Implementation of personalized situation-aware service

Tack-Don Han, Hyung-Min Yoon, Seong-Hun Jeong, and Bum-Seok Kang

***Abstract*—New service concepts of sharing contexts with support for communication on wireless networks are beginning to emerge for mobile devices with diverse embedded sensors. In order to promote services based on sharing contexts in mobile devices, this paper introduces the MoCE (Mobile Context Explorer) architecture. MoCE was designed to support context-based services with context sharing on wireless networks in UTOPIA (Ubiquitous computing TOwn Project: Intelligent context Awareness). The major purpose of this research is to provide a situation-aware environment, known as U-Town, by using sensors and mobile devices currently available to the public.**

The paper next introduces U-Theme Park service, which is one service in the U-Town environment. U-Theme Park uses MoCE to realize a personalized environment from various sensors.

***Keywords*—Context-based service, mobile context management, mobile application framework**

I. INTRODUCTION

Future computing environments will enable context-based services that capture and make efficient use of the data available through the sensors attached to various mobile and embedded devices [1][2]. Recently, mobile devices have begun to incorporate with various sensors, such as camera, GPS, and RFID, and support communication via wireless networks such as hot-spots (Wi-Fi, WiBro) [3][4]. Such mobile devices use various context-based services to share information and computing resources among themselves via discovery [2][5][6][7].

This paper emphasizes on the services that support, so called, data-sharing particular to the situational data extracted by the sensors of mobile devices. The data are known as contexts. Using contexts, new mobile applications have been designed in our ubiquitous computing project named UTOPIA (Ubiquitous computing Town Project: Intelligent context Awareness) [8][9]. As shown in Fig. 1, UTOPIA aims to provide intuitive interaction in ubiquitous computing environments and to make new interface and service concepts for efficient interaction [8]. This project has been performed as a joint research program of Yonsei University with SAIT (Samsung Advanced Institute of Technology). We describe the research into situation-aware services and P2P contexts sharing performed in UTOPIA, 2004.

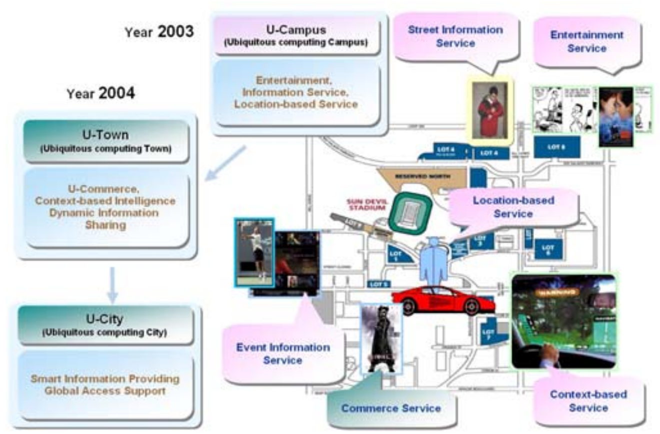

Fig. 1. UTOPIA (Ubiquitous computing Town Project: Intelligent context Awareness) [8]

We particularly classified the characteristics of contexts and defined context flow. In order to support the context flow, we developed MoCE (Mobile Context Explorer) to support P2P context sharing, using a proposed two discovery protocols in a wireless environment. New concepts of mobile services with MoCE are proposed in this paper.

II. CONTEXT FLOW

To support context-based services, mobile devices gather and recognize situational information called contexts to provide more convenient services to the user. Contexts are generated by environmental changes or by various user activities. Context also includes information generated by a computing system, a machine or user's tool [2][3][7]. Schilit and Adams defined context as "the constantly changing execution environment" and they classified context into computing environment, user environment, and physical

This research was supported in part by the Korea Science & Engineering Foundation (KOSEF) under the Basic Research program (No. R01-2005-000-10898-0) and Samsung Advanced Institute of Technology (SAIT) and Korea Ministry of Education & Human Resources Development (MEHRD) under Grant BK21 (Brain Korea 21) Project in 2003-2005.

Tack-Don Han is now with the Department of Computer Science, Yonsei University, Seoul 120-749 Korea. (e-mail: hantack@ kurene.yonsei.ac.kr).

Hyung-Min Yoon is now with Samsung Advanced Institute of Technology, Giheung 449-712 Korea. (e-mail: yoonhm@ chol.com).

Seong-Hun Jeong is now with the Department of Computer Science, Yonsei University, Seoul 120-749 Korea. (e-mail: shjeong@ kurene.yonsei.ac.kr).

Bum-Seok Kang is now with the Department of Computer Science, Yonsei University, Seoul 120-749 Korea. (corresponding author to provide phone: +82-2-2123-2715; fax: +82-31-365-2579; e-mail: led2571@ kurene.yonsei.ac.kr).

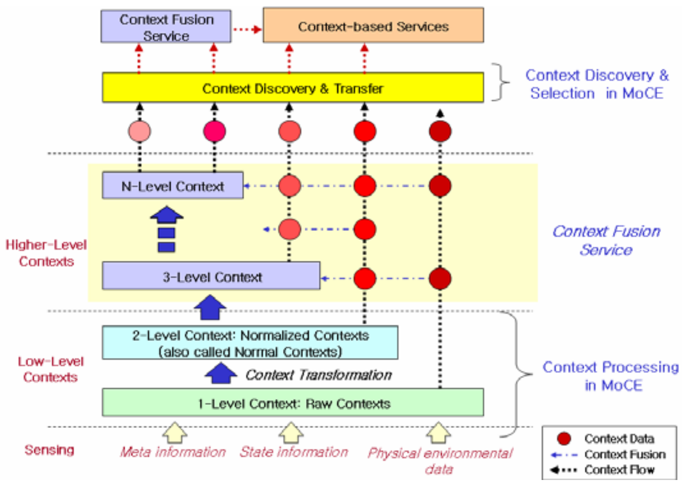

Fig. 2. Context Flow

environment [10]. Dey and Abowd also defined context as "any information that can be used to characterize the situation of an entity [7]."

From a different point of view, we divided contexts into low-level contexts and higher-level contexts, and defined low-level contexts as raw contexts and normal contexts. Raw contexts are the variations in environments comprised of informative data to be extracted from sensors or computing devices. The raw contexts are physical, environmental data from sensors (temperature, scene, etc.), state information of electrical systems, including computing system (network bandwidth, etc.), and meta-information about digital content (user preference data, digital documents, etc.). Normal contexts are defined as normalized data having the common data structure required by services. Examples of low-level contexts are shown in Fig. 2. Higher-level contexts are generated from low-level contexts using context fusion, defined by the requirements of the services providing intelligence, and derived from through aggregation of lower-level contexts via context fusion. Context fusion assembles context information from a combination of related context services and extracts higher-level understanding from lower-level sensory data.

In the case of a driving information service, as shown in Fig. 3, contexts from several sensing devices are transferred to the service according to context flow. The driving information service supports the user's driving with a meeting schedule from PIMS (Personal Information Management System), the camera and GPS of the car's navigation system. Meeting schedule and location, scene in front of the car, and location and speed of the car are acquired as raw contexts. Among these contexts, meeting schedule and bitmap image data of the scene are transferred to normalized contexts. These normalized contexts are aggregated and transformed to higher-level contexts. These higher-level contexts are time of arrival at meeting location and driving information such as road number, speed limit, traffic status, and so on. The driving information service gathers those higher-level contexts and displays them to driver or sends the information to the meeting attendees.

III. TECHNOLOGIES FOR THE U-TOWN ENVIRONMENT

In the U-Town environment, user interactions can be performed through image-based sensors, touch screens, etc. Many components are required to execute the service in the U-Town environment, which operates in mobile devices. These mobile devices can communicate with each other or with the other management systems through a wireless network. Specifically, the service uses a MoCE (Mobile Context Explorer) framework designed for providing and using contexts for intelligent services in a mobile environment.

A. *Image-based sensor (Color code)*

Color code is a two-dimensional (2D) code that represents

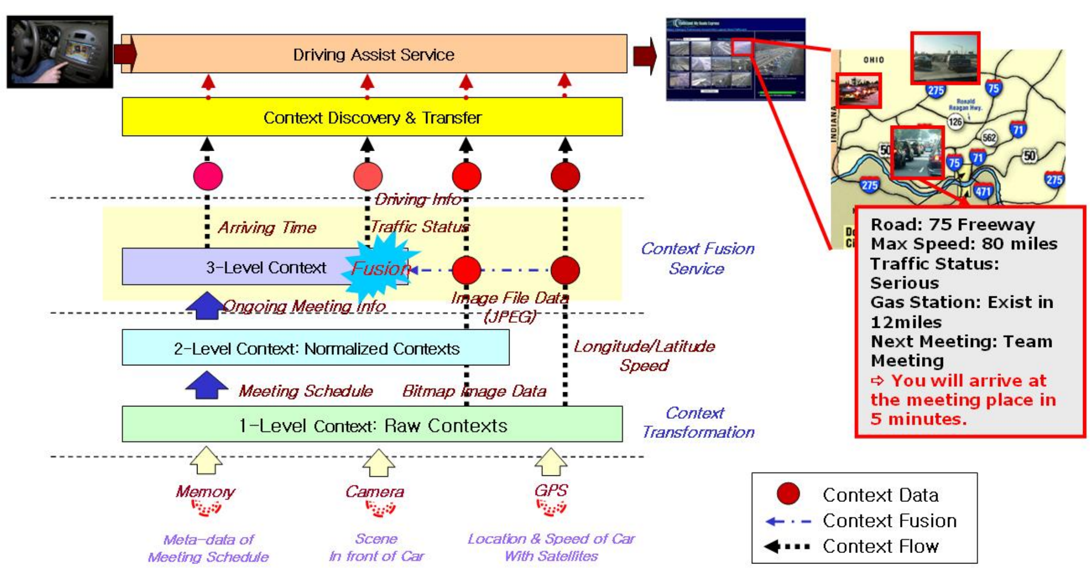

Fig. 3. Context Flow: Driving Information Service

TABLE I
COLOR FORMATION OF COLOR CODE [11]

Code type	5 × 5 Color Code	5 × 8 Color Code	5 × 5 Gray Code	5 × 8 Gray Code
Color	4 colors	4 colors	3 level gray	3 level gray
Unique patterns	17 billion (4×4^{17})	72,057 trillion (4^{28})	43 million (3^{16})	23 trillion (3^{28})

data using either four or more colors, or three or more levels of gray. A color code can be used effectively to link off-line objects in the real world and on-line digital data in cyberspace. TABLE I shows several models for color code according to color formation [11].

Since the color code is printable, it can be attached to various objects, including the print medium itself. Moreover, the information associated with the color code can be accessed anytime and anywhere, via wired and wireless communication channels. It is possible to retrieve useful information from a pre-designated database by capturing and recognizing the color codes present. When a camera in a mobile device detects a color code, a decoding program interprets the color code ID, accesses the database, and provides appropriate information, such as URL, moving picture, or contact address. Fig. 4 shows this model for processing color codes.

A new interface using color codes allows any printed information to be included in the media that can be read automatically by using a camera. Moreover, this interface can be adapted to the Internet easily without directly typing on the keyboard. To summarize, color codes have the following advantages:

1) Color: Until now, the existing codes used only a combination of black and white to present information. This paper details the design and implementation of new codes using color as a major characteristic to compare with other codes or code systems. Since codes using colors are difficult to recognize, research on this approach has yet to be carried out.

2) Cost-efficiency: Color codes are more cost-efficient than electronic tags and barcodes. Color codes, like barcodes, are printable. Although barcodes consists of only black and white, they need high quality printing ink and paper. Electronic tags need expensive memory to store RFID. Color codes, on the other hand, can be used well, regardless of print medium or ink quality.

3) General reader: Color codes can be read using general input devices such as cameras, as opposed to the expensive laser or image scanners required to read barcodes and 2D codes. Likewise, high-precision electronic readers are required to decode RFID tags.

Since color code technology offers cost-efficiency and ease-of-use, information management using color codes would pave the way for various applications linked physical objects and digital data in cyberspace.

B. MoCE Architecture

MoCE was designed to support context-based services with context sharing on wireless networks. In order to share contexts, a mediation architecture is needed to interface and control context requests and offers between devices. With the mediation architecture, context mediator (CM), which supports context consumers was designed. In addition, Embedded Context Provider (ECP), which senses raw contexts from various sensors and transfers them to CM was designed for controlling the access to sensors.

As shown in Fig. 5, the MoCE is composed of a MoCE core layer, a MoCE communication layer, and a MoCE interface layer. The descriptions for the three layers follow:

- *MoCE Interface Layer:* This layer provides interface between CM and services, and manages the list of sensor devices attached to ECP.
- *MoCE Core Layer:* This layer manages the operations of all the components. The core layer of ECP controls context gathering and transformation, and remote-event generation. The core layer of CM controls the connected sensors, remote event processing, and context database.
- *MoCE Communication Layer:* The communication layer discovers context providers (ECP) by using Context Discovery Protocol and transfers the contexts

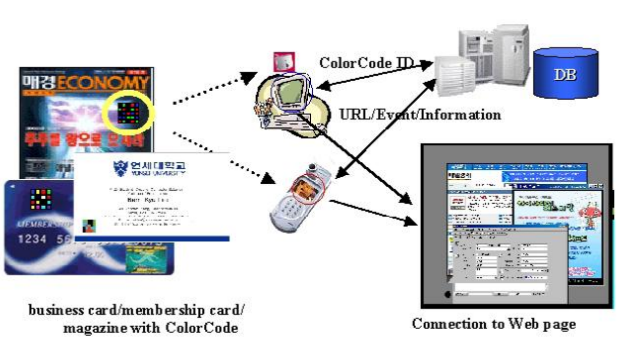

Fig. 4. Color Code System Processing Model

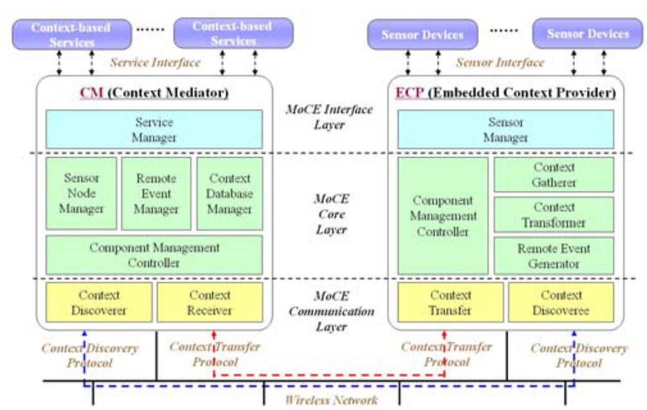

Fig. 5. Layered System Architecture of MoCE

from ECP to CM.

These layers of MoCE were designed to be structurally independent of each other. With the independence between layers, MoCE supports flexibility and extensibility in each component.

1) Context Mediator (CM): The CM, positioned inside the user's device, scans for the ECP, which provides the needed context in the vicinity, and forwards the received context to the service. At this stage, the service communicates with the API(Application Programming Interface)-based service interface designated by MoCE.

2) Embedded Context Provider (ECP): The ECP takes a role in managing the sensors connected to small embedded devices or mobile devices. The ECP extracts the context, sending it to the CM. Sensors acquire context and transmit it to the ECP via the API-based sensor interface, designated by MoCE.

3) Context Discovery Protocol: The context discovery protocols of MoCE were designed for discovering context providers on a hot-spot wireless LAN environment. This protocol does not use directories or registries, such as Lookup Service of JINI™ [6] and DA (Directory Agent) of SLP [12], for discovering context providers. These directories should be maintained with the most recent service's list/proxy resulting from the movements of the mobile devices and the continuous addition/subtraction of context providers. In context discovery protocol, however, context providers receive a discovery request via UDP multicast, and then if a provider may supply the requested context, it sends a discovery reply via UDP unicast. Thus, a user can discover context providers by interpreting variable attributes, such as location, which change dynamically. However, this protocol has some problems with communication among many CMs and ECPs in a hot-spot wireless LAN environment that have low bandwidth and poor stability. When ECPs receive a discovery request from CMs sharing an identical access point in a hot-spot wireless LAN environment, those with matching context attributes will send out discovery reply almost simultaneously. As a result, the access point or hub receiving those reply packets will most likely overflow the transfer buffer; dropouts will then occur, causing the discovery success rate to be lowered. To solve this problem, in this paper, we propose two protocols. Upon accepting a discovery request from the ECP, Fast Discovery Protocol (FDP) waits for a random time and then transmits a discovery reply. This protocol blocks ECP from sending simultaneous discovery replies and minimizes the packet loss rate caused by overflow in the AP's transfer buffer overflow. If the total number of ECPs increases and packet dropouts occur, then the CM should retransmit the discovery request to complete the discovery process. Reliable Discovery Protocol (RDP), our second proposal, is such protocol where the ECP breaks down the discovery reply into equally-sized packets, and then each fragmented packet waits for a random time and transmits it. This protocol reduces the size of simultaneously sent discovery replies and avoids overflow in the AP, or hub, with timing differences. It is possible that the CM detects the existence of all ECPs, as loss of the whole set of packets is unlikely. If, in fact, there was a packet loss, successful completion of the discovery could be accomplished through retransmission via UDP unicast. This method has the advantage of being able to avoid network traffic increase from UDP multicasts. On the contrary, this protocol is incapable of analyzing the data with parts of the lost packet.

4) Context Transfer: Context transfer is required to acquire contexts after appropriate ECPs are discovered. This stage performs the negotiation for connection establishment between CM and ECP with a password-based security authority. The transmission of context is provided by two mechanisms: (1) transmission after the CM's request (pull method); (2) transmission after a specific event in the ECP, such as change of context (push method).

IV. Implementation of U-Theme Park Service

A. Concept and Scenario

The U-Theme Park service prototype is one of the U-Town services in the UTOPIA project. U-Theme Park provides various services based on a user-centric context. A user's

(a) Location Information Service (b) Buddy Finder Service (c) Attraction Information Service (d) Tour Path Recommendation Service

Fig. 6. Screenshots of U-Theme Park service

mobile device gathers context, which it acquires from various sensors or computing devices, using the MoCE framework. Currently, U-Theme Park is composed of four detailed services, as shown in TABLE II. A scenario for each service in U-Theme Park is detailed below.

◆ *Location Information Service*

1. A user's mobile device automatically recognizes a current location using GPS.
2. The current user's location is highlighted on the theme park map in the mobile device.
3. The user can recognize his or her location and attractions in the surrounding area.

◆ *Buddy Finder Service*

1. All members of a group appoint their secure password.
2. If a user hopes to find a buddy's location, the user inputs the buddy's name and secure password.
3. The user can find the buddy's location on the theme park's map.
4. The mobile device periodically traces the buddy's location on the map for the user's needs.

◆ *Attraction Information Service*

1. A user selects an attraction on the theme park map or recognizes a color code attached in front of an attraction.
2. The service provides the user with detailed information about the selected attraction.
3. The user clicks the 'Current Status' button to learn the current wait time or showtime of the attraction, and a live snapshot of the attraction is transferred to the mobile device, which displays it to the user.

◆ *Tour Path Recommendation Service*

1. A user selects his or her tour preference for amusement rides among some menus on the screen, for example, thrill, romance, and youths.
2. The mobile device gathers all of the attraction's wait times and showtimes.
3. The mobile device calculates an optimized tour path by user's preference.

All these services operate on the user's mobile device. Moreover, these services gather and use locations, waiting times, showtimes, tag recognition, and snapshots using the MoCE framework. Fig. 6 shows screenshots of each detailed service on the mobile device.

B. System Architecture

The prototype of U-Theme Park service uses previously stored map information for the theme park, GPS and camera sensor in the user's mobile device. The service also has a form where a user's mobile device is in communication with the server system of the theme park and other users' mobile devices via 802.11g wireless network.

TABLE II
U-THEME PARK: SUMMARY OF DETAILED SERVICES

Service Name	Description
Location Information Service	tracing current location of the user using GPS
Buddy Finder Service	tracing location of family members or friends
Attraction Information Service	retrieving information of various rides and events
Tour Path Recommendation Service	recommending a optimized tour path according to user's preference

U-Theme Park is implemented using the MoCE framework and gathers context using CM. CM not only provides context in user devices, but also provides context in other computing devices to a service. Fig. 7 shows the currently developed system architecture.

As shown in Fig. 7, location from GPS and tag information that has been recognized by camera sensor in the user's device, are provided to the service via ECP. Location, one of the contexts generated by the user device, is periodically sent to the service. Tag recognition information is also transmitted to the service according to change of context. The location information service shows the user's location on the map according to gathered location context, and the Attraction Information Service shows amusement ride and event information according to inputted tag IDs.

Buddy Finder Service can receive location context by communicating with ECP which is installed on other user devices via wireless network. First of all, the service locates a friend via discovery protocol. And then if the user wants to track the location of the buddy, the service periodically receives location from the buddy's ECP.

Finally, the Tour Path Recommendation Service gathers current state information, waiting time, and riding time of amusement rides from ECP, which may be installed at control system of all attractions. Based on gathered information and inputted user's preferences, the service selects attractions suitable for the user, and the mobile device shows an optimized path to user, based on moving time, waiting time, and using time of a selected attractions.

V. CONCLUSION AND FUTURE WORK

This paper designs and implements a prototype of U-Theme Park service for user-centric, situation-aware service. U-Theme Park service is a part of U-Town of the UTOPIA project, and uses various sensors and mobile devices in a ubiquitous computing environment. U-Theme Park service uses managing information of a theme park, such as wait time for an amusement ride, and event information, as well as GPS and camera sensors for acquiring location and tag information.

Currently, U-Theme Park consists of a Location Information Service, a Buddy Finder Service, an Attraction Information Service, and a Tour Path Recommendation Service. Each service provides useful information to the user by using the various contexts around the user.

The MoCE framework is used to implement the U-Theme Park service. The MoCE framework supports context-based services with context sharing on wireless network. The framework gathers context, which is generated by various sensors and computing devices, and transfers context to the service to develop situation-aware service.

In the future, we will improve our service to make it more convenient for users by communicating between several computing devices and sensors that can exchange context mutually in a wider range of areas. For example, we can comment on a U-Government plan and we are on schedule to provide this kind of service at Shinchon, Seoul, in 2006; our UTOPIA team will participate in this plan.

ACKNOWLEDGMENT

We would like to thank Samsung Electronics Co. Ltd. and KTF for providing the smart phones (MITS 400) and the mobile phones (KTF cellular phone). We also acknowledge the CDMA network support of SKT (SK Telecom) and KTF. Many thanks to the Office of Information Systems, the Informatization Promotion Committee and the Office of External Affairs & Development at Yonsei University for providing administrative support for the UTOPIA project.

REFERENCES

[1] P. Brézillon, “Focusing on context in human-centered computing,” *IEEE Intell. Syst.*, Vol. 18, pp. 62-66, 2003.

[2] A. Daftari, N. Mehta, S. Bakre, and X. H.Sun, "On design framework of context aware embedded systems," in *Proc. Monterey Workshop*, Chicago, USA, Sept. 2003.

[3] J. Mantyjarvi, J. Himberg, and P. Huuskonen, "Collaborative Context Recognition for Handheld Devices," in *Proc. 1st IEEE International Conference on Pervasive Computing and Communications (PerCom'03)*, Fort Worth, USA, 2003, pp. 161-168.

[4] S. Benford, C. Magerkurth, and P. Ljungstrand, "Bridging the PHYSICAL AND DIGITAL in Pervasive Gaming," *COMMUNICATIONS OF THE ACM*, vol. 48, no. 3, pp. 54-57, Mar. 2005.

[5] T. Hofer, W. Schwinger, M. Pichler, G. Leonhartsberger, J. Altmann, and W. Retschitzegger, "Context-awareness on mobile devices - the hydrogen approach," in *Proc. International Hawaiian Conference on System Science (HICSS'03)*, Hawaii, USA, Jan. 2003, pp. 292-301.

[6] R. Gupta, S. Talwar, and D. P. Agrawal, “Jini home networking: A step toward pervasive computing,” *IEEE Computer*, Vol. 35, pp.34-40, 2002.

[7] A. K. Dey, D. Salber, and G. D. Abowd, "A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications," *Human-Computer Interaction (HCI) Journal*, vol. 16, pp. 97-166, 2001.

[8] T. D. Han and et al., "Implementation of New Services to Support Ubiquitous Computing for Campus Life," in *Proc. 2nd IEEE Workshop on Software Technologies for Embedded and Ubiquitous Computing Systems (WSTFEUS 2004)*, Vienna, Austria, May 2004, pp. 8-13.

[9] T. D. Han and et al., "Implementation of New Services to Support Ubiquitous Computing for Town Life," in *Proc. Workshop on Software Technologies for Future Embedded & Ubiquitous Systems (SEUS 2005)*, Seattle, USA, May 2005, pp. 45-49.

[10] B. Schilit, N. Adams, and R. Want, "Context-aware computing applications," in *Proc. IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, USA, 1994, pp. 85-90.

[11] T. D. Han and et. al., “Machine readable code and method and device of encoding and decoding the same,” Japan Patent 3336311, Aug. 2002.

[12] E. Guttman, "Service Location Protocol: Automatic Discovery of IP Network Services," *IEEE Internet Computing*, vol. 3, pp. 71-80, 1999.

Tack-Don Han received the B.S. degree in electronic engineering from Yonsei University, Seoul, Korea, in 1978, and the M.S. degree in computer engineering from Wayne State University in 1983. In 1987, he received the Ph.D. degree in computer engineering from the University of Massachusetts at Amherst.

He is a professor in the Department of Computer Science at the Yonsei University, Korea. His research interests include high performance computer architecture, media system architecture, and HCI(Human Computer Interface). Contact him at hantack@kurene.yonsei.ac.kr