社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2005－UBI－8 (52)
2005／6／10

ubi-UCAM 2.0: A Unified Context-aware Application Model for Ubiquitous Computing Environments*

Yoosoo Oh, Choonsung Shin, Seiie Jang and Woontack Woo

GIST U-VR Lab.
Gwangju 500-712, S.Korea
{yoh,cshin,jangsei,wwoo}@gist.ac.kr

ABSTRACT

Context-aware framework can provide multi-service to multi-user without any explicit user's commands by exploiting contexts. Recently, several research activities have been made on a context-aware framework. However, the existing researches do not consider the context-aware framework to support multi-user and multi-service by exploiting the user-centric context. Thus, we propose ubi-UCAM 2.0, a Unified Context-aware Application Model for Ubiquitous Computing Environments, which ensures an independent structure between a set of sensors and services. The proposed model enables that various kinds of sensors and services share the user-centric context in terms of 5W1H represented by the structured context. It makes the user-centric context meaningful by context integration and inference. Furthermore, it resolves user conflictions and service conflictions by the history-based context management technique. Therefore, it can provide intelligent services to users according to user context in smart home environments.

Keywords: Context-aware application model, Context Representation, Context Integration, User/Service confliction

1. INTRODUCTION

Context-aware applications provide services that are appropriate to the particular people without any explicit user's commands by exploiting contexts. They utilize information about surrounding environment with a user obtained from various kinds of sensors. Context-aware framework supports interaction between a user and his environment by exploiting contexts between heterogeneous sensors and various services. Thus, it is required to develop context-aware framework to be provided intelligent services that users want in Ubiquitous Computing environments.

Recently, several research activities have been made on context-aware frameworks. Context Toolkit of GATECH [1], ACHE of Colorado Univ. [2], TEA System of TecO [3], Contextual Service Framework of CMU [4], and ubi-UCAM of GIST [5] are some of them. However, the existing context-aware framework should solve following problems. First, it is hard for a simple text-based context representation to be commonly used for many different kinds of sensors and various services. Second, it is difficult to efficiently integrate the inputted contexts and convert them into the meaningful context. Finally, it cannot present the context management technique to resolve user conflictions and service conflictions. The previous ubi-UCAM 1.0 [5] also has the above problems.

Therefore, we propose ubi-UCAM 2.0, a Unified Context-aware Application Model for Ubiquitous Computing Environments, which ensures an independent structure between a set of sensors and services. The proposed model enables that various kinds of sensors and services share the user-centric context in terms of 5W1H represented by the structured context. It makes the user-centric context meaningful by context integration and inference. Furthermore, it resolves user conflictions and service conflictions by the history-based context management technique.

The proposed model can seamlessly provide user-centric services to multiple users. Also, it ensures a mutual connectivity between a set of different kinds of sensors and various services. It can support intelligent services based on a user's intention or emotion. Therefore, it will take charge of an important role as context-aware framework in Ubiquitous Computing environments.

This paper is organized as follows. We explain the basic structure of ubi-UCAM 2.0 in Chapter 2. We discuss context representation for ubi-UCAM 2.0 in Chapter 3. We describe the context integration/inference in Chapter 4, and the context management for solving the confliction in Chapter 5. Finally, we show the experimental results in Chapter 6 and make a conclusion in Chapter 7.

* This work was supported by DSC of Samsung Electronics Co., Ltd., in S.Korea.

2. ubi-UCAM 2.0

In this paper, we propose ubi-UCAM 2.0 which is a unified context-aware application model for ubiquitous computing environments. The ubi-UCAM 2.0 consists of ubiSensor and ubiService. The ubiSensor consists of physical sensor, feature extraction module, preliminary context generator, and self configuration manager. The ubiService consists of Self Configuration Manager, Context Integrator, Context Manager, Interpreter, and Service Provider. Figure 1 shows the architecture of the proposed ubi-UCAM 2.0.

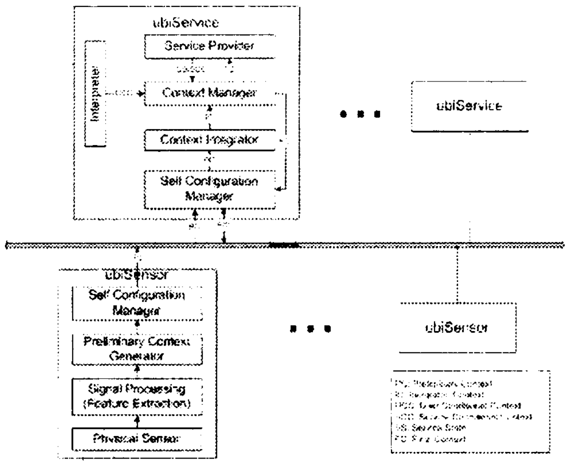


Figure 1: The architecture of ubi-UCAM 2.0.

The unified context expressed with 5W1H ensures independence between sensors and services. It also has advantage of being re-used by other services. In addition, it can reduce additional management to form the context according to individual service. The definition of context type is as follows.

- **Preliminary Context (PC):** Context which expresses the feature information and sensor description generated by ubiSensor.
- **Integrated Context (IC):** Context which forms the complete 5W1H generated in Context Integrator
- **User Conditional Context (UCC):** Context which expresses the service and the condition for its execution that a user defines in the wearable personal station
- **User Conditional Context' (UCC'):** Context converted from UCC as the same form of IC in Interpreter
- **Service Conditional Context (SCC):** Context which expresses the service and the basic condition for its execution that a developer defines in Service Provider
- **Final Context (FC):** Context which resolves user confliction or service confliction in case that IC is matched with UCC' in Context Manager
- **Final Context' (FC'):** Final Context which updates the state of service in Context Manager

The ubiSensor plays a role in forming PC by perceiving a change about a user and his environment. The ubiSensor transfers part or everyone of 5W1H context into ubiService according to a sensor type. Preliminary Context Generator of ubiSensor plays a role in converting feature extracted from a physical sensor into the formatted 5W1H context. The Self Configuration Manager of ubiSensor multicasts PC to ubiServices which are dynamically connected to ubiSensor.

The ubiService plays a role in providing the application service that a user wants to be provided to the user by recognizing context. Self Configuration Manager of ubiService receives contexts through forming a multicasting group dynamically. It supports ad-hoc networking which all ubiSensors and ubiServices can share context in the same active range through making a multicasting group. Context Integrator collects Preliminary Contexts (PCs) in a periodic interval from various kinds of ubiSensors in the same active range with ubiService, and classifies the context as each item of 5W1H particularly. And it applies decision making method into each sub-context fusion according to a characteristic of each item, and generates IC.

Context Manager takes charge of searching the condition of context which corresponds to IC in Hash table, and executes the appropriate service. Hash table has the condition of context about the specific service function. If the condition is matched, Context Manager acquires information about the service execution, and transfers the information to Service Provider. Service Provider manages the implemented code of service module that ubiService provides, and operates service directly after receiving necessary information about service execution. That is, Service Provider calls a necessary service module according to context condition if a developer implements a service module to be provided to a user in Service Provider. Also, Service Provider delivers SCC, which is provided to all users basically to Context Manager if there is no service condition that a user defines from Interpreter. Interpreter provides the environment where a user can designate context condition for service execution. In particular, Interpreter interprets history of change of UCC learned in the wearable personal station with a user into the same format of IC, and delivers it to Context Manager.

3. CONTEXT REPRESENTATION

The context in ubi-UCAM 2.0 is made up of subentries for 5W1H. These subentries are frequently used by several context-based services. The structured context provides a way to accurately represent user's situation because it helps ubiService to maintain various information generated by ubiSensors. In addition, it increases the possibility of matching IC with conditional contexts by context hierarchy.

The structured context consists of user context and system context as shown in Figure 2. User context represents users' situation in terms of 5W1H. The ubiSensor will fill out some or all parts of the subentries in 5W1H depending on its own capability. Especially, when several users access the same ubiService simultaneously, context of only one user is chosen. System context describes the state of ubiSensor or ubiService. State of a ubiSensor will be referred to by ubiServices which make high-level context through fusion of low-level contexts. State of a ubiService determines if that service will be executed when several conflicting ubiServices are triggered simultaneously.

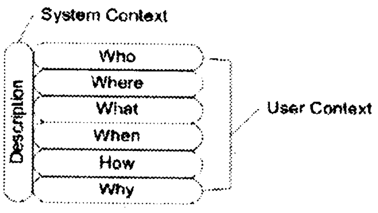


Figure 2: The structured event context

- **Who:** "Who" consists of user's identity, characteristics, and relationship. Identity of users is unique within their group, and is represented by social security number or ID and password to authenticate them as a group member. Characteristics describe user's attributes such as height, weight, birthday and sex. Relationship is priority information to access services in user's group such as home and office.
- **Where:** "Where" comprises of user's indoor or outdoor location. Outdoor coordinates are represented in geographical hierarchy such as country, city and address. Indoor coordinates are described as symbols for naming a floor or room in a building.
- **What:** "What" is information of the ubiSensor or the ubiService being used by a user. It consists of identity and attributes of ubiSensor or ubiService. Identity distinguishes one ubiSensor or one ubiService from others, and is represented in terms of URI (uniform resource identity). An attribute has subentries, such as functions' list of a service and information of a sensor's property.
- **When:** "When" is time information generated when a user accesses ubiServices and moves around. It can be represented by absolute or symbolic time. The absolute time is used to mark the occurrence of an event. The symbolic time describes pattern of user's activity in a day, and is represented in terms of morning, afternoon, evening and night.
- **How:** "How" represents body conditions or gestures of a user. The body condition information consists of electromyogram (EMG), blood volume pressure (BVP), heart rate, galvanic skin response (GSR), respiration, and temperature. The gestures consist of body motion, position of hands and movement of legs.
- **Why:** "Why" consists of user's expression, intention, and emotion. Expression is represented as happiness, sadness, fear, anger, surprise or disgust. Intention is the user's will to access a ubiService with explicit commands. Emotions relate to user's mental conditions but are generally difficult to be described.
- **Description:** "Description" provides extra information about status of ubiSensor or ubiService. Since all contexts have uncertainty, it is necessary to generate accurate context by means of extra information explaining surrounding situations. 5W1H just describes user-centered situations but does not represent how much the described context is precise. To reduce such uncertainty, the status of the ubiSensor is represented by sensing area, confidence and context types. In addition, 5W1H is not enough to explain the surrounding environments, so that it is difficult to represent contradictory relationship among applications. To solve such conflicts, the status of a ubiService is described by working area and name of currently running function. Therefore, description plays an important role in complementing 5W1H about situations of users as well as environments.

4. CONTEXT INTEGRATION

Context Integrator collects inputted contexts at periodic intervals from many different kinds of sensors or application service, and reconstructs a meaningful integrated con-text. Context integration makes IC of a 5W1H form by decision making for human-centric service. Context Integrator is composed of Context Object Analysis module, Preliminary Context Fusion module, Final Context Fusion module, Context Inference Engine, and Integrated Context Generator. Figure 3 shows the architecture of Context Integrator.

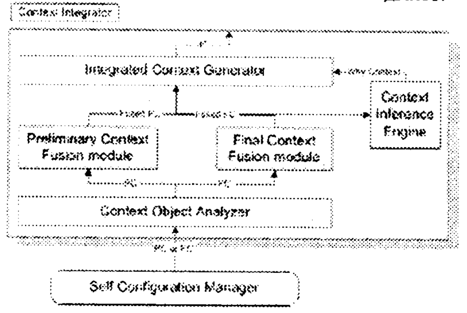

Context Integrator
Integrated Context Generator
Context Inference Engine
Preliminary Context Fusion module
Final Context Fusion module
Context Object Analyzer
Self Configuration Manager

Figure 3: The architecture of Context Integrator.

Context Object Analysis module collects contexts in user-centered view, and classifies the inputted contexts as PCs and

FCs. Preliminary Context Fusion module integrates the inputted PCs as a formatted 4W1H according to characteristics of an each sub-context of 4W1H. Final Context Fusion module simply integrates the inputted FCs according to the user. Context Inference Engine plays a role in inferring "Why" con-text by using the result of Context Fusion module. Context inference extracts user's stress level or intention by observing a change of sub-contexts. Finally, Integrated Context Generator makes IC which can contain information of user's emotion or intention.

4.1 Preliminary Context Fusion module

Preliminary Context Fusion module plays a role in integrating the inputted PCs as a form of 4W1H. Sub-contexts are suitable to express characteristics of a 5W1H context more in detail. The following is the description about the integrated method based on a characteristic of each sub-context.

- **Who Fusion:** The "Who" context has sub-context, such as identity, priority, sex, weight, and height. Identity can be decided by using a voting method which elects a leader among votes. Identity can be decided even though it in PC doesn't contain any information. For example, Context Integrator can build identity information to an unknown user as "User + Number" by comparing the number of person in the environment with the number of the inputted identity. This makes ubi-UCAM 2.0 provide a user-centric service. The remained sub-contexts are updated by the latest information.
- **What Fusion:** The "What" context of ubiSensor consists of sensor ID, sensor type, and accuracy. Sensor ID and sensor type express the unique information that each sensor has, and they can describe a characteristic of PC. For example, if they are filled with the information about location or tracking sensors, we can know that the delivered PC includes position information. Accuracy shows reliability of PC generated in a sensor, and this can be used as the basic information to integrate "How" or "Why" context.
- **Where Fusion:** The "Where" context has sub-context, such as coordinates and symbolic location. The fusion of "Where" context is used to analyze a behavior pattern of a user by using position information expressed with coordinates or a symbol. For example, Context Integrator can know that a user is passing in front of a specific device through a change of location information during the specific interval and information about symbolic location.
- **When Fusion:** The fusion of "When" context imprints time-stamp on every inputted PC. And this fusion obtains the efficient results by flexibly varying time of integration of Context Integrator. Also, this fusion plays a role in imprinting time-stamp in the time when IC is generated.
- **How Fusion:** The fusion of "How" context integrates bio-signal, control information, and so on. In case of bio-signal, this fusion filters only the proper information by using threshold value, such as mean/variance/HF power/LF power of PPG, GSR, and SKT. And this fusion integrates sub-contexts of "How" by selecting dominance among the current input and the previous input.

4.2 Context Inference Engine

Context Inference Engine is a module to generate the "Why" context among elements of 5W1H. Context inference extracts user's stress level or intention by observing a change of sub-contexts. This is to use transition of context [6]. Therefore, it can get a new reasoned context by observing location-change, proximity-change, and function-time change. A change of the "Where" context means a change of a region where a user moves. In addition, it obtains the reasoned information that a user currently walks or runs by calculating a speed. A change of "When"/"What" context means that the device which adjoins a user is continuously changing. This means a change of avail-able devices in a present place which a user exists in present time. Context Inference Engine can infer what the device or service for which currently a user has an interest is. A change of absolute time of the "When" context expresses a change of expected activity time. It infers whether it is time to eat lunch or to work. And, it is based on the user profile information. If this inference extends, it can deduct information of history, schedule, and expectation of users.

5. CONTEXT MANAGEMENTS

Context Manager plays a role in resolving conflict between multiple user and multiple devices in context-aware applications. In addition, Context Manager uses short-term history of users and services to resolve conflicts dynamically. As shown in Figure 4, Context Manager consists of Context Preprocessor, Context Database, Conflict Manager and Final Context Deliverer.

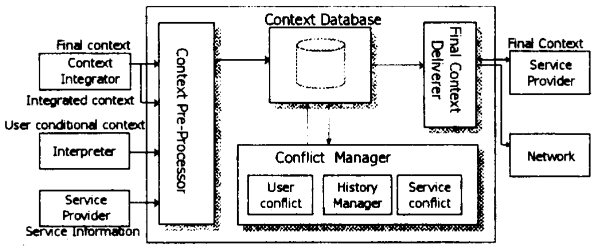


Figure 4: The architecture of Context Manager.

Context Preprocessor performs preprocessing about ICs to be used in Context Manager. To do this processing, the Context Preprocessor deals with UCC', SCC, IC, and FC. Conditional contexts coming from Service Provider and Interpreter are added to Context Database..

Context Database keeps necessary context and related information in order to support generation of final context. It contains several kinds of context such as, conditional context, final and user context within a service area, and conflict history of users and a registered service

Conflict Manager consists of User Conflict Manager, Service Conflict Manager, and Context History Manager. User Conflict Manager manipulates UCC' in two steps: building a user conflict list and selecting a proper user from it. At first, User Conflict Manager makes a conflict list of context on users who are expected to cause conflict, including those who are currently using the service. In the next step, User Conflict Manager selects one user from the conflict list based on their priority obtained from Context History Manager. In the case of more than two users' situation, User Conflict Manager selects one user having highest priority in the same service region.

Service Conflict Manager resolves conflicts caused by multiple services which are trying to share resource in a service area. Service Conflict Manager plays a role in solving conflicts which are caused by other services in the service area. It creates context which contains information about the identity and a stop command of the service, if resources involved in other services are the same as those in this service. As a result, the application responds to changes of other services which cause conflict, using FCs which come from other services. And then, it can play a role in preventing the service causing conflict with other services. To detect possible conflicts, it checks to see if there are any services which use the same resource before delivering the context. Finally, it sends the context to Final Context Deliverer when there aren't any services which are related to the same resource.

Context History Manager keeps track of conflict history among users and services over a short period of time. It manipulates conflict contexts in two steps: context selection and context accumulation. In the context selection, History Manager receives conflict context from Conflict Manager. Then, Context History Manager adds the conflict context to conflict history. Context History Manager calculates the weight of conflict context based on the conflict history by applying Bayesian probability to conflict history of users and services.

6. EXPERIMENTS

The proposed ubi-UCAM 2.0 was tested in smart home test-bed, ubiHome [7]. The ubi-UCAM 2.0 was implemented with J2SDK 1.4 in order to be applied to various service platforms. The experiment was done by comparing ubi-UCAM 1.0 [5] and ubi-UCAM 2.0. For the experiment, we implemented TV application service (ubiTV service) in ubi-Home. The ubiTV service was implemented to interact with various sensors and ser-vice in ubiHome. The ubiTV provides media service such as music, and movie service as well as television service. And, ubiTrack [9] which tracks user's location, and CouchSensor (ubiCouch) [7] which detects user's action, were utilized as ubiSensors together with this service (Figure 5).

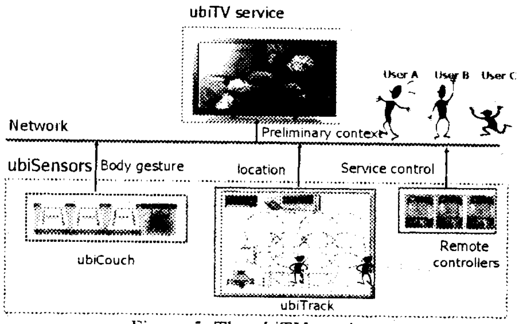


Figure 5: The ubiTV service.

In the experiment 1, we tested how ubi-UCAM 1.0 and ubi-UCAM 2.0 influenced each service. Table 1 shows the comparison results of experiment 1. Service Execution means a procedure that manipulates a channel or sound volume, including the execution of ubiTV service. Multi-service support means to provide the other application services to a user. The other application services are electric lamp service, music service, and movie service (cMP [7]). And Multi-user support is the analysis about whether conflicts between users can be resolved.

Table 1. The comparison between ubi-UCAM 1.0 and ubi-UCAM 2.0.

Framework	Service Execution	Multi-service support	Multi-user support
ubi-UCAM 1.0	Good	Normal	Bad
ubi-UCAM 2.0	Good	Good	Good

As the results, the ubi-UCAM 2.0 can provide multi-service to a user by context integration and inference at the same time. That is because ubi-UCAM 2.0 automatically controls service by using the structured context. Also, ubi-UCAM 2.0 can provide service to a suitable user, by considering personal characteristics and the priority.

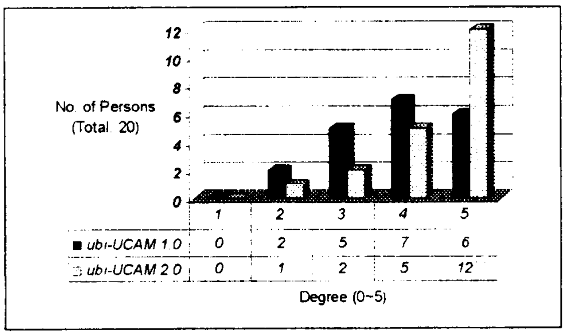
In the experiment 2, we evaluated usability about ubiTV, when ubi-UCAM 2.0 is applied to ubiHome test-bed. Then we estimate usability by averaging data obtained from 20 users' average in Table 2. Average learning time means how long the user who does not use the proposed application service takes the time to perform the application service. Usage efficiency time represents when users who use more than one time reuse the application service. Also, we estimated Average rehabilitation rate about how frequently users make a mistake and restore this, with the following equation (2).

$$\text{Average rehabilitation rate}(\%) = \left(\frac{\text{AverageNo. of rehabilitation}}{\text{AverageNo. of mistake}}\right) \cdot 100 \qquad (2)$$

Table 2. Usability test between ubi-UCAM 1.0 and ubi-UCAM 2.0.

Framework	Average learning time	Average usage efficiency time	Average rehabilitation rate
ubi-UCAM 1.0	18 sec.	11 sec.	65 %
ubi-UCAM 2.0	12 sec.	5 sec.	85 %

As the experiment results, we could know that Average learning time and Average usage efficiency time in ubi-UCAM 2.0 were less, and Average rehabilitation rate in ubi-UCAM 2.0 was more than that in ubi-UCAM 1.0. Specially, Average rehabilitation rate represents ubi-UCAM 2.0 is more efficient. Based on the results of Table 2, we could know that ubi-UCAM 2.0 reflects user's demands more correctly by context integration and inference. Therefore, we were able to know that the ubi-UCAM 2.0 raised usability of service, like Table 2. As the result of the degree of satisfaction of service about 20 volunteers (Figure 6), we could know ubi-UCAM 2.0 provided more comfortable service to a user.



	1	2	3	4	5
■ ubi-UCAM 1.0	0	2	5	7	6
□ ubi-UCAM 2.0	0	1	2	5	12

Figure 6: The degree of satisfaction.

7. CONCLUSIONS

In this paper we proposed ubi-UCAM 2.0, a Unified Context-aware Application Model, which ensures an independent structure between a set of sensors and services. The proposed model can seamlessly provide user-centric services to multiple users. Also, it ensures a mutual connectivity between a set of different kinds of sensors and various services. It can support intelligent service based on a user's intention or emotion. Therefore, it will take charge of an important role as context-aware framework in Ubiquitous Computing environments. In future work, we will develop context integration and conflict resolution method to be adapted by itself in a given environment.

REFERENCES

[1] D. Salber, A.K. Dey and G.D. Abowd, "The Context Toolkit: Aiding the Development of Context-Aware Applications," In the Workshop on Software Engineering for Wearable and Pervasive Computing (Limerick, Ireland), Jun. 2000.

[2] M. C. Mozer, "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., pp. 110-114, 1998.

[3] A. Schmidt, K. A. Aidoo, A. Takaluoma, U. Tuomela, K. Van Laerhoven, and W. Van de Velde, "Advanced interaction in context," In H.W. Gellersen, editor, Proc. of First International Symposium on Handheld and Ubiquitous Computing (HUC99), volume 1707 of LNCS, pages 89-101, Springer-Verlag, 1999.

[4] G. Judd, P. Steenkiste, "Providing Contextual Information to Pervasive Computing Applications" IEEE International Conference on Pervasive Computing (PERCOM), Dallas, March 23-25, 2003.

[5] S.Jang, W.Woo, "ubi-UCAM: A Unified Context-Aware Application Model," Lecture Note Artificial Intelligence, Vol.2680, pp. 178-189, 2003.

[6] H. Wu, "Sensor Data Fusion for Context-Aware Computing Using Dempster-Shafer Theory," PhD thesis, Carnegie Mellon University, Pittsburgh, Pennsylvania 15213, December 2003.

[7] Y.Oh, W.Woo, "A unified Application Service Model for ubiHome by Exploiting Intelligent Context-Awareness," UCS2004.

[8] Jiawei Han, Micheline Kamber, Data Mining: Concepts and Techinques, Morgan Kaufmann., 2001.

[9] S.Jung, W.Woo, "UbiTrack: Infrared-based user Tracking System for indoor environment," ICAT'04, 1, paper 1, pp. 181-184, 2004.