Ubi-UCAM: A Unified Context-Aware Application Model*

Seiie Jang and Woontack Woo

KJIST U-VR Lab.
Gwangju 500-712, S. Korea
{jangsei,wwoo}@kjist.ac.kr

Abstract. Context-aware application plays an important role in the ubiquitous computing (ubiComp) environment by providing the user with comprehensive services even without any explicitly triggered command. In this paper, we propose a unified context-aware application model which is an essential part to develop various applications in the ubiquitous computing environment. The proposed model affirms the independence between sensor and application by using a unified context in the form of Who (user identity), What (object identity), Where (location), When (time), Why (user intention/emotion) and How (user gesture), called 5W1H. It also ensures that the application exploits a relatively accurate context to trigger personalized services. To show usefulness of the proposed model, we apply it to the sensors and applications in the ubiHome, a test bed for ubiComp-enabled home applications. According to the experimental results, without loss of generality, we believe it can be extended to various context-aware applications in daily life.

1 Introduction

Ubiquitous computing (ubiComp) allows users to get comprehensive services with ubiquitous computing resources in daily life [1][2]. The sensors and applications in ubiComp-enabled environment will be more intelligent with the development of related technologies, such as embedded networking, pervasive sensing, and intelligent processing. Such a smart environment potentially provides the personalized intelligent services without any explicit user's commands in the near future. In order to achieve such intelligent services, the environment needs to obtain user-centered context information without distracting the users.

Over the last few years, various research activities on context-aware applications have been reported. For example, ACE (Adaptive Control of Home Environment) is a system to control temperature and lighting conditions at home by training the daily life patterns of the residents using Neural Net [3]. Both EasyLiving [4] and AwareHome [5] have showed how context information can be used in the home environment. Meanwhile, MIM (Multimedia Interface Manager) showed how to recognize the user's context through various modalities (i.e. seeing, hearing, touching) through camera, microphone, and haptic glove [6]. Note, however, that contexts used in those applications have different meanings and formats according to the chosen applications.

---

* This work was supported by University Research Program by MIC in Korea.

P. Blackburn et al. (Eds.): CONTEXT 2003, LNAI 2680, pp. 178–189, 2003.

In this paper, we propose a unified context-aware application model that can be used in the ubiComp-enabled home environment [7]. The proposed model consists of two main blocks, i.e. ubiSensor and ubiService. The ubiSensor creates a preliminary context in the form of Who (user identity), What (objects identity), Where (location), When (time), Why (user intention/emotion), and How (user gesture), called 5W1H, by monitoring the user in the environment. The ubiService determines an integrated context by merging preliminary contexts from various ubiSensors and generates the final context that triggers a user-centered service.

The proposed model has various advantages over conventional context models. For example, like a Context Toolkit [8][9], it does not use any mediation for context. However, it maintains independence between sensors and applications by separating the role of Context Toolkit into ubiSensor and ubiService. Then, the ubiSensor generates a preliminary context instead of directly passing the sensed raw data to the Context Toolkit. The resulting context can be shared by all ubiServices and, thus, by all applications. As a result, the context reusability also can be guaranteed. The ubiService also ensures that the application exploits a relatively accurate context to trigger personalized services by feeding back the integrated context to ubiSensors.

This paper is organized as follows: In Section 2, we explain basic terminologies used in this paper. In Section 3, we describe the proposed unified context-aware application model in the ubiComp-enabled home environment. The implementation and experimental results are explained in Section 4 and 5, respectively. Finally, the conclusion and future works are discussed in Section 6.

2 Context for the UbiComp-Enabled Home

Smart Home plays an important role as an application in UbiComp-enabled services. However, the present state of Smart Home focuses on home automation to control doors, lights, elevators, etc. automatically by device-controlling technology such as LONWORKS [10][11], or home networking to connect various information appliances together. UbiComp-enabled Home shall support not only home automation and home networking but also personalized services based on context. To implement the UbiComp-enable Home, we have to overcome the restrictions from which existing context-aware application model suffers, especially dependence between sensor and application, and chaos of context definitions.

Nowadays a sensor of UbiComp-enabled Home depends on its own services. Because of the dependence, developers of context-aware application suffer from adding/replacing/deleting a sensor(s) and from modifying many source codes. Also it is hard to reuse a sensor(s) in other applications.

This dependence can be reduced by using smart sensor in UbiComp which has capability in sensing, processing, and networking. The sensor is indirectly connected to application through the networking and generates unified information for several applications through the processing. It is easy for a sensor to be added, deleted, or replaced by another and reused by other applications. This paper shows that smart sensor converts signals into high level context and transmits this context to application. Specifically, it changes sensed signals to context in forms of 5W1H by its own processing and transmits the context to various applications through its own networking. Therefore, it may guarantee the independence of sensor as well as the reusability for application.

UbiComp-enabled Home requires well-defined context. However, in most of the applications reported, the context is not well defined. Previous context-aware applications mainly use ad-hoc definitions according to the selected applications. For example, Schilt et al. defines context as information about the user and object such as identity and location [12]. Dey et al. defines context as sensed information by the application such as identity, location, activity and state of people, groups and objects [13]. Note however that those definitions may be inconsistent, i.e. changing depending on the selected applications, since such definitions are only suitable for the specific applications.

To solve the problem, we define 5W1H as a unified context so that it can be applicable to all applications in ubiComp environment [1][2]. In general, many context-aware applications retrieve information or trigger a service according to a part of 5W1H such as user identity, location, and time. One theory suggests a unified context, in the form of 5W1H, provides information enough to be used by several applications. Therefore, the unified context model exploiting 5W1H may work in most context-aware applications without loss of generality.

It is necessary that applications of UbiComp-enable Home analyze context to support the user-centered service. To get precise context, we define different levels of context, i.e. preliminary, integrated and final context. The preliminary context generated from a sensor is not enough to trigger a proper service. In general, the extracted context from a sensor may not be accurate or even incomplete since a sensor may not generate all 5W1H. Thus, we introduce integrated context and, thus, final context. The integrated context is completely filled with 5W1H by merging preliminary contexts from a set of sensors. The final context is refined to trigger a user-centered service, which is a set of parameters to be used by a service function. As a result, an application developer may easily design context-aware applications by specifying the condition of the service trigger as a 5W1H.

3 Ubi-UCAM: Unified Context-Aware Application Model

The proposed ubi-UCAM, a unified context-aware application model in ubiquitous computing environment, consists of ubiSensor and ubiService, as shown in Fig. 1. The ubiSensors generate a set of preliminary context. Then the ubiSensors and the ubiServices exchange context through embedded networking modules. The ubiService yields the integrated context by merging the preliminary contexts from a set of ubiSensors and generates the final context by refining the integrated context with the current state of ubiService. Besides, ubiService multicasts the integrated context to ubiSensors, currently connected to ubiService, to help ubiSensor update the preliminary context. The final context is used to trigger the user-centered service.

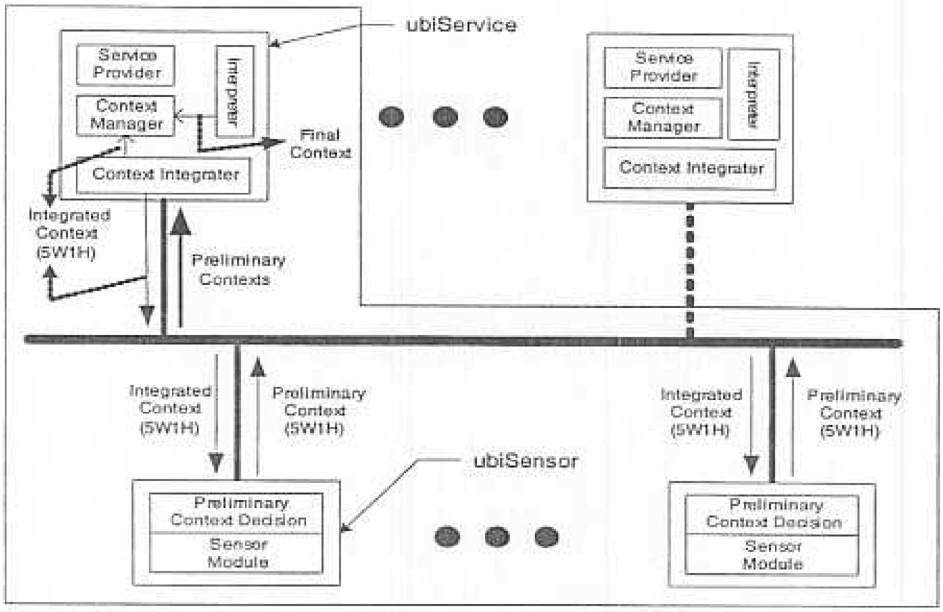

Fig. 1. Ubi-UCAM Architecture

3.1 UbiSensor

The ubiSensor, as shown in Fig. 1, consists of both sensor and preliminary context decision modules. The sensor module monitors the activities of the user in the environment. Then, the context decision module creates the preliminary context in the form of 5W1H by analyzing the sensed signals. As shown in Table 1 the preliminary context is decided using the predefined 'context library' for a specific application. Note that both 'how' and 'why' components among 5W1H, corresponding to the gesture/action and intention/emotion of user, may require more complicated processing. However, to make the problem simple, all 5W1H is determined by the predetermined context library. Accordingly, the ubiSensor referring the same context library generates the same preliminary context.

The resulting preliminary context can be represented in the message format, as shown in Fig. 2. It is more flexible to express preliminary context by using tab character to separate each element of 5W1H. The '-' character also presents empty element, which results from the fact that a sensor module cannot determine the whole 5W1H at a time.

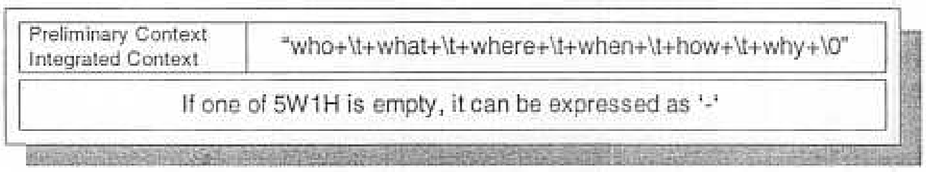

Fig. 2. Context Message Format

Table 1. Context Library

Preliminay Context	Definition
Who	User identity (User Name)
What	Real or virtual object related to user intention
Where	Location of the user or the object
When	Time (YearDateHourMinute)
How	User gestures or action
Why	User intention/emotion to control some services

3.2 Ubiservice

The ubiService, as shown in Fig. 1, consists of four main modules; context integrator (CI), context manager (CM), interpreter (INT), and service provider (SP). CI collects preliminary contexts for a given time (ΔT) from a set of ubiSensors connected to the ubiService, and decides the integrated context. As shown in Fig. 3, the preliminary contexts are aligned and elements of 5W1H in the same column are merged into the integrated context by voting. In case of 'why', we use simple linear mapping, which can be improved by adapting Neural Net. The resulting integrated context has a complete user-centered 5W1H and is forwarded to CM. Simultaneously, the integrated context is multicasted to all ubiSensors.

CM compares the integrated context with all context conditions in a hash table to trigger SP, as shown in Fig. 4. If a context condition is matched, CM calls a function of SP that is associated with the context condition. Otherwise, CM discards the integrated context. The hash table manages context conditions as a key and information of function as a value. The table supports both 1:1 and N:1 relations between a key and a value and also guarantees fast search of integrated context in context conditions. After delivering the selected information and corresponding function to INT, CM runs the service function with the final context from INT.

INT provides CM with the final context, e.g. function name and parameters to trigger specific SP. The final context is generated by mapping the context condition to the parameters based on the current state of ubiService. SP is a set of functions to be triggered as service of ubiService. Each function is associated with a context condition in the Hash table and requires parameters to work. Fig. 5 shows context flow among CI, CM, INT, and SP.

3.3 Networking

The ubiSensor is connected to a network that provides a lookup service maintaining attributes of ubiSensors such as state of connection with ubiService, a sort of preliminary context, etc. The ubiService requests ubiSensors to the lookup service with the needed attributes, and the lookup service returns information of ubiSensor that can provide a preliminary context satisfying the attributes. After receiving information, ubiService directly connects to ubiSensor based on the information. The connection between ubiSensor and ubiService is implemented with middleware such as JINI [14]. Each ubiSensor notifies its own state of connection to the lookup service whenever a change occurs.

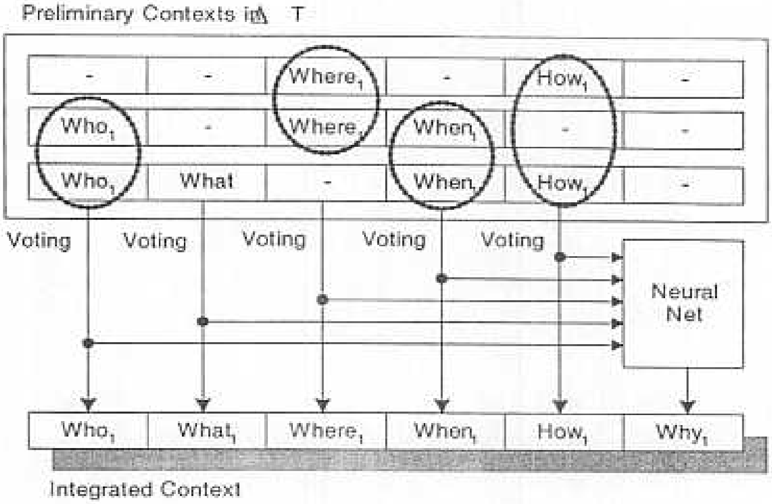

Fig. 3. Integrated Context Processing

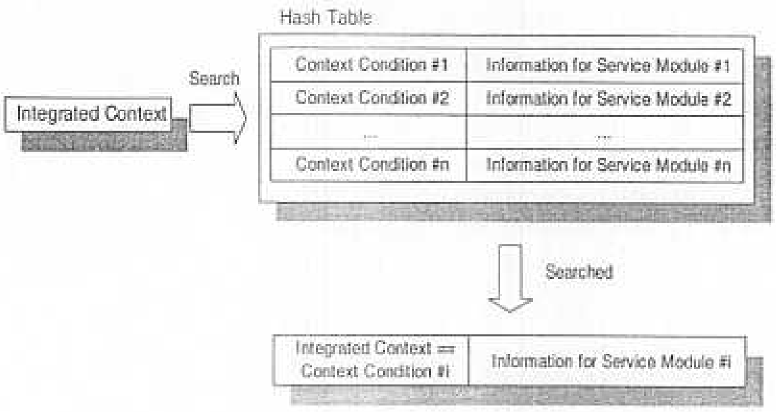

Fig. 4. Searching Context Condition in Hash Table

4 Application: Context-Aware Movie Player

We applied the proposed ubi-UCAM to 'ubiHome', a testbed for ubiComp-enabled home applications. In ubiHome, several ubiSensors (e.g. portable memory, IR sensor, on/off sensor, 3D camera, etc.) provide the preliminary contexts in the form of 5W1H corresponding to user/object identity, location, gesture, time etc. To show the usefulness of the proposed model, we developed a ubiService, which is called c-MP (Context-aware Movie Player).

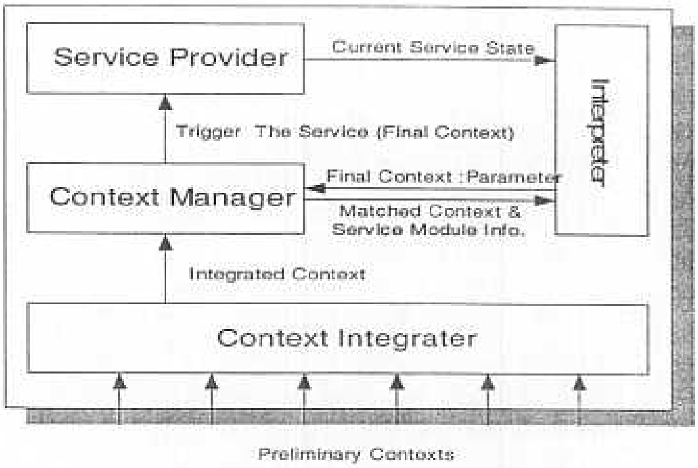

Fig. 5. Context Flow in ubiService Components

The c-MP provides residents in ubiHome with context-aware services. The c-MP provides user-centered services based on the context such as user's identity (Who), user's location (Where), time (When), gesture (How), object for movie player (What) and user's intention to control movie player (Why). For example, after a resident enters a living room with ubiKey, he/she sits down on a sofa in front of the TV. Then, a ubiService menu automatically appears on the monitor. If the resident selects movie player from the menu, the c-MP displays a list of movie titles with user-wise history, as shown in Fig. 6. When the resident rises from his/her sofa, c-MP automatically pauses the movie. If he/she comes back and sits down within 30 seconds from the kitchen for snacks or beverages, c-MP resumes the movie. While, he/she does not come back in 30 seconds or goes out of ubiHome, c-MP saves the paused status and time and automatically stops. The resident can control the movie player by his/her gestures as well as by remote controller [15] . For example, he/she can increase volumes by raising a right hand up and decrease volumes by putting it down. He/she can enlarge screen size by raising a left hand up and lessen screen size by putting it down.

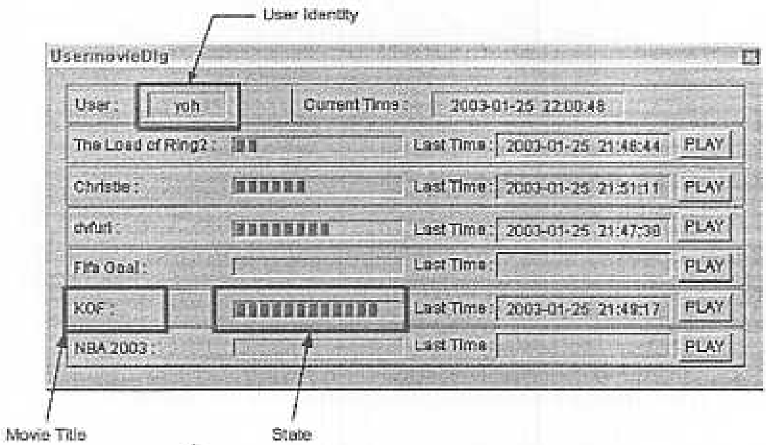

Fig. 6. Example of Context-aware Service

The c-MP gets the preliminary contexts from several ubiSensors such as ubiKey [16], ubiFloor [17], CoachSensor [16] and SpaceSensor[15], as shown in Fig. 7. The ubiKey using portable memory as sensor module generates the identity of the user (Who), location (Where), entering/exiting information (How) and entering/exiting time (When). The ubiFloor, where an on/off sensor is attached per 2cm * 5cm space, yields relative position to TV (How) and time (When). The CoachSensor, where three on/off sensors are embedded in coach, determines the pose of the body, standing up/sitting down (How), and time (When). The SpaceSensor using a 3D camera analyzes hand/body gestures (How) and time (When).

As shown in Table 2, each ubiSensor generates the preliminary context based on context library of ubiHome. For example, when the user, S.Jang (a resident of ubiHome) enters the living room, the ubiKey makes a context message such as "sJang\t-\t LivingRoom\t200301271940\tEnter\t-". When he sits down on a sofa in front of the TV, the CoachSensor generates a context message such as "-\t-\tCoach\t200301271942\t SitDown\t-". If he stands up on the ubiFloor and moves toward the TV, ubiFloor generates a context message such as "sJang\tTV\t-\t200301271944\Comming\t-". If he raises his right hand, the SpaceSensor generates a context message such as "-\t-\t-\t200301271800\t RightHandUp\t-". Finally, all context messages are delivered to the c-MP.

Table 2. Example of Context Library for ubiHome

Preliminay Context	Definition
Who	Name of resident in ubiHome i.g. wWoo, sJang, yOh, sLee, dHong, sKim, yLee, ySuh, sOh, mLee, sjOh, wLee, shLee, kKim, smJung
What	Service Object in ubiHome - real object : Light, TV, MoviePlayer, AV Player, Movie Title - virtual object: Volume, Speed, Size, Luminosity
Where	Location information of ubiHome - LivingRoom, Kitchen, BedRoom
When	Time (YearDateHourMinute) - 200301271900
How	User gestures which are emuerated in a predefined form for ubiHome - Enter, Exit, SitDown, StandUp, Coming, Going, G(Select), G(Play), G(Stop), G(Pause), G(FastFoward), G(VolumeUp), G(VolumeDown), G(SizeUp), G(SizeDown), G(TurnOn), G(TurnOff), G(Bright), G(Dark)
Why	User intenstion and emotion - Intention: to-Play, Select, Stop, Pause, Increase, Decrease, Select, TurnOn, TurnOff, - Emotion: Happy, Angry, Sleepy, Active

The c-MP consists of CI, CM, INT and SP. The CI, as shown in Fig. 8, gathers preliminary contexts every 0.5 seconds. Then it fills an integrated context with 4W1H determined by voting and an empty 'Why'. The remaining element 'Why' can be determined by lookup table or Neural Networks. The CM searches a context condition in the Hash table to find a matched integrated context. If matching occurs, CM triggers a service function of c-MP with the final context. The INT translates the resulting integrated context into the final context in the form of parameters considering the current state of ubiService.

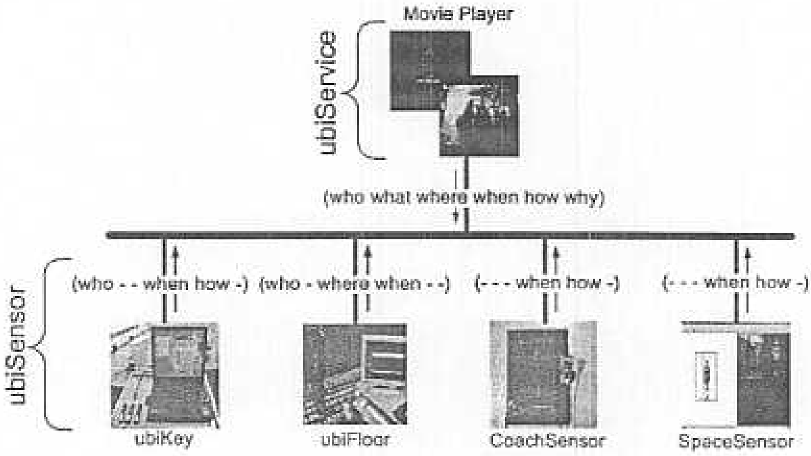

Fig. 7. Example of ubiSensor and ubiService

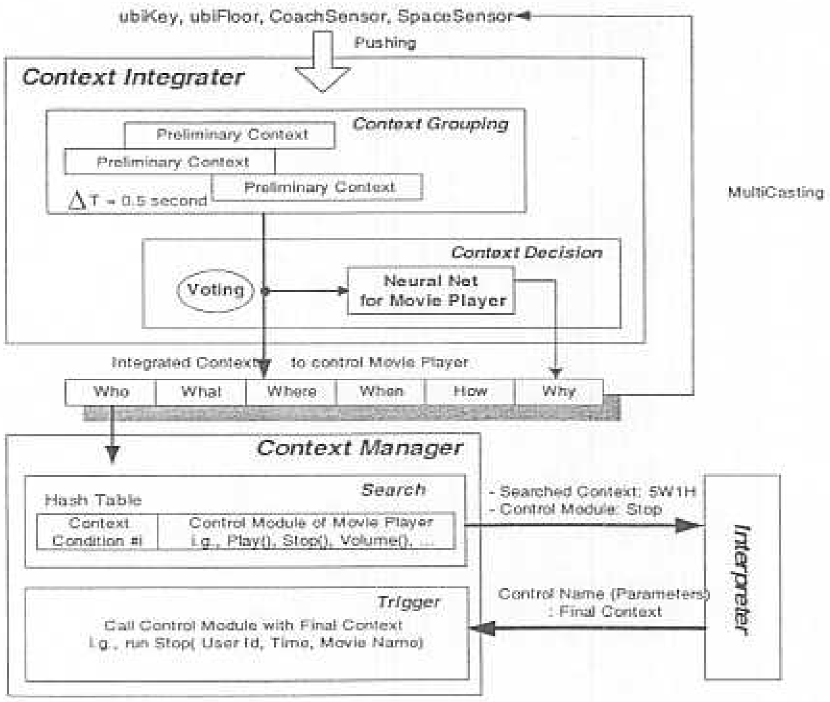

Fig. 8. Context Flow of c-MP

Fig. 9 shows the relationship between the states of c-MP and the integrated contexts. The SP supports control functions such as Play(), Stop(), Select(), Size(), Volume(), Pause(), and FF() according to the context condition.

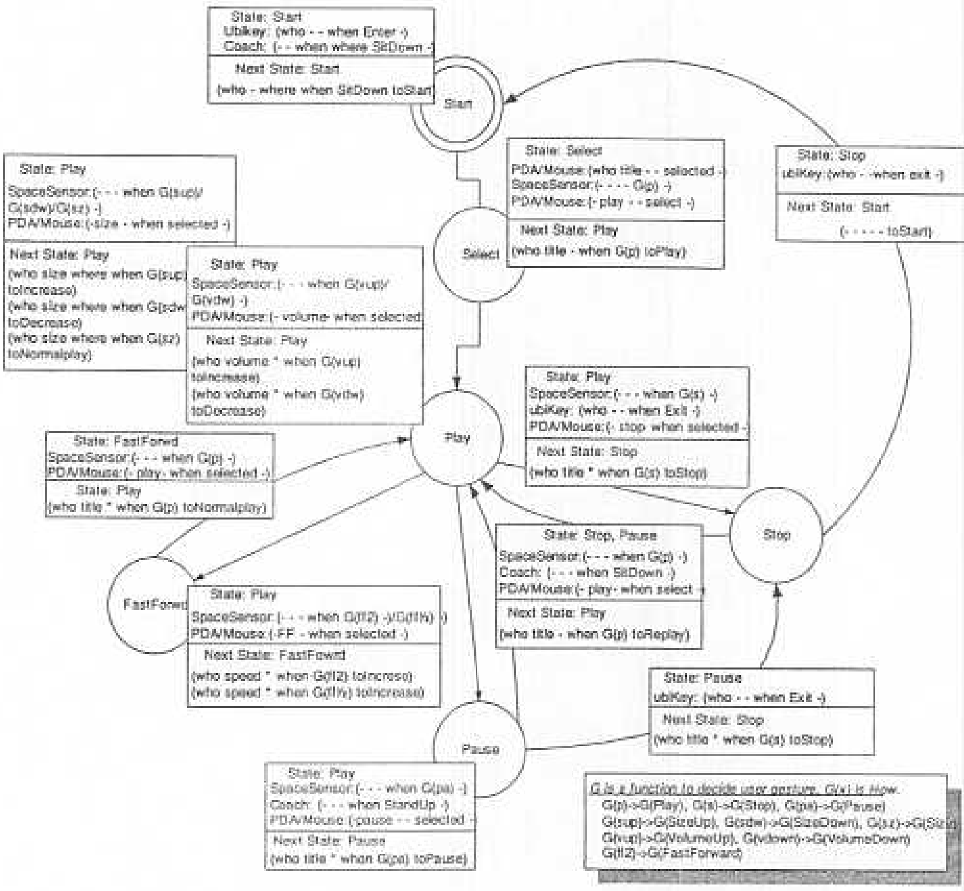

Fig. 9. States & Integrated Contexts of c-MP

5 Experiments

To show the usefulness of the proposed ubi-UCAM, we applied it to a context-aware application, c-MP. And we compared it with a noncontext-aware application, WinAmp, a normal movie player with new skin (java-juke) [18]. Fourteen volunteers (the fifties: 2 persons, the forties: 1 person, the thirties: 3 persons, the twenties: 6 persons, the teens: 2 persons) tested both applications and reported the convenience and satisfaction.

With an assumption that a user was in ubiHome to watch a movie, we measured the time and the number of explicit commands required to start a movie on TV, waiting time per explicit command, and CPU usage of the computer (CPU: PentiumIII

800MHz, Memory: 256MB, OS:WindowsXP). The results are shown in Table 3. As shown in Table 3, when a user watches a movie with Winamp, he/she must click serveral times to selcet a movie. As a result, it requires much attention and long time duration. While with c-MP, he/she needs only two or three explicit commands to select it, since the c-MP automatically provides a user-centered list of movies according to his/her preferences and previous activities. Therefore, c-MP requires relatively less attention and shorter time duration than those of Winamp. The main tradeoffs are waiting time and CPU usage because the c-MP requires processing to get a proper context.

Table 3. Quantatitive Factors

	WinAmp	c-MP
Time duration	20~35 sec	8~12sec
# of Explicit Command	5~12	2~3
Waiting Time per Explicit Command	100~350ms	500~1200ms
CPU Usage	10~15%	30~40%

We have also analyzed the degree of complexity in learning and usage of each player, and the results are summarized in Table 4. As shown in Table 4, most of the participants had no problem in learning how to contol WinAmp with new skin because they were familiar with WinAmp, while they spent some time to learn the instruction of the c-MP, gesture-based commands. Note, however, that after getting familar with the c-MP, they quickly adapted to the new interfaces. Additionally, most of them were satisfied with the personalized movie-playing list that showed the status of the movie (to be watched, to be paused, not to be watched) with time information. Especially, the fifties were positive about controlling movie player by their gesture because they could give attention to a movie without an annoying remote controller. The teens were interested in the auto-play/pause/stop functions because they often movied around ubiHome.

Table 4. Qualitative Factors

	WinAmp	c-MP
Learning Complexity	Easy	Easy
Learning Time	1 minute	2~3 minutes
Usage Complexity	Normal	Easy
Satisfaction	60%	80%

6 Discussion

In this paper, we proposed the ubi-UCAM, a unified context-aware application model in the ubiquitous computing environment and applied it to an ubiComp-enabled home application. The proposed model introduces a unified context in forms of 5W1H that can be shared by various applications and specifies the role of context (i.e., preliminary context, integrated context, final context). According to the experimental results, the proposed model affirms the independence between sensor and application by using a unified context in the form of 5W1H, and exploits relatively accurate context to trigger personalized servieces. However, we must expand on the expression of context because it is difficult to represent the complex context for all application through only context in the form of 5W1H.

References

1. Jang, S., Lee, S., Woo, W.: Research Activities on Smart Environment. IEEK, Magazine. Vol. 28. (2001) 85–97.
2. Yoon, J., Lee, S., Suh, Y., Ryu, J., Woo, W.: Information Integration System for User Recognition and Location Awareness in Smart Environment. KHCI 2002.
3. Mozer, M.: The Neural Network house: An environment that adapts to its inhabitants. In M. Coen (Ed.), Proceedings of the American Association for Artificial Intelligence Spring Symposium on Intelligent Environments. Menlo, Park, CA: AAAI Press. (1998) 110–114
4. Shafer, S., Brumitt, B., Meyers, B.: The EasyLiving Intelligent Environment System. CHI Workshop on Research Directions in Situated Computing (2000)
5. Dey, A., Salber, D., Abowd, G: A Context-aware Infrastructure for Smart Environments. In Proceedings of the 1st International Workshop on Managing Interactions in Smart Environments (MANSE '99). Dublin, Ireland (1999)
6. Marsic, Medl, I. A., Flanagan, J.: Natural communication with Information Systems. Proceedings of the IEEE. Vol.88, No. 8. (2000) 1354–1366
7. Jang, S., Woo, W.: Architecture of Context based Application in Ubiquitous Computing Environment. KHCI (2003) 346–351
8. Dey, A. K., Abowd, G.: The Context Toolkit: Aiding the Development of Context-Aware Applications. Proceedings of the Workshop on Software Engineering for Wearable and Pervasive Computing. Limerick, Ireland (2000)
9. Salber, D., Dey, A. K., Abowd, G. D.: The Context Toolkit: Aiding the Development of Context-Enabled Applications. In the Proceedings of the 1999 Conference on Human Factors in Computing Systems. Pittsburgh, PA. (1999) 434–441
10. LONWORKS, http://www.ktechon.co.kr/spbook/sp_lonworks.html
11. LONWORKS, http://echelon.com
12. Schilit, B., Theimer: M. Disseminating Active Map Information to Mobile Hosts. IEEE Networks. Vol.8. No.5. (1994) 22–32
13. Dey, A. K.: Understanding and Using Context. Personal and Ubiquitous Computing, Special issue on Situated Interaction and Ubiquitous Computing. Vol.5. No.1 (2001)
14. JINI, http://www.sun.com/software/jini
15. Hong, D., Woo, W.: A Vision-based 3D Space Sensor for Controlling ubiHome Environment. In Proceeding of HCI (2003)
16. Oh, Y., Jang, S., Woo, W.: User Authentication and Environment Control using Smart Key. KSPC 2002. Vol. 15. No. 1. (2002) 264
17. Lee, S., Song, H., Ryu, J., Woo, W.: Music Player with the Smart Floor. KSPC 2002. Vol. 15. No. 1. (2002) 158
18. Java-duke, http://www.winamp.com/skins/