Multiagent Place-Based Virtual Communities for Pervasive Computing

Tuan Nguyen[1], Seng Loke[2], Torab Torabi[2], Hongen Lu[2]
Department of Computer Science and Computer Engineering,
La Trobe University, VIC, 3086, Australia
[1] *at6nguyen@students.latrobe.edu.au*
[2] *{s.loke, t.torabi, h.lu}@latrobe.edu.au*

Abstract

This paper proposes a multiagent based virtual community as a new means to support pervasive computing services. We give a conceptual definition of the concept of Place-Based Virtual (PBV) Community. A PBV-community can be used in an environment of context-aware applications and is where agents can help users' interactions with services within the PBV-community. We have proposed a model for a PBV-Community and implemented a prototype called "student digital assistant" based on the agent-oriented approach. We have developed a multiagent architecture and protocols to realize the PBV-community. Agents in this architecture can cooperate and collaborate to provide different services to end users based on the user's context.

1. Introduction

Pervasive computing or ubiquitous computing is developing rapidly and promises many benefits for people [2],[18]. There are many proposed solutions for solving the issues in pervasive computing. A multiagent methodology is widely adopted [5],[13]. There is also one proposed approach in using the concept of virtual community for pervasive computing [6],[13]. However, according to Porter, there is no consensus among researchers about appropriate formal definitions for virtual communities and types of virtual communities [12]. In addition, there is lack of a common framework for solving the problem.

This paper proposes the novel concept of the place-based virtual community (PBV-Community, for short) based on context-aware ideas in computing, where a digital community is superimposed over a place (or a marked up area).

We consider the environment with users, computing devices, objects, sensors as the community. Members in the community are managed by agents. When users are moving in that environment, users are monitored and are helped automatically based on their context.

For illustration, we have implemented a prototype called "student digital assistant application" that uses the PBV-community concept. We have developed a JADE[1] multiagent architecture to realize the PBV-Community.

The rest of this paper is organized as follows. In section 2, we present a survey on background knowledge and related work. In section 3, we propose a new definition of place-based virtual community based on earlier ideas of the notion of virtual community and methodology to implement the PBV-Community. Section 4 describes the implementations and techniques to address issues in PBV-Communities. The last section discusses future developments and conclusion.

2. Background and Related Work

The main goal of ubiquitous computing is development of applications that can run autonomously and can change its behaviour depending on the user's context [18],[19]. However, building this kind of software is a very complex task. There are many approaches for building pervasive computing applications including: using multiagent system, creating the environment as a virtual community with many different types of components, and using ontology for knowledge sharing and context reasoning. Table 1 shows some of related work using different approaches for building pervasive applications.

	Multiagent System	Virtual Community	Ontology
D-Me [5]	Yes		
MyCampus [15]	Yes		Yes
ActiveCampus[6]		Yes	
CoBra [4]	Yes		Yes
CONSORT [16]	Yes		Yes

Table 1. Approaches for building pervasive applications

Cozzolongo *et al* [5] use the software agent to interact with the environment to exploit the services provided by the environment based on user's context. This project is limited to a "smart to do list" case

[1] JADE – Java Agent Development Frame Work, http://jade.tilab.com

0-7695-3113-X/08 $25.00 © 2008 IEEE
DOI 10.1109/PERCOM.2008.52

study. According to the user's context and a content list, the agent will decide to guide the user what he/she should do. The MyCampus project aims to assist students in different tasks such as planning an evening, organizing study group, recommending places of interest and filtering messages [15]. The context in this project includes places/locations, power and scalability of the environment which are derived from an ontology. However, MyCampus does not discuss the learning community in the campus environment.

In addition, the MyCampus project uses the Semantic Web technology with a virtual community concept to build context-aware services in a university campus [6]. ActiveCampus considers university campus as a learning community where students and teachers can exchange information and interact with each other. In MyCampus, the location information is obtained only by using the Pango product that tracks the 802.11 signals based on a set of pre-calibrated fixed points.

In addition, Sashima [16] has implemented location-aware middle agents for providing intelligent information to users in the spirit of context-aware computing. The location-aware middle agents are responsible for all queries about user's location. This project also uses the JADE framework to implement the agents.

Thus, multiagent system is considered best fit for building distributed and complex applications [7]. However, though in this distributed environment, components or objects can interact with each other tightly and they can share the knowledge and information as in the community. In fact, Weiser first proposed the virtual community approach using MUD technology in [18]. Today, using a multiagent System instead of MUD has shown advantages. In the next section, we propose place-based virtual communities as an abstraction for building context-aware applications. We also propose an agent architecture that suits the definition as a proof of concept.

3. Approach

3.1. The notion of the virtual community

According to the Merriam-Webster dictionary[2], a community is a unified body of individuals or people with common interests living in a particular area, or an interacting population of various kinds of individuals in a common location. The community may be a group of people with a common characteristic or interests living together within a larger society.

[2] Community, http://mw1.merriam-webster.com/dictionary/community

As the number of people connected to the Internet is increasing, people with the same interests, may form a virtual community. Howard Rheingold, author of the book Virtual Communities [14], defines virtual communities as "organized around affinities, shared interests, bringing together people who did not necessarily know each other before meeting online."

Pervasive computing system can include many distributed components such as computers, software agents, and real-world objects. We will define a suitable conceptual definition for virtual community.

3.2 Place-based Virtual Community

After analysing many definitions about virtual community [12], we propose the following definition for *place-based virtual community* with consideration to context-aware computing:

> *A place-based virtual community is a group of people, objects or agents, sharing common interests, attributes, and knowledge, that may share a common physical place at a specific time. Object can be a computational device or just a sensor, or a tag. The community is managed and operated by autonomous agents. Agent communications and co-operations are based on protocols, policies that give benefit to the members.*

The PBV-Communities (PBVCs) may share part of or the same physical place. A PBV-Community (PBVC) may have rules, policies, protocols that members must follow. Each member of the community can have their own privileges or responsibility that can be referred to as roles, obligation, prohibition, and rights.

Members are able to interact with each other and have some basic knowledge about each other. Such knowledge about members includes member's properties or context. The context might be the location of each member, member's interest-profile or user's tasks.

We would like to use the word "place" instead of "location" because when refer to a "location", it is usually a reference to a single point in the coordinate system. Furthermore, people usually refer to "place" in normal conversation; for example, people usually invite their friends to come to their *place* not to their *location*.

In addition, pervasive computing means computing everywhere. Pervasive computing forces computers to co-exist in the real world with people [19]. Computers and objects though distributed everywhere but also organized within a coordinated and managed community so to provide necessary benefits. The knowledge about other members in a PBVC can be very important. By knowing each member's location,

the PBVC can manage and provide suitable services for each user in a certain place.
Figure 1 illustrates this concept that several PBVCs can share the same physical place. A user may belong to many PBVCs at the same time [9], [10], [11]

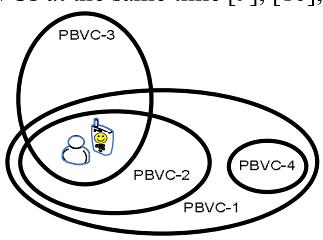

Figure 1. The PBVC Concept

For example, a user who is inside the University campus (PBVC-1) is also in the Library (PBVC-2) and in the study carrel number 2 (PBVC-3). The PBVC-4 can be a bookshop separate from library but still in the main campus. Each different area can be superimposed with a different digital community to provide different services to members. For each community, there will be at least one agent taking care of the members. A task can be *check-in* or *exit* from the community, which can be paralleled to the user walking in or out of the "place". There are also many software agents within the PBVC, which provide services to users.

The PBVC superimposed over a place might also depend on the time, occasion or activity. For example, many clubs and societies share a hall in the sport centre of the university. In the morning, the hall belongs to the volleyball club but in the evening, the hall belongs to the soccer club – with a different digital community superimposed over the same place in the different times.

Users can choose to use the services provided by the communities or just ignore them. A user can also ignore the distracting services by specifying his/her immediate task that can be used as a filter (so that only services relevant to the task are displayed to the user on the user's mobile device screen).

3.3. Protocols

According to our definition, the PBVC also contains protocols for members that can communicate with each other.
Figure 2 shows the join/leave protocol. Before querying for services in a particular PBVC, the user-agent must join that community. The user has to contact the concierge agent to know what is inside the PBVC. The join-protocol is simple and works as follows: when a location-agent detects the user to be in place, it sends the user a notice on the user's display with the PBVC agent's name. The user can communicate with that PBVC to request permission to join the community. If the PBVC is not "over-crowded" or still active at that time, PBVC can let the user-agent join (as in Web sites, some pre-registration and authentication can also be employed if needed). When the user with a mobile device steps out of the physical area belonging to PBVC, the user-agent will automatically leave the community (telling it to release resources, save state if necessary, etc).

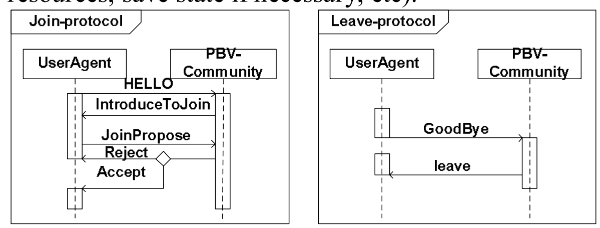

Figure 2. Join and Leave protocol

Figure 3 shows the location protocol for the location agent using GPS data, and another using the sensors data. The Location-Agent can look through the database to find what users are in what communities. This technique works in outdoor environments.

For the indoor system, we use sensors to detect user's location. In this application, we use the iButton[3] as a sensor. The iButton is a computer chip enclosed in a 16mm thick stainless steel can. In each iButton, there is a unique button ID. Depend on what type of button we use, we can have memory on that button so up-to-date information can travel with a person or object anywhere they go. The steel iButton is small and portable enough to attach to a key fob, ring, watch, or other personal items. It can be used daily for applications such as access control to buildings and computers, asset management, and various data logging tasks. When user docks the iButton to the reader, the Sensor-Agent also checks the database to find out user id and current location.

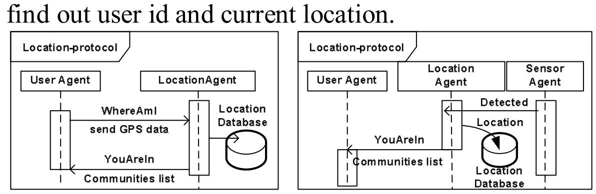

Figure 3. Location Protocol

When the user moves, the UserAgent sends messages to the ContextAgent to set his/her context. The ContextAgent by receiving the message GetMyContext will query the UserContext database. In this example, we predefine some context such as eating, health, studying, exercising, enjoying, working, shopping and resting. Because the context is stored in the database, it can be extended for future use.

4. Implementation And Case Study

To illustrate the idea of the PBVC, we developed an application called "student digital assistant" to provide

[3] iButton, http://www.maxim-ic.com/products/ibutton/

information and services in the campus to assist students. The main goal of the system is to provide suitable services for the student based on his/her context. There are many definitions about context. The process of setting the user's context is difficult. According to Abowd and Dey, context is any information that can be used to describe the character or quality of the entity [1]. In this application, the entity is the user. The user's context is user's location and user's interest profile. The user's interest profile is used to describe what the user is interested in. Users can change this information using tools provided by the application.

We use Google Earth[4] to pre-define an area for each PBVC. Figure 4 shows the predefined area for each PBVC. We use 4 points to define the area to form a rectangle. Each point of the rectangle is determined by GPS coordinates data.

Because the GPS does not work indoors, we defined the virtual area to be larger than the real area. Therefore, when the user is in front of a particular place, the system can detect the user's location. For example, when the user is in location X in Figure 4, the system will know that s/he is inside those PBVCs: Library, Bookshop, East Lecture Theatre (ELT) and Agora. There are many ways to define an area for a PBVC.

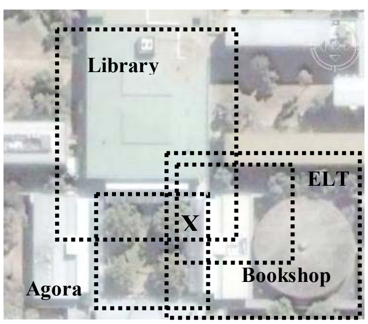

Figure 4. The PBVCs for outdoor places

In our prototype, we choose the rectangle because it is relatively simple to define to start with. In general, the rectangle can be used with most building structures such as houses, rooms and buildings. Furthermore, using a rectangle is convenient when we want to calculate if a user is inside or outside a particular PBVC. Apart from rectangles, with more GPS points, n-sided polygons can be used.

Figure 5. The devices in this application

For the indoor system, we use iButton technology to detect user's location as shown in Figure 5.

4.1. Multiagent system

Agent-oriented approaches are well suited for developing complex, distributed systems [7]. It has many advantages compared to traditional software development. In our prototype, we use JADE, a popular agent development framework.

Figure 6 shows the agents' interactions in the PBVC. In each PBVC, there will be common (or default, or de facto standardized) agents that support users when they join the community [11]. These agents work like the concierges at the hotel. They can help users exploit the services provided within the PBVC.

As shown in the left side of model in Figure 8, agents take care of the user's location detection. There are different types of agents responsible for different types of location detection technology. In this prototype, we use three types of technologies to detect user's location. The GPSLocationAgent is an agent that helps the system detects user's location based on Global Positioning Systems technology. The second type of agent is iButton sensor agent. This agent takes care of iButton, when user docks the iButton to the reader, the reader will check the ID stored in the button and update the database. The third agent detects user's locations by calculating the signal strength from the access point to the user device (which is used in systems such as Ekahau[5] though we did not use it in this prototype).

The right hand side of model in figure 8 shows the agent interactions within the community.

When the user travels in the real world, the UserAgent on user's mobile phone checks for its physical location by connecting to the Bluetooth GPS receiver (the arrow 1). In our implementation, we use the mobile phone (Nokia E61) that have the Location API (JSR-179) to get GPS coordinates data. The E61 is Java enabled and can host JADE/LEAP agent.

[4] Google Earth, http://earth.google.com/

[5] http://www.ekahau.com/

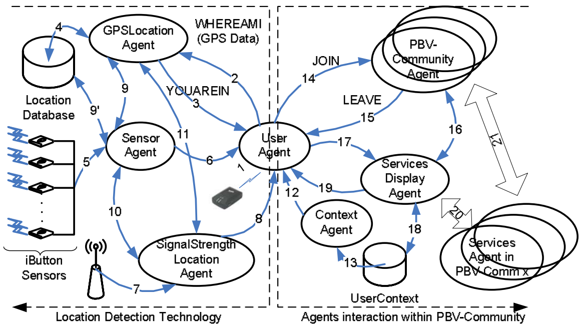

Figure 6. Agents Interaction Model

The UserAgent will request the location agents about its location by sending raw GPS data to the GPSLocationAgent (arrow 2). The GPS Location Protocol will provide the "user-friendly location" information given the raw GPS data (arrow 3). For example, the position agent will display the location to the user that s/he is in front of the LIBRARY and next to the BOOKSHOP instead of, s/he being at coordinates Latitude 37°43'14.27"S, longitude: 145° 2'53.97"E. To present the "user-friendly location" information to the user, the GPSLocationAgent has to query the location database (arrow 4). The location database stores the four coordinates' data of the rectangle for a specific area.

When we test the device in reality, the accuracy of the GPS device affects the results. The Figure 5 shows the BT-74R Bluetooth GPS Receiver was used in the experiment has accuracy below 10m in the open sky (from the product manual). However, the test environment is near buildings, so that affected the GPS accuracy. When the user's device is moved near to the boundary of the area, it can be inside the area one time and outside the area another time even if X did not move. To overcome this situation, we propose blur edges that makes the rectangle ±5% to ±10% bigger. As illustrated by the interaction in arrows 9, 10, 11 in Figure 6, location agents can help each other to provide results that are more accurate by comparing readings. We have to balance the accuracy with the cost of tracking user locations.

The GPS system does not work indoors well so we implemented the iButtons system to detect the user indoors. For example in Figure 5, the user wears a ring that has an iButton attached to it. When the user docks the iButton with the reader, the location agent will see that the reader detects the iButton ID, and then will update and process the information from the memory of the button (arrow 5). According to where iButton is detected the sensor agent will check the predefined location database (arrow 9') to inform UserAgent that s/he is in particular room (arrow 6).

There are many works using mobile phone signal strength to detect user location. Varshavsky *et all* [17] use GSM phone signal strength to detect user location. We mention the signal strength method in this diagram (arrow 7 and 8) but leave this for future work.

Until now, there is no perfect technique for detecting user's location in the environment. Each of the user's location detection technique has advantages and disadvantages about the range, indoor, outdoor and accuracy. As mentioned, to help the system work more effectively, each location agent can correct each other so that it can help the system know the user's location more accurately (resolving ambiguities or giving location information at different levels of granularity). Three arrows (9), (10), (11) show that the location detection agents can talk and share the knowledge about user's location with each other.

Before starting travel, the user agent can set its context profile to let the system assist it more effectively by sending information to the ContextAgent (arrow 12). The ContextAgent can update the user activities profile in the UserContext Database (arrow 13). If the user's context does not exist, the ContextAgent will create a new entry. Otherwise, it will update the context for the UserAgent based on the user input.

Once the UserAgent knows its location, it will contact the PBVC agent to execute the JOIN or LEAVE protocol (arrows 14&15), as appropriate. In reality, a user can be a member of many PBVCs at the same time. By the definition of PBVC, if the user is at the intersection of many PBVCs, s/he can join multiple PBVCs at the same time.

If the joining process is successful, the PBVC agent will send a message to the Service display agent (arrow 16) to tell it about the new user agent that has joined the community.

After the user joined the PBVC, s/he might want to know that what services are provided in the community. As mentioned, the main PBVC agent acts like the concierge at the information desk of a hotel. The PBVC agents can introduce information about the community and tell the UserAgent who to contact for specific services (i.e., referral to other agents representing these services that the UserAgent can communicate directly with).

When the user wants to list what services are provided in the PBVC, the user contacts the ServiceDisplayAgent (arrow 17). We suppose that there will be many services in a PBVC. The number of services available will increase if the user joins many

PBVCs. In order to provide services that are related to the user's interests, the ServiceDisplayAgent has to consult the UserContext database to get user's interests (arrow 18). Then, the ServiceDisplayAgent will match services in particular PBVCs that users are members of with the user's context so that they can be displayed on the user's screen.

The ServiceDisplayAgent and PBVC agents can also communicate with specific service agents to get general information (arrows 20,21). There are many services in each PBVC.

4.2. Demonstration

Figure 7 shows the internal interactions between agents.

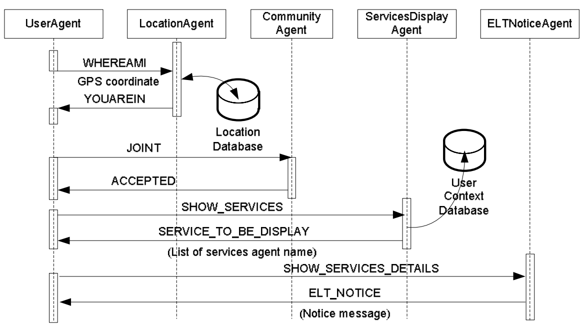

Figure 7. Internal agents' interaction

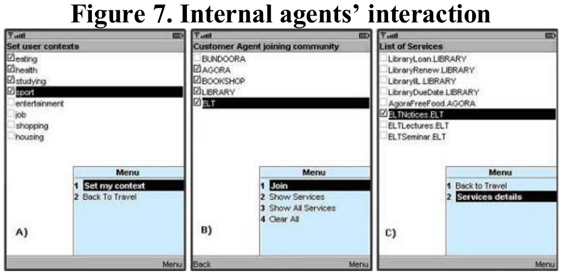

Figure 8. User browses filtered services based on user contexts/interests database

When user are traveling, UserAgent requests its coordinate information from the PositionAgent every 20 seconds using the position protocol. The PositionAgent will query the database and find out the user's current location. Note that apart from polling, the UserAgent can also subscribe to be notified of places (or PBVCs) it finds itself in (i.e., opt-in to receive pushed messages).

Figure 8 shows the screen shots from our prototype. Figure 8A) shows the set context menu for pre-defined context. Setting the user context is a way for the user to define his/her own interest profile (e.g., interested in eating, health and sport) beforehand so that the profile can be issued to ServiceDisplayAgents in PBVCs to obtain services of interest.

Figure 8B shows the communities around user's location and then the user can decide which one to join. Now the user standing in X is within the five communities: Bundoora Suburb, Agora, BookShop, Library, and ELT.

After successfully joining, the UserAgent will list the services in those PBVCs. In reality, there will be many services for the same area. The user must select an appropriate service for him/her-self via the ServiceDisplayProtocol. In Figure 8 C, the user profile at the beginning of demo is set to eating, health, study and sport, so the ServiceDisplayAgent has to match the services with this user's context and location.

For viewing the current state of the PBVC, we also build the viewer application, which shows objects or users within that community.

5. Conclusion and Future Work

In conclusion, we have defined the place-based community and prototyped an application for proof of concept. We have also explained the case for a multiagent approach combined with a PBVC for solving the computational aspects of our concept. In addition, we defined protocols for agent communication and user's interactions in the PBVC. We have discussed the case of the user being in multiple communities at the same time. Our notion of context has been location, time and communities, and we can generalize the work to involve other aspects of context such as agent communication and coordination, ontology and objects in the vicinity. Finally, with the PBVC, we can build similar context-aware applications for multiple areas such as public transport and role-based assistant systems within PBVCs.

The agent interaction model in Figure 6 can be applied for other services such as the bus system. Consider the scenario when the user is waiting for a bus, s/he wants to know the bus timetable. S/He would like to know how many bus stops from his/her current bus stop the next coming bus is currently at. The bus also reports its locations to the control centre. The control centre will provide the user an estimate time for the next coming bus (e.g., Melbourne's Yarra trams have an SMS service to query time-to-arrival of trams). When a user wants to have another choice, for example, using trams or trains, the system will calculate from the user's position to the nearest tram stop or nearest stations. In this application, we can use GPS as location awareness techniques. Such services can be a service within a larger PBVC such as Melbourne (or zones within Melbourne).

In addition, our application can be considered as a location-based services application [8]. However, the services are organized and managed by agents. This approach can help the process of service discovering and utilizing easier.

An ontology can help to clarify the structure of knowledge or lay out assumptions about the content of entities, i.e. about "what is there?" in a certain concept [3]. In our implementation, there are many agents communicating with each other, if we have no ontology, there cannot be common vocabulary for representing information in the protocols used. To make the agents in the system understand each other, we should design ontology for each PBVC and for messages in the protocols. In this project, we use Protégé[6] to create the ontology for the PBVC.

We plan to develop and refine our ontology of PBVCs, and to implement more such PBVCs at varying scales, and investigate their interplay. In addition, it would be good to investigate standardized deployments of this notion in relation to other earlier work such as Agentcities.[7]

6. Acknowledgement

The author *Tuan Nguyen* is supported by Project 300 of Ho Chi Minh City, Vietnam

7. References

[1] G. D. Abowd and A. K. Dey, "Towards a Better Understanding of Context and Context-Awareness," in the 1st international symposium on Handheld and Ubiquitous Computing, Germany, 1999, p. 304.

[2] N. A. Bradley and M. D. Dunlop, "Toward a Multidisciplinary Model of Context to Support Context-Aware Computing," Human-Computer Interaction, vol. 20, p. 43, 13/12/2005

[3] B. Chandrasekaran, J. R. Josephson, and V. R. Benjamins, "What are ontologies, and why do we need them?," IEEE Intelligent Systems and Their Applications, vol. 14, pp. 20-26, 1999.

[4] H. Chen, T. Finin, and A. Joshi, "An ontology for context-aware pervasive computing environments," The Knowledge Engineering Review vol. 18 pp. 197 - 207 , 2003.

[5] G. Cozzolongo, B. D. Carolis, and S. Pizzutilo, "A Personal Agent Supporting Ubiquitous Interaction," in WOA04, Italy, 2004, p. 8.

[6] W. G. Griswold, P. Shanahan, S. W. Brown, R. Boyer, M. Ratto, R. B. Shapiro, and T. M. Truong, "ActiveCampus: Experiments in Community-Oriented Ubiquitous Computing," Computer IEEE, vol. 37, pp. 73-81, Oct. 2004.

[7] N. R. Jennings, "An agent-based approach for building complex software systems," Communications of the ACM archive, vol. 44, pp. 35 – 41, 2001.

[8] A. Küpper, Location-Based Services: Fundamentals and Operation: Wiley, 2005.

[9] S. W. Loke, Context-Aware Pervasive Systems: Architectures for a New Breed of Applications: AUERBACH, 2007.

[10] S. W. Loke, "Modelling Service-Providing Location-Based E-communities and the Impact of User Mobility," in Distributed Communities on the Web: 4th International Workshop, DCW 2002, Sydney, Australia, , Sydney, Australia, 2002, pp. 155-183.

[11] T. Nguyen, S. W. Loke, and T. Torabi, "The Community Stack: Concept and Prototype," in Proceeding of the 21st International Conference on Advanced Information Networking and Applications Workshops Toronto, Canada, 2007, pp. 52-58

[12] C. E. Porter, "A Typology of Virtual Communities: A Multi-Disciplinary Foundation for Future Research," Journal of Computer-Mediated Communication, vol. 10, 2004.

[13] Rakotonirainy, S. W. Loke, and A. Zaslavsky, "Towards multiagent Support for Open Mobile Virtual Communities." Special Session on Multi-Agent Information Systems, Monte Carlo Resort, Las Vegas, Nevada, USA, June 26 - 29, 2000

[14] H. Rheingold, The Virtual Community: The MIT Press, 2000.

[15] N. M. Sadeh, E. Chan, and L. Van, "MyCampus: An Agent-Based Environment for Context-Aware Mobile Services," in Proceeding of Workshop on Ubiquitous Agents on embedded, wearable and mobile devices (ubiagents 2002), Bologna 2002.

[16] A. Sashima, N. Izumi, and K. Kurumatani, "Location-aware middle agents in pervasive computing," in The 2004 International Conference on Pervasive Computing and Communications, 2004, pp. 820-826

[17] A. Varshavsky, M. Chen, E. d. Lara, J. Froehlich, D. Haehnel, J. Hightower, A. LaMarca, F. Potter, T. Sohn, K. Tang, and I. Smith, "Are GSM Phones THE Solution for Localization?," in Proceedings. 7th IEEE Workshop on Mobile Computing Systems and Applications, 2006. WMCSA '06. , Semiahmoo Resort, Washington, USA, 2006, pp. 20- 28.

[18] M. Weiser, "Some Computer Science Issues in Ubiquitous Computing," Communications of the ACM vol. 36, p. 9, 1993.

[19] M. Weiser, "Ubiquitous Computing," http://sandbox.xerox.com/ubicomp, 1996

[6] Protégé, http://protege.stanford.edu/

[7] AgentCities, http://www.agentcities.org/