members to acquire the peculiarities of CCCIE.

Community construction technology primarily requires (a) timely and context-aware service discovery technique to identify appropriate service provider subjected to real-time constraint in highly dynamic operational environment of ADCS, (b) defining flexible service area in which discovered service is shared among other members to reduce network cost and to further improve timeliness. Existing distributed service discovery strategies in area of peer to peer and wireless networks employ TTL based *n-ring* model [10]–[12] which consists of *n* successive iterations. Since all iterations re-visit the area that has already been covered by previous endeavors, it results in deterioration of timely service search/provision. Also, before initiating next iteration certain delay, proportional to current ring size, is introduced to ensure that response of service (if discovered) reach at source node. Moreover, various heuristics in *n-ring* search [11], [12] have focused on finding optimum

Manuscript received February 2, 2009.
Manuscript revised May 25, 2009.

[†]The authors are with the Department of Computer Science, Tokyo Institute of Technology, Tokyo, 152–8522 Japan.
a) E-mail: malik@mori.cs.titech.ac.jp

DOI: 10.1587/transinf.E92.D.1836

search radius; but they assume uniform distribution of nodes in the network, and require knowledge like total number of nodes and maximum hop number. Therefore, it becomes difficult to estimate suitable size of searching-ring (or appropriate TTL) in presence of non-uniform distribution of nodes, and highly dynamic operational environment (like ADCS). Likewise, all existing SDPs [27] don't discuss how to specify search radius to facilitate context-aware service discovery in presence of multiple SPs.

Moreover, existing distributed SDPs in area of mobile computing [5], [8], [16], [17] follow inflexible push-pull model where service advertisements pushed by SP are disseminated in entire network. However, service area construction: pushing service advertisement in network, and service area reconstruction: updating the changes caused by dynamic attributes or inherent mobility, in whole network would result in excessive bandwidth and memory consumption. Furthermore, service discovery conventionally considered as an application layer function provides only address of SP and subsequent discovery of route to the service by the underlying routing protocol requires another exchange of messages which results in significant communication and memory overhead along with huge latency.

To enable efficient (timely) service discovery in ADCS, novel idea of steady expanding search coupled with snail's pace advancement and swift boundary confining mechanism is proposed. Moreover, the proposed pull-push approach defines the flexible service area construction and reconstruction leading to better memory and bandwidth utilization. Also, our cross-layer and context-aware approach not only improves the efficiency in terms of bandwidth/memory utilization and the overall latency, but also ensures context-aware service discovery by exploiting collaborative processing of dynamic contexts.

The rest of paper is structured as follows: Next section illuminates underlying basics of CCCIE and CPS; and exhibits the system architecture of ADCS. Section 3 elaborates progressive ripple-based community construction. Section 4 includes evaluation and simulation results showing effectiveness of the proposed technology. Section 5 reviews related work, while last section draws conclusion and outline future work.

2. Community Context-Oriented Collaborative Information System: Overview and Basics

2.1 Context-Attribute Based Community Pervasive Services

The rapid headway in domains of mobile and ubiquitous computing has realized the significance of context-aware systems. Existing context-aware systems gather logical and physical contexts in *service-consumer logic* at end devices, embed them with requests, and dispatch to suitable service provider [9]. Obtained directly from users or deduced from their previous interactions with system, logical contexts include information like user interests/privileges, service expiry time, identity information etc. On the other hand physical contexts are gathered directly from devices, and include information like location, time and device characteristics etc. The logical and physical contexts (static or dynamic) are processed only at SP by selecting appropriate *local business logic* to satisfy the users' requirements [9]. However, apart from physical and logical contexts, we identify a class of composite contexts that are either inferred from former ones or include infrastructure and network dynamic attributes, application and communication related QoS aspects. More importantly, unlike some static contexts, all composite contexts are dynamically determined at time of lookups and require collaborative processing among nodes -rather than processing merely at requester or service provider.

Besides primitive dynamic physical context of location and logical context of users' interests, CPSs encompass various composite contexts, depending on particular class (Fig. 1) of CPS, like time-distance[†], application and communication related QoS aspects, application execution contexts at end-devices, dynamically varying infrastructural properties etc. [21]. As shown in Fig. 1, CPS could be classified according to three criteria: coverage area, information delivery and application support [21]. Generally, wireless ad-hoc based or cellular based CPS include time-distance oriented m-commerce services offered in towns/city [15], on-the-fly delivery of dynamic application-oriented CPS, data sharing services providing environmental information gathered from static or mobile sensors attached to devices [21]. Examples of time-distance oriented m-commerce CPS are time-sales application [3] or restaurant discovery at lunch time by businessmen in certain area with common interests considering composite attribute (time-distance) and logical or physical attribute like user's preferences (price, cuisine, parking space, seats availability), location, service expiry time. Likewise, the examples of application oriented CPS include large file sharing (code distribution, streaming) in ubiquitous ad-hoc environment [29], [30].

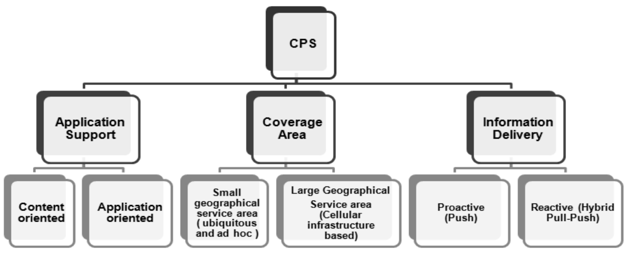
CPS
Application Support
Coverage Area
Information Delivery
Content oriented
Application oriented
Small geographical service area (ubiquitous and ad hoc)
Large Geographical Service area (Cellular infrastructure based)
Proactive (Push)
Reactive (Hybrid Pull-Push)

Fig. 1 Taxonomy of CPS.

[†]Time-distance discerns average estimated time required reaching from current location of user to service provider considering aspects of traffic, weather and topographic conditions of city/town in cellular or WLAN based CPS in city or small town [3], [21].

2.2 Dynamic Grouping Based ADCS

ADCS employs dynamic grouping to support flexible deployment of concurrent services at network level. In presence of large number of services, dynamism due to user's mobility and their change in the interests/demands and variation in context-attributes calls for service deployment, deletion and re-deployment continuously. Consequently, group members make dynamic adjustment (join/leave per group) accordingly. The group of nodes which possess information of same CPS forms one "community" (Fig. 2). If the user triggers the request for any service whose information is not available at its respective base station, the base station node performs discovery mechanism to become member of that service (community). Groups may overlap with each other as one node may join multiple communities.

Unlike proactive service dissemination periodically [4], service contents are reactively disseminated when community members make dynamic adjustment. Moreover, in case of application-oriented CPS, service contents are selectively disseminated to only associated nodes to reduce memory and processing burden on Base Station (BS)/backbone nodes. This is important when there is large number of community services running in the network, with every BS node hosting fraction of them.

2.3 ADCS Architecture

Based on the concept of ADS [1], Autonomous Decentralized Community System (ADCS) architecture is proposed where all nodes coordinate and cooperate to procure timely service delivery to mobile users having common interests. Since in highly dynamic environment the state of the nodes, the stability of connections, the status of SPs are highly unpredictable and maintaining repository of the SP addresses cannot guarantee high-assurance of the system, therefore, to meet the adaptability requirement in constantly and rapidly changing operating environment of ADCS, application-aware adaptive content-code communication [1], [2] is employed (Fig. 2).

ADCS leverages mechanism to publish, discover and access static and mobile services taking into account various surrounding contexts. The roaming users carrying their handsets in their respective *cells*, share services. Besides mobile services, ADCS also supports various static time-distance oriented m-commerce CPS. Each base station node, referred as node, keeps information about neighboring nodes with respective link cost of dynamic context (e.g. time-distance) in Neighbor Node Table (NNT). Each node also possesses *monitoring and management module* (MMM) to monitor the user's presence and manage their published services, and *Service Registration Module* (SRM) to store all published services in corresponding cell. Moreover, ADCS employs underlying middleware to recognize raw contexts, update them, infer composite contexts from raw contexts by inference engine and store all of them in *context repository*. Therefore, each node judges autonomously to join/leave the community network based on its user's preferences and other surrounding contexts. Furthermore, in order to enable application oriented CPS considering resource-constrained mobile devices, each BS node is assumed to support embedding μ-jini proxy [26], which serve clients as virtual Thin Client (VTC) server for service delivery. The overall *ADCS* system architecture is loosely-connected, loosely-control and self-adapting.

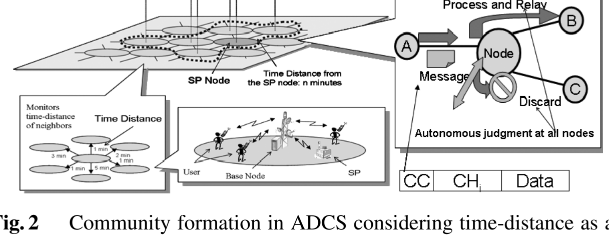

Fig. 2 Community formation in ADCS considering time-distance as an example of context-attribute.

3. Autonomous Pull-Push Community Construction Technique

Autonomous pull-push community construction aims at dissemination of service contents to facilitate users with same requirements, considering message cost and timeliness, in dynamic operating environment of ADCS. Service dissemination in ADCS could be classified into *push based* and *pull-push based*. In *push based* approach, SP periodically multicast the service to current community members (nodes having same users' demands) through leaders of sub-communities [4]. The intuition behind this technique is that members are densely distributed, service contents are highly dynamic and group is relatively stable. In case service contents are static and groups (communities) are highly dynamic or the members are sparsely distributed, we define pull-push mechanism where new member search the existing group members/SP to retrieve the service contents in order to overcome communication overhead of *push based* services. Inspired by the cooperation among social communities, the service discovered by one node in ADCS is shared among other nodes to improve timeliness and reduce network cost. The set of nodes which store/cache service contents construct the community service area. Next paragraph outlines main problems involved in community construction process.

All existing directory-less SDPs (both overlay based and overlay-less) employ flooding scope parameter to confine the broadcast/multicast traffic in process of locating or advertising any service. Since in *n-ring model* successive iteration re-visits the area covered by previous one, and certain wait enough to reach the response to requester before starting new ring is introduced, the efficiency in terms of timeliness or message cost deteriorates significantly. Also each ring introduces certain wait proportional to ring size, to ensure reply message can reach from responder (if exists) to requester, adds further latency. In this regard analysis in [10] shows that the expansion ring scheme used to define flooding scope parameter can't decrease expected latency; rather it increases both cost and latency dramatically. It also shows that even having knowledge of total number of nodes and maximum hop number, cost saving of even the optimal scheme is negligible and the simple searching scheme which is to search the entire area only once actually is the best with respect to both cost and latency. Likewise, SDPs in area of mobile computing [5], [8], [16], [17] either broadcast request message in whole network to find the service or send advertisement packets in entire network, resulting in not only inefficient bandwidth and memory usage but also end up with broadcast storm problem. These factors foster need for addressing this crucial problem with fundamentally new approach.

3.1 Autonomous Progressive Ripple Based Community Service Discovery (PRCSD)

Although significance of Service Discovery Protocols (SDPs) is prevalent among all wireless and wired network domains, the recent advents of web 2.0 and cloud computing has further accelerated the need for efficient (timely but involving reduced message cost) service discovery protocols. Therefore, proposed PRCSD technique aims at timely discovery with reduced network cost.

3.1.1 Overview

PRCSD involves autonomous expansion of the searching region using snail's pace progressive advancement of ripple (search ring). More specifically, when requesting node triggers discovery process, its neighbors process this request, wait for certain time to slow down propagation of request message, and then forward the request to their neighbors like original requester (if they can't satisfy the request). The set of nodes that forward request message constitute *Progressive Ripple Zone* (PRZ). In order to thwart expansion of *PRZ* to the entire network, reply messages triggered by SP propagates quickly (faster than request message) to suppress further propagation of request ripple. Thus, snail's pace advancement of request message coupled with steady expanding ring shifts the task of expanding a searching region from the requester to the forwarding nodes. In other words, all nodes coordinate and cooperate for required optimization parameter of timeliness/message cost in service discovery. Depending on the node resources, service contents are cached either by all nodes (exhaustive caching) or selectively cached [24] in PRZ, to satisfy potential incoming users. Note that reply/suppress message is forwarded by only those nodes who already received corresponding request message. To enable context-aware discovery, waiting time at each node is decided pertaining to position of node from the requester in case of logical and physical contexts, while in presence of composite attributes their numerical values are utilized. The whole process is shown in Fig. 3.

In case of wireless ad-hoc based application-oriented CPS where nodes are memory constrained, information contents are sent to only requester, while short service advertisement message suppresses progressive request ripple. Amid service advertisement propagation, the actual service information is forwarded to the requester instantly. Note that in contrast to conventional SDPs where service advertisements or requests are propagated in entire network by push or pull mechanism [8], [16], pull-push approach of PRCSD defines flexible area of network advertisement/service dissemination, depending on speed of request message, thus reducing significant network overhead.

Moreover, contrary to two-phased conventional layered approach where SDP being application layer protocol finds the address of service provider and passes it to routing layer to acquire service information by suitable path, PRCSD employs one-phase cross-layer approach to further reduce the network overhead and latency. More specifically, contrary to existing cross-layer approaches [16] which are tightly coupled with specific routing protocol and involve dependency of name/resource resolution protocol, proposed approach favors an entirely reactive approach supporting application level specification of destination. In nutshell, compared to SDPs based on n-ring model, progressive ripple based context-aware technique procures timely service discovery along significant reduction of network overhead.

3.1.2 Detailed Operations

ADCS employ content code communication [2] to meet adaptability requirement and to enable language-independent service discovery. Figures 4 and 5 depict format of service discovery request message and reply/suppress message respectively. Here, *Content Code* (CC) identifies service name while *Characterized Code* (CH_i) specifies service attributes e.g. product price, its manufacturing/expiry date, SP location etc. to leverage semantic enabled searching. Amid propagation of request message *HopCount* increments, which is used to determine appropriate waiting time at each node to enable service discovery in case of physical and logical contexts. The parameter *RequestMessageId* in reply message describes which service query this reply

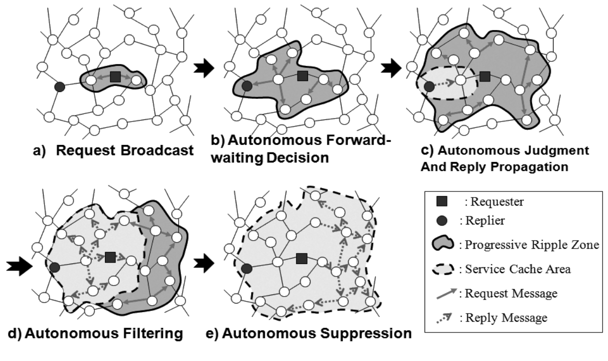
a) Request Broadcast
b) Autonomous Forward-waiting Decision
c) Autonomous Judgment And Reply Propagation
d) Autonomous Filtering
e) Autonomous Suppression
■ : Requester
● : Replier
: Progressive Ripple Zone
: Service Cache Area
: Request Message
: Reply Message

Fig. 3 Illustration of progressive ripple-based discovery.

message corresponds to.

During service discovery process, each node may take any of the following five states: *Normal, Request-Processing, Request-Forward-Waiting, Reply-Waiting and Message-Absorbing*. State transitions among different states are shown in Fig. 6. Given below are the detailed autonomous operations performed by each node, along with the descriptions of states a node may acquire.

a) Autonomous Request Monitoring: In Normal state each node keeps on monitoring the arrival of request message. Upon receiving request message, the node autonomously registers its content code along ReqMsgId, and sets all neighboring nodes, except the source node, as destination in NNT.

b) Autonomous Request Processing: In *Request-processing state*, node processes the received request message and autonomously judges, whether the node can reply to this request or not, based on the content code information available in *service registration module*. In case of inability to satisfy request, it enters into *request-forward-waiting state*; otherwise, it creates the service- reply message, sends it to all neighbors listed in NNT, and enters into *Message-Absorbing state.*

c) Autonomous Request Forwarding: In order to enable key concept of sedate advancement of request message, each node in *Request-forward-waiting state* autonomously introduces delay T_w: the length of the request forward waiting timer. In case of composite contexts-attributes, each node decides T_w proportional to respective context attribute value taken from its *context repository*. The usage of composite contexts like communication load 'enforce' the least loaded path to propagate service contents towards requester, while time-distance return the cumulative time-distance of path which require least time to reach the location of SP [21]. Likewise, in case of physical and logical contexts, each node autonomously introduces delay pertaining to its position from the requester to enable semantic service search. Upon expiry of this timer, the node forwards the request message to neighboring nodes listed in NNT. If request message with same content code and *ReqMsgId* is received through another neighbor, it is autonomously discarded.

d) Autonomous Replying: During *reply-waiting state* the node autonomously waits for receiving the reply message, corresponding to the request message it has forwarded. More specifically, a set of nodes in this state (or forwarders) constitute *Progressive Ripple Zone* (PRZ).

Upon reception of reply message the node autonomously forwards this message to all neighboring nodes except the sender of reply message, and enters into *Message-Absorbing state*. As there is no delay in *Reply-waiting state*, it helps reply message to pursue the request message to swiftly confine the propagation of request ripple. Each node caches *ReqMsgId* for certain time in order to discern whether reply message should be accepted and forwarded in *PRZ*. If the *ReqMsgId* specified in reply message doesn't correspond to stored information of request message (content code and *ReqMsgId*), this reply message is discarded and not forwarded further toward any neighbor. This helps selective propagation of suppress message only in PRZ.

e) Autonomous Message Suppression: Before expiry of timer T_w, if node receives reply/suppression message, it autonomously stops forwarding both request and reply messages by entering *Message-Absorbing State*. Likewise, node goes from *Request-processing* or *Request-forward-waiting* to *Message-Absorbing State* in order to avoid re-entering the cycle of the service discovery. During this state node discards both kinds of received messages. Moreover, upon expiry of timer T_{Absorb} it autonomously deletes entry of the forwarded request message from NNT, and goes to *Normal* state.

Content Code	CH_i	Requester Node ID	Request Message ID	Sender Node ID	Hop Count

Fig. 4 Format of request message.

Content Code	CH_i	Reply MsgID	Replier Node ID	Request MSgID	Request NodeID

Fig. 5 Format of suppress message.

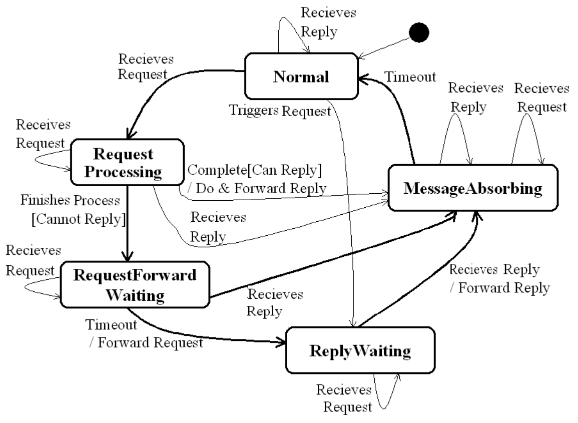

Fig. 6 State transition diagram.

3.2 Autonomous Service Area Construction Technique

Management of service information, an important facet of any SDP, includes aspects like where to store service information, service advertisement diameter, time duration for which service information will be retained etc. [8]. In community SDP service area construction means network area in which discovered service information or the actual contents will be stored/cached to expedite timeliness and reduce broadcast traffic. Surveys [8], [16], [17] reveal that almost all decentralized SDPs in area of mobile computing like one described by Cheng and Marsic, Varshavsky, Reid et al. and Konark advertise services in whole network; while few like GSD, Field Theoretic approach advertise services only up to certain advertisement diameter. However, these protocols don't involve autonomous decision at each node, but merely take it as global network parameter. In our approach the discovered service information or actual contents are shared, either among all nodes present in PRZ (exhaustive caching) in case of content oriented CPS or selective caching [24] in case of application-oriented CPS, implicitly by reply/ suppression message. To locate the service, nodes first inspect their own registry and in case it is not available they broadcast request to find service information from nearest community member. The size of service area construction depends on average (T_w) and relative position of users from nearest responder (users' temporal and spatial distribution). Note that, from user point of view, discovery time by employing PRCSD; and response time due to service area construction will significantly improve as compared to accessing SP individually by each requester's node using n-ring method. For example, the time-bounded sale offered by chain-store or nearest restaurant lunch service in busy business street is discovered by one node and if it is shared among other nodes, having same requirements, in certain zone it will result in significant improvement of response time and reduction of network cost. Note that if all users broadcast their request individually, it will result in broadcast storm (may result in discovery failure), along with high delay to all users during service acquisition.

In case of CPS that involves cumulative value of composite contexts attributes (e.g. time-distance from SP, or network load on path to SP) require updation of context upon detection of significant change at any point. However, updating cached contexts in presence of minor change generates huge traffic rendering system highly unstable, while updating sporadically compromises the accuracy of system. Therefore, each node autonomously decides the threshold CA_{th}, according to its cumulative value of context-attribute from SP, by triggering updation-thread to update information from respective node to down whole tree (For details refer [14]). Each node determines CA_{th} as $\delta / \sum_{i=0}^{l-1} \gamma_i$ where δ is constant like cache holding time and $\sum_{i=0}^{l-1} \gamma_i$ is cumulative value of context-attribute from SP to respective node. Moreover, since community construction involves cross-layer implementation, route-updation also leads to context-updation. Updating changes of these contexts are termed as reconstruction of community [14].

4. Evaluation

Assuming network to be a circular area with uniform random distribution of nodes, all located within M hops of the requester placed in centre, the expected number of nodes N_j that are exactly j-hops away are $q(j^2 - (j-1)^2) = q(2j-1) = N_j$ where q is number of nodes in first hop from requester. The total number of nodes N_T located within the circular region of M hops from the requester become $\sum_{h=1}^{M}(2h-1)q = qM^2 = N_T$. Considering T_p as processing time and τ as transmission time at each node, the time a flooded packet takes to reach a node k hops away is $k(T_p+\tau)$. In n-ring scheme, if the SP is j hops from the source node and hop limit $h_{lm} < j < h_{lm+1}$, for $lm \in \{0 \dots n-1\}$, then to find the SP (S), the source node has to fail for the first lm times, taking $2h_{lm}(T_p+\tau)$ time for the mth attempt $m \in \{1 \dots lm\}$ and succeed at the (lm+1)st attempt, that takes $2jT_p + \tau$; time. Thus total latency to search SP present j-hop away in n-ring scheme is $L_j = \sum_{lm=1}^{k}(2h_{lm}+2j)(T_p+\tau)$. The probability that a SP is j hops away from requester is $\rho = N_j/N_T - 1$ for $j \in \{1 \dots M\}$ and the average response time R_T to search a random SP in n-ring scheme becomes:

$$R_T = \left[\sum_{j=1}^{M} 2N_j j + \sum_{lm=1}^{n-1} \sum_{j=lm_k+1}^{h_n} (2h_{lm}N_j)\right](T_p+\tau)/N_T - 1 \tag{1}$$

Compared to n-tier scheme, progressive ripple based discovery technique has average response time R_T:

$$R_T = \sum_{j=1}^{M}(2T_p + T_w + 2\tau)j * \frac{N_j}{N_T - 1} \tag{2}$$

Where, T_w is forward-waiting-time at each node. Assuming T_p and τ are constant at each node T_w becomes key parameter to determine best response time with least message cost: smaller the T_w better the response time and vice-versa; but smaller T_w results in higher message cost. As it is hard to predict number of incoming users of any CPS, intuitively it is important maintain flexibility: make small size of community (PRZ). In order to balance this tradeoff or making initially small size community without deteriorating response time for initial users, next section delineates autonomous and dynamic adjustment of T_w. Comparison of (1) and (2) depicts clearly that proposed technique has significant improvement over n-ring method. In phase of community construction suppose total number of nodes (Ω) in community area cache the discovered contents (thus total SPs β become $S + \Omega$) and the X be random variable representing nearest community member β_i located $j+1$ hops away, R_T further reduces to

$$R_T = \sum_{j=1}^{M}(2T_p + T_w + 2\tau)j * \left[1 - \frac{j^2}{M^2}\right]^{\beta} \tag{3}$$

4.1 Simulation Results

To evaluate the performance of our approach, 6d-regular mesh graph is used where total number of nodes in system could be expressed as $f(n) = 3n^2 - 3n + 1$. Here, n is number of levels in regular hexagonal mesh graph. Simulations were conducted to verify the behavior and response time of proposed technique where response time is equal twice the discovery time. Table 1 outlines the key simulation parameters along respective values. OMNeT++3.3 [28] framework was used for simulations.

4.1.1 Simulation I

This simulation aims at evaluation of PRCSD using T_w based on both composite and physical/logical contexts, and compares with native n-ring method with respect to response time. Furthermore, it shows how to tune T_w considering tradeoff between timeliness and flexibility.

Here, 'D' is the total number of hops between fixed central requester and replier whose position was varied from 1st to 15th hop. Values of T_w in case of dynamic contexts were assigned at each node according to normal distribution with $\mu = 3$ and $\delta = 2$. For logical and physical based context-aware discovery, T_w were assigned values of 2, 4 and 8 milliseconds.

Figure 7 shows response time of the proposed technique is proportional to the distance 'D' (number of hops) between a requester and a replier. However, the response time for *n-ring* scheme worsens drastically as the 'D' increases. Note that cross-layer implementation approach was done in case of both n-ring and proposed schemes. The results of delay according to dynamic contexts were averaged over 15 simulations runs. Quantitative analysis shows that compared to n-ring, PRSD shows on average improvement of 49% with $T_w = 2$, 42.5% with $T_w = 4$, and 29% with $T_w = 8$. Considering the tradeoff between response time and message cost (considering only current request) decreasing T_w from 8 to 2 results in average improvement of 28% in response time at cost of 49% increase in message cost.

Figure 8 reveals that there exists trade-off between user's requirements of timeliness and system requirements of flexibility; therefore, a mechanism is needed to maintain the flexibility of system without compromising response time. Note that number of messages here reflects the extent of flexibility/adaptability. The large number of messages means bigger PRZ or less flexibility in the system and vice versa. One may observe that small T_w not only results in timely service discovery for initial users but also for future potential users as contents will be shared in large PRZ; however, in presence of small number of users of many CPS, it will result in excessive bandwidth consumption of the system. Contrary to this, when T_w is set to large value the total message cost is small and thus contributes to better flexibility but response time deteriorates. It is worth mentioning that if only service discovery is considered (no service area construction) then this trade-off is between timeliness and number of messages incurred during single discovery. Quantitative analysis shows that increasing T_w from 1 to 40 shows an improvement of 62% in message cost at cost of 72% decrease in response time. Thus, it is imperative to think about assigning appropriate T_w to discover service timely but with least message cost/flexibly. Given that distance between requester and responder increases, quick request message results in higher cost, but it is more logical to discover SP quickly if it lies nearby. Therefore, each node should autonomously and dynamically decides T_w with respect to total hops covered by request message. In order to illustrate the impact of T_w for effective discovery, simulation was setup with following condition to achieve optimal value in given setup.

$$T_w = \begin{cases} \text{Number of hops (the number of hops} < 8) \\ 7 \text{ (number} >= 8 \text{ hops)} \end{cases} \quad (4)$$

As depicted in the Fig. 9 that when each node adjust T_w dynamically according to criteria of Eq. (4), response time is almost same as static $T_w = 4$. Likewise, as shown in Fig. 10 that autonomous dynamic forward-waiting adjustment time brings total number of messages almost close to the case when static $T_w = 8$. This shows the effectiveness in terms of improving both timeliness and flexibility, if T_w is set according to position of node between requester and replier.

4.1.2 Simulation II

These simulations were carried out to show how pro-

Table 1 Simulation parameters.

Parameter	Description	Value[unit]
τ	Transmission delay	1[ms/msg]
T_{Absorb}	Time limit to accept incoming messages	50 [ms]
$\boldsymbol{T_P}$	Processing time at each node	5 [ms/msg]
N_T	Total No. of nodes	721
h_i	Hop limit for n-ring scheme	5,10,15

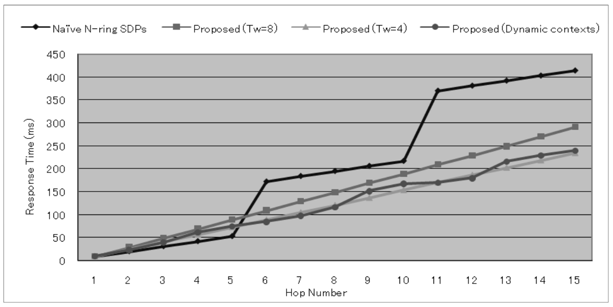

Fig. 7 Comparison of response time.

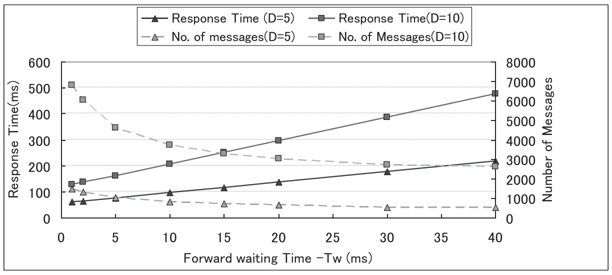

Fig. 8 Trade-off between response time and message cost.

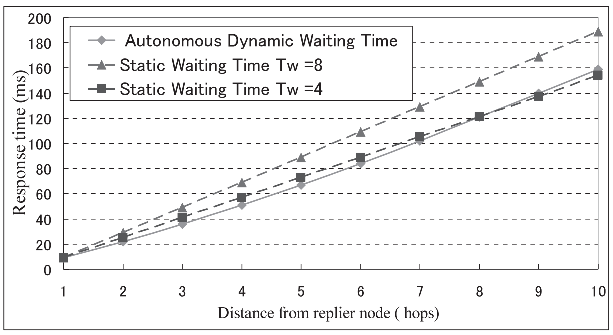

Fig. 9 Dynamic adjustment of T_w for lower latency.

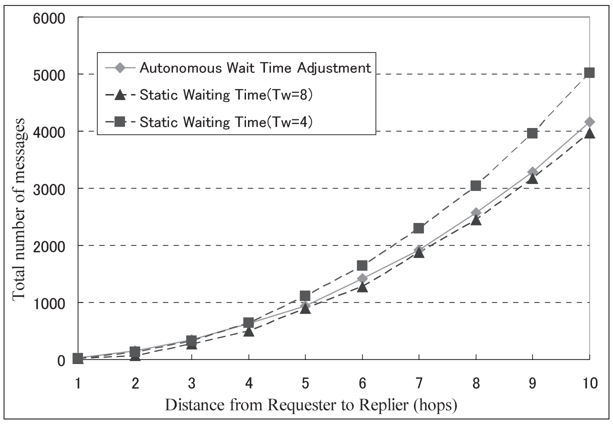

Fig. 10 Dynamic adjustment of T_w for lower message cost.

posed technique performs in case of selective caching of application-oriented CPS in resource constraint environment. Here requester was placed in the centre, while CM community members were uniformly distributed in the network. Note that T_w was set considering logical and physical context attributes.

Figures 11 and 12 illustrate the network traffic and response time averaged over 100 simulation results. Figure 11 shows that as CM increases, response time improves; hence, in presence of multiple virtual service providers/ community members, response time is dependent on total number of CM and forward-waiting time. Likewise, Fig. 12 depicts that total network traffic is function of number of community members CM and T_w. Here, message cost reduces by increasing T_w at each node, for example, with community members $CM = 8$, increasing T_w from 2 to 32 results in 80% decrease in network traffic at cost of 121% increase in time delay. Our results also indicate that increasing community members above than 15 and setting T_w above than 16 results in 2 fold reduction in network traffic compared to n-ring model. This shows that proposed discovery mechanism could be tuned to discover service in grouping network scenarios of bandwidth constrained ubiquitous environments where low traffic generation is required.

It is important to note from combined results of Figs. 11 and 12 that in presence of sufficient CM/users, both timeliness and message cost will improve as requests are not only satisfied by original requester but other members too.

4.1.3 Simulation III

This simulation exhibits the effectiveness of exhaustive caching (sharing of service by all nodes in PRZ) of discovered service, which is the case of content based time-distance oriented CPS.

It could be observed from Fig. 7 that as compared to naïve n-ring method effectiveness of PRCSD increases sharply as the distance between SP and user increases. However, in order to show the average improvement from user point of view in practical scenarios, 500 users were uniformly added (spatially) in the network (721 nodes) at rate

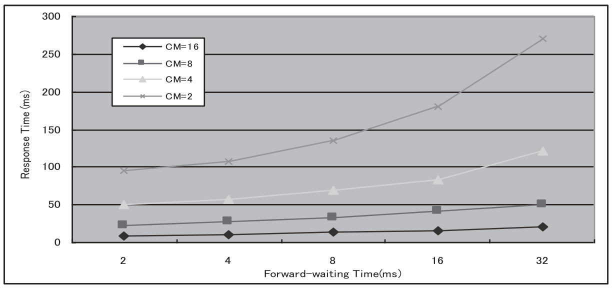

Fig. 11 Response time considering multiple community members.

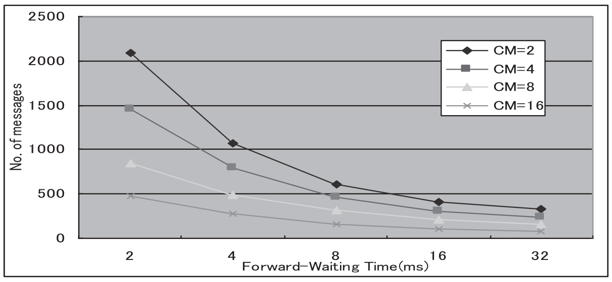

Fig. 12 Message cost considering multiple community members.

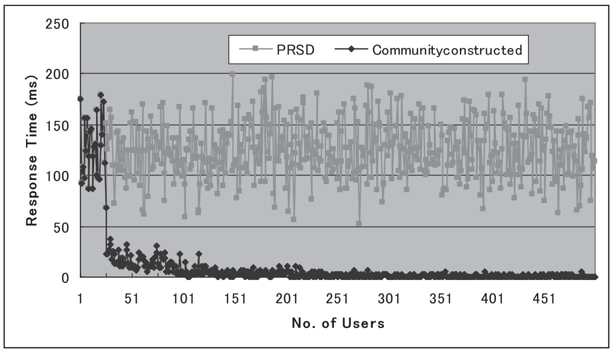

Fig. 13 Response time comparison (PRSD & community-construction).

of 25 users per minute. The SP was placed in the middle and T_w was set dynamically according to Eq. (4). The results were averaged over 20 simulation runs. Figure 13 depicts that after addition of initial 25 users response time improves significantly due to sharing of contents in all PRZs. Note that as same request is triggered simultaneously by number of nodes, *Message Absorbing state* lets only one request to be forwarded thus reducing significant network traffic.

Analysis reveals that mean of average of all simulation runs (of response time) in case of PRCSD is 125 ms, while once community has been constructed it reaches to 11 ms; and average of n-ring (not shown in graph) was 355 ms (with 5 rings each with 3 hop limit). Thus as compared to n-ring with given setup, PRCSD shows average improvement of 64% while community construction brings improvement of 91% over PRCSD. Thus from user point of view community construction ensures significant improvement in both discovery time and response time.

5. Related Works

Location Based Systems (LBS) provide services based on the geographical location of the ubiquitous mobile devices [6], [7], [25] but they don't consider provision of customized services based various context like time aspect and users interests. The group-keyed service-oriented dissemination in ADCS is different from application level multicasting [23] where all dynamically changing contents are periodically multicast to satisfy demands of all group members. Whereas, ADCS involve dynamic grouping of nodes due to varying surrounding situations but service contents are static most of the time. Moreover, dynamic grouping in ADCS is different from overlay based systems [23] as in ADCS nodes only have addresses of physical neighboring nodes.

Service discovery has remained well-studied problem in both wired and wireless networks. Existing service discovery protocols are mainly categorized based on two criteria: architecture and layer-implementation [16]. Regarding architecture, they are further divided into centralized directory based and decentralized/directory-less. Centralized directory based, protocols like JINI, Salutation, UPnP, UDDI and SLP mainly target service discovery in wired stable infrastructure networks [16], while Lanes, Service Rings, SSD, DSDP perform service discovery in mobile computing environments [5], [8], [16], [17]. Since centralized protocols are less robust against failures, various attempts with decentralized approach were taken. Among these, protocols like Allia [8], [16] take overlay implementation to control the flooding but they incur huge overhead due to bootstrapping phase and overlay maintenance. Therefore, most of decentralized based protocols like DEAPspa -SD, DSD [16] take overlay-less implemece, Konark, PDP, Allia, GSD, AODV-SD, M-ZRP, ODMRPntation to discover service by either pushing unsolicited advertisement or multicasting/broadcasting request of all services in entire network which result in inefficient utilization of memory and bandwidth. Likewise, existing distributed context-aware service discovery techniques enable physical and logical contexts based service discovery without specifying the search limit [27].

On the basis of layered implementation SDP have either implementation at application layer or network layer (cross-layer). All above mentioned distributed protocols are application layer while network layer protocols include AODV-SD, LSD, M-ZRP and ODMRP. The existing cross layer based implementations are routing protocols specific and employ expansion n-ring scheme to find target node through pull mode or share the services proactively in small area. Unlike them, our approach employs pull-push mechanism along application-aware adaptive content-code routing implementation.

Since service or resource discovery is general problem in various networks, different analytical attempts [11], [12] have focused on finding optimum search radius in n-ring scheme; but these all require global knowledge of maximum hop number and assume uniform distribution of nodes for optimization which make n-ring scheme less practical. Notable reactive routing protocols in area of ad hoc network like DSR and AODV [18] also employ expansion n-ring scheme to find target node. Moreover, target discovery in Peer to Peer systems [19] and distributed computing systems [20] for load distribution and efficient discovery of web services [22] only TTL based n-ring model is used.

Unlike all above directory-less approaches, community service discovery protocol has three novelties: a) snail pace advancement of request ripple followed by quick follow up of reply message to suppress request ripple for controlling broadcast and achieving timeliness; b) context-aware cross-layer approach for enhancing timeliness; c) efficient in terms of bandwidth and memory utilization as it involves pull-push mechanism to define the service area rather than push or pull or explicit push-pull mechanism of all other SDPs [8], [16], [17].

6. Conclusion and Future Directions

This paper has introduced concept of *Community Context-attribute-oriented Collaborative Information Environment* (CCCIE) to enable service provision in accordance with users' situations (location and time) and their varying preferences. Moreover, *Autonomous Decentralized Community System* (ADCS) architecture was discussed for provision of *CPS* to mobile users.

The pull-push autonomous community cross-layer and context-aware approach not only improves the efficiency in terms of the overall latency and bandwidth/memory utilization, but also ensures selection of best routing path or facilitate users with quickest accessible service by exploiting collaborative processing of dynamic contexts. Unlike conventional n-ring TTL model based SDPs, the proposed technique satisfies the assurance property by fulfilling requirements of online-properties, flexibility and timeliness. The average latency of proposed method is much lower than n-ring model which is further reduced by taking cross-layer approach. Moreover, contrary to n-ring TTL based SDP, where requester or replier optimize the size of ring in centralized controlled manner, proposed technique involve all nodes to coordinate to achieve flexible service area construction by tuning the key parameter T_w-considering trade-off between timeliness and message cost. In case of both context-aware and context-less delay, the smaller average T_w achieves timeliness not only for the requester but also for potential future users; however, larger average delay T_w results in bandwidth and memory efficiency. Likewise in case of context-less T_w it was shown that autonomous decision by each node pertinent to requester achieves timeliness with less message cost as compared to having constant delay at each node. Owing to its language independence and underlying generalized reactive routing protocol, proposed technique could be tailored for IP-based target discovery in any network, can be easily extended for semantic level search in areas of P2P architectural based web services, and also for resource discovery in peer to peer networks where IP address are not available e.g. WSN.

As the current implementation is based on infrastructure based cellular and wireless LANs, we plan to extend implementation of proposed service discovery technique for ad hoc wireless network with different MAC protocols. Moreover, in presence of multiple service instances, context-aware service selection is an issue that can't be handled by n-ring model; therefore, we plan to extend our work in this direction too.

References

[1] K. Mori, "Autonomous decentralized system and its strategic approach for research and development," IEICE Trans. Inf. & Syst., vol.E91-D, no.9, pp.2227–2232, Sept. 2008.

[2] K. Mori, "Towards integrated methods for high-assurance systems," Computer, vol.31, no.4, pp.32–34, April 1998.

[3] T. Ono, N. Kaji, Y. Horikoshi, H. Kuniyama, K. Ragab, and K. Mori, "Autonomous decentralized community construction technology to assure quality of services," Proc. IEEE FTDCS, pp.299–305, 2004.

[4] K. Mahmood, X. Lu, and K. Mori, "Autonomous community construction technology to achieve service assurance in ADCS," IEICE Trans. Inf. & Syst., vol.E91-D, no.9, pp.2259–2266, Sept. 2008.

[5] F. Zhu, M.W. Mutka, and L.M. Ni, "Service discovery in pervasive computing environments," IEEE Pervasive Computing, vol.4, no.4, pp.81–90, 2005.

[6] I.A. Junglas and R.T. Watson, "Location-based services," Commun. ACM, vol.51, no.3, pp.65–69, March 2008.

[7] A. Kupper, S. Helal, and P. Bellavista, "Location-based services: Back to the future," IEEE Pervasive Computing, vol.7, no.2, pp.85–89, April 2008.

[8] A.N. Mian, R. Beraldi, and R. Baldoni, "A survey of service discovery protocols in multihop mobile ad hoc networks," IEEE Pervasive Computing, vol.8, no.1, pp.66–74, 2008.

[9] C.B. Anagnostopoulos, A. Tsounis, and S. Hadjiefthymiades, "Context awareness in mobile computing environments," Wirel. Pers. Commun., vol.42, no.3, pp.445–464, Aug. 2007.

[10] Z. Cheng and W.B. Heinzelman, "Searching strategies for target discovery in wireless networks," Ad Hoc Networks, vol.5, no.4, pp.413–428, Elsevier, May 2007.

[11] N. Chang and M. Liu, "Revisiting the TTL-based controlled flooding search: Optimality and randomization," Proc.10th MobiCom, pp.85–99, Philadelphia, PA, USA, 2004.

[12] B. Krishnamachari and J. Ahn, "Optimizing data replication for expanding ring-based queries in wireless sensor networks," WIOPT, pp.361–370, Boston, April 2006.

[13] Z. Cheng and W.B. Heinzelman, "Flooding strategy for target discovery in wireless networks," Proc. International Workshop on Modeling Analysis and Simulation of Wireless and Mobile Systems, pp.33–41, San Diego, California, USA, Sept. 2003.

[14] K. Mahmood, X. Lu, Y. Kanamaru, and K. Mori, "Autonomous decentralized community construction technology for high quality information service," Proc. 28th ICDCS Workshop, pp.563–568, June 2008.

[15] K. Mahmood, Y. Horikoshi, S. Niki, X. Lu, and K. Mori, "Progressive ripple-based service discovery or high response time in autonomous decentralized community system," 12th IEEE FTDCS, pp.81–87, Oct. 2008.

[16] J. Su and W. Guo, "A survey of service discovery for mobile ad hoc networks," IEEE ICCCAS, pp.398–404, 2008.

[17] Z. Gao, Y. Yang, J. Zhao, J. Cui, and X. Li, "Service discovery protocols for MANETs: A survey," Proc. 2nd Conf. MSN, vol.4325, pp.232–243, Hong Kong, China, Dec. 2006.

[18] M. Abolhasan, T. Wysocki, and E. Dutkiewicz, "A review of routing protocols for mobile ad hoc networks," Ad Hoc Networks, vol.2, pp.1–22, Elsevier, 2004.

[19] "The Gnutella protocol specification V.0.4," www.clip2.com/GnutellaProtocol04.pdf

[20] http://gaia.cs.uiuc.edu/papers/GaiaSubmitted3.pdf

[21] K. Mahmood, S. Niki, X. Lu, and K. Mori, "Autonomous hybrid pull-push context-aware community service dissemination technology to achieve high assurance," Proc. ISADS, Athens, Greece, March 2009.

[22] F.B. Kashani, C.C. Chen, and C. Shahabi, "WSPDS: Web services peer-to-peer discovery service," ISWS, pp.733–743, Las Vegas, Nevada, USA, June 2004.

[23] C.K. Yeo, B.S. Lee, and M.H. Er, "A survey of application level multicast techniques," Comput. Commun., vol.27, no.15, pp.1547–1568, Sept. 2004.

[24] E. Cohen and S. Shenker, "Replication strategies in unstructured peer-to-peer networks," ACM SIGCOMM, USA, 2002.

[25] J. Raper, G. Gartner, H. Karimi, and C. Rizos, "A critical evaluation of location based services and their potential," Journal of Location Based Services, vol.1, no.1, pp.5–45, March 2007.

[26] C. Lee, S. Helal, and D. Nordstedt, "The μJini proxy architecture for impromptu mobile services," SAINT Workshop, pp.4–117, Jan. 2006.

[27] C. Doulkeridis, V. Zafeiris, K. Nørvåg, M. Vazirgiannis, and E. Giakoumakis, "Context-based caching and routing for P2P web service discovery," Distributed and Parallel Databases, vol.21, no.1, pp.59–84, Feb. 2007.

[28] www.omnetpp.org

[29] P.J. Maroron, A. Lachenmann, D. Minder, J. Hahner, R. Sauter, and K. Rothermel, "TinyCubus: A flexible and adaptive framework for sensor networks," European Workshop on Wireless Sensor Networks, Jan. 2005.

[30] T. Heimfarth and P. Janacik, "Ant based heuristic for OS service distribution on ad hoc networks," Biological Inspired Cooperative Computing, vol.216, Springer, 2006.

Khalid Mahmood received the B.Sc (Hons.) and M.S degrees in computer Science from International Islamic University, Islamabad, Pakistan in 2001, 2004, respectively. He joined Department of Computer Science, Tokyo Institute of Technology as research student in 2006 and entered into doctoral course under supervision of Prof. Kinji Mori in 2007. His research interests include autonomous decentralized systems, community cooperative information systems, ubiquitous computing, wireless sensor network, and distributed application architecture.

Xiaodong Lu received the B.S. degree in electronic engineering in 1997, the M.S. degree in computer science in 2000, both from Lanzhou University, P.R.China, and PhD degree in computer science from Tokyo Institute of Technology, in 2005. He is an assistant professor in the Department of Computer Science at Tokyo Institute of Technology. His research interests include distributed and high-assurance information systems and mobile agent.

Yuji Horikoshi received the B.S. and M.S. degree in information engineering from Tokyo Institute of Technology Japan in 2003 and 2005 respectively. His research interests are autonomous decentralized system and mobile computing.

Kinji Mori received the B.S., M.S. and Ph.D. degrees in the Electrical Engineering from Waseda University, Japan in 1969, 1971 and 1974, respectively. From 1974 to 1997 he was in System Development Lab., Hitachi, Ltd. In 1997 he joined Tokyo Institute of Technology, Tokyo, Japan as a professor. His research interests include the distributed computing, the fault tolerant computing, wireless sensor network and the mobile agents. Currently he is head of department of computer science, and Fellow of IEEE, and member of IPSJ and SICE, Japan.