Context-Aware Active Services in Ubiquitous Computing Environments

Aekyung Moon, Hyoungsun Kim, Hyun Kim, and Soowon Lee

With the advent of ubiquitous computing environments, it has become increasingly important for applications to take full advantage of contextual information, such as the user's location, to offer greater services to the user without any explicit requests. In this paper, we propose context-aware active services based on context-aware middleware for URC systems (CAMUS). The CAMUS is a middleware that provides context-aware applications with a development and execution methodology. Accordingly, the applications based on CAMUS respond in a timely fashion to contextual information. This paper presents the system architecture of CAMUS and illustrates the content recommendation and control service agents with the properties, operations, and tasks for context-aware active services. To evaluate CAMUS, we apply the proposed active services to a TV application domain. We implement and experiment with a TV content recommendation service agent, a control service agent, and TV tasks based on CAMUS. The implemented content recommendation service agent divides the user's preferences into common and specific models to apply other recommendations and applications easily, including the TV content recommendations.

Keywords: Ubiquitous computing, context-aware, ubiquitous robot companion, sensor, recommendation.

Manuscript received May 17, 2006; revised Jan. 19, 2007.

Aekyung Moon (phone: + 82 42 860 6735, email: akmoon@etri.re.kr), Hyoungsun Kim (email: kimhs@etri.re.kr), and Hyun Kim (email: hyunkim@etri.re.kr) are with the Intelligent Robot Research Division, ETRI, Daejeon, Korea.

Soowon Lee (email: swlee@mining@ssun.ac.kr) is with the School of Computing, Soongsil University, Seoul, Korea.

I. Introduction

The role of context has recently gained great importance in ubiquitous computing. Ubiquitous computing environments are characterized by many sensors that can detect a variety of different contexts. The context is the information that can be used to characterize the situation of the entities that are considered relevant to the interaction between a user and various applications, including both the user and the applications. This leads to the consideration that applications should take advantage of the contextual information, such as the user's location, to offer greater services to the user; therefore, ubiquitous computing environments must provide middleware support for context-aware applications [1]. Recently, various applications of the user's location-based services have been required such as buddy finding services and position tracking services [2].

Taking this into consideration, we have developed the context-aware middleware for URC systems (CAMUS) [3]. The CAMUS provides a framework for the development and execution of context-aware applications for a ubiquitous robotic companion (URC) [4], such as network-based hardware robots or software robots. To do this, CAMUS gathers contextual information from various sensors and delivers the appropriate contextual information to various applications. Furthermore, CAMUS provides autonomous service agents which are aware of the context, so that they can adapt themselves to different situations. Among other capabilities, the content recommendation service agent provides a method for easily locating interesting items from a large amount of information based on a user preference context. The main reason is that even though the advent of the Internet makes a large volume of content available to users, filtering the relevant items incurs an overhead cost.

In this paper, we propose context-aware active services in CAMUS. The proposed content recommendation service agent and context-aware tasks are developed based on the service framework and task development methodology of CAMUS. To evaluate the proposed system, we apply the proposed active services to the TV domain. The rapid expansion of TV broadcasting channels has made it difficult for users to find the TV programs they desire. Accordingly, TV users are confused about how to find their preferred programs, as several hundred programs are available every day. Over the past few years, a considerable number of studies have been conducted on TV content recommendation systems. These TV content recommendation applications should match the user's desired TV programs and recommend programs with a high user preference [5]-[8]. The proposed content recommendation service agent improves upon previous content recommendation systems. It is more intelligent than previous applications because it provides service without explicit requests and divides the user's preferences into common and specific models. It also has a strong influence on other recommendation applications. In addition, we have implemented a TV control service agent and context-aware TV task based on CAMUS.

The remainder of this paper is organized as follows. Section II introduces related works on context-aware middleware in ubiquitous computing environments. Section III discusses the system architecture of CAMUS, and section IV presents the context-aware service agent and task based on CAMUS. In section V, we implement and experiment with active services in the TV domain. Finally, section VI summarizes this study.

II. Literature Review

Since Mark Weiser [9] proposed the concept of ubiquitous computing, a significant amount of work has been devoted to context-aware computing [10]-[14]. However, this research has remained in the experimental stage of developing prototypes because various contextual information applications and their technologies have only been considered for specific applications in specific platforms. Recently, to rectify this problem, research has been actively progressing to develop flexible middleware which can provide an infrastructure for context-aware services.

The Context Toolkit was developed as an intermediary which deals with the context between a sensor and an application service [15], [16]. It collects contextual information from sensors, analyzes it, and delivers it to the application services. As the name already suggests, the Context Toolkit is not middleware, but a toolkit for implementing various types of context-aware applications in distributed environments.

Ubi-UCAM provides a unified context-aware application model that consists of a ubiSensor and a ubiService, and is applied in a ubiHome [17]. In the ubiHome, several ubiSensors provide the preliminary context corresponding to the user, location, and time, and ubiService provides a context-aware movie player.

Reconfigurable context-sensitive middleware (RCSM) for pervasive computing separates the sensors and application services, and delivers the necessary context information to applications through an ad hoc network [18]. It also collects, analyzes, and interprets contextual information. When this contextual information satisfies the conditions of the registered application services, RCSM delivers the information to the application services. The client-side application decides whether its application services should be executed.

Scarlet proposed a method of exchanging contextual information between heterogeneous platforms [19]. The aim of Scarlet's research was to make it possible to acquire contextual information from sensors in mobile devices through XML SOAP.

Service-oriented context-aware middleware (SOCAM) has also been proposed for context-aware service development. A formal context model was proposed based on ontology using ontology web language (OWL) to address issues regarding the semantic representation, context reasoning, context classification, and dependency [20].

The GAIA system has also been developed as a middleware which can obtain and infer different types of contextual information from various environments [21]. In GAIA, the context is represented by 4-ary predicates written in DAML+OIL (DARPA agent markup language + ontology interface layer). The middleware provides functions which infer high level contextual information from low level sensor information, and predict the context in advance by using historical context information. It also aims to support the development and execution of portable applications for active spaces.

This paper discusses several of the main technical issues regarding context-aware application development for ubiquitous computing environments using CAMUS to support the user preference context and user location context information.

III. Context-Aware Middleware for URC Systems

1. System Architecture of CAMUS

The CAMUS provides a framework for the development and execution of context-aware applications for network-based robots. The URC is a new concept for a network-based service robot: it allows the robot to extend its functions and services by utilizing external sensor networks and remote computing servers. In the URC, CAMUS plays the important role of the software infrastructure that expands the robot's functions and

services, improves the context-awareness in the ubiquitous computing environment, and enhances the robot's intelligence. To do this, CAMUS supports the following functions:

- gathering contextual information from various sensors and delivering appropriate contextual information to various applications and/or tasks,
- inferring higher level contexts from low level contexts,
- supporting a universal data model (UDM) to represent and manage the contextual information,
- providing an event-condition-action (ECA) rule-based program language extending the Java program language,
- tracking the location information of users or other entities to support location-based services,
- providing an event driven application development framework, and
- supporting a service framework enabling the easy integration of various sensors and legacy applications.

Figure 1 shows the system architecture of CAMUS. The CAMUS is composed of four parts: the CAMUS main server (CAMUS-MS), the service agent manager (SAM), the service agent (SA), and a communication framework called *Planet*.

The CAMUS-MS is a framework that collects contexts from the SA and then uses the contextual information. In addition, it supports a variety of functions for context-aware application development. In particular, the CAMUS-MS controls all information about the user context including environmental contexts and user preferences which are required for content recommendations. It then sends events related to the context changes to the applications and helps the applications to perform suitable actions for the context. There is another point that is important for the CAMUS-MS. It offers a service framework which can connect to the basic service agent of a robot controller and a variety of software, such as voice recognition, image recognition, and motion detection. In the next section, we describe the detailed components of the CAMUS-MS.

The SAM is a program which manages and controls SAs within the environment. To do this, the SAM is installed within various environments in a location. It obtains information from a variety of sensors in the environment, sends the information to the CAMUS-MS, receives instructions from the CAMUS-MS, and controls the SAs in the environment. Therefore, the SAM can be installed in any location, such as rooms or offices, or in a robot platform or PDA.

An SA executes the functions of the legacy applications and sensors installed in physical places through communication with the SAM and the CAMUS-MS. An SA is an object proxy for a device or an application which interacts with the

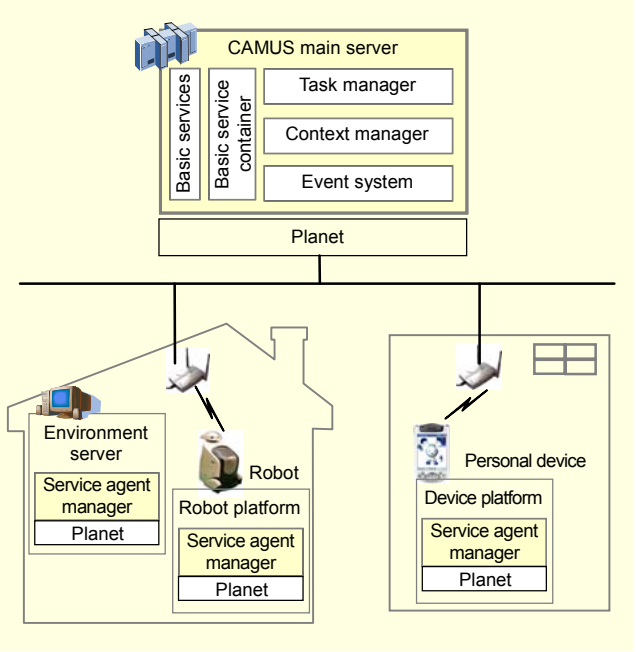


Fig. 1. System architecture of CAMUS.

CAMUS-MS. It has an interface whish is accessed by the CAMUS-MS. For example, suppose that the SA which provides the user's location needs to interface with the RFID sensor in the current environment to send the user's ID and physical location. The SA needs to interface with the device driver that is supported by the RFID sensor's vendor. Accordingly, this implemented program becomes the SA to transfer the user's location.

Planet is the communication framework of a network-based robot system. The CAMUS-MS communicates with the SAM through Planet. A network-based robot system should consider not only the general computing environment, but also the relatively limited device environment in resources such as embedded systems. Therefore, Planet provides a binary message encoding method to minimize the number of messages to be transmitted. It also considers efficient methods of transmitting large amounts of data, such as voice and images. Moreover, Planet handles the long term operation of robots, such as autonomous navigation and speech synthesis. Finally, Planet controls these simultaneous accesses in multi-sessions using locking mechanisms. Planet has been developed to support various programming languages, including Java, C/C++, and C#, run through various operating systems.

2. Conceptual Architecture of CAMUS

Figure 2 shows the conceptual architecture of CAMUS. The sensor framework processes input data from various sources, such as physical sensors, legacy applications, and user

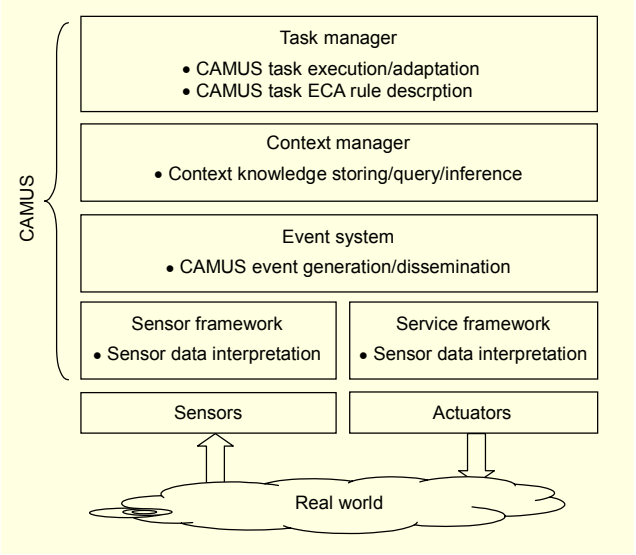


Fig. 2. Conceptual architecture of CAMUS.

commands, while considering the current situation. The sensor framework then transfers the input data to the context manager. For example, voice and image information from users, temperature and humidity information from physical sensors, user schedules from applications, and user preferences may be included as sensor information. In particular, the sensor interpreter included in the sensor framework for voice commands is the *voice recognizer*, which translates voices into textual commands, processes them considering the current situation, and transfers them to the context manager.

The event system generates and manages events from physically distributed environments and is responsible for exchanging messages among the CAMUS components. Above all, it delivers the events to the context manager and task manager so that the CAMUS tasks are able to recognize the situational changes and update the existing context model.

The context manager manages the contextual information gathered from the sensor framework, which is represented by the UDM. When the contextual information in the environment changes, the context manager transfers the events to the event manager. The events are then delivered to the task manager to supply the necessary contextual information required to execute tasks. The event notification system is implemented using the subscribe and publish model.

The task manager initiates individual tasks, manages the on-going task processes, and executes the actual tasks considering the situation. To accomplish this, the task manager has an inference engine to process the facts and rules supplied by a task.

The service framework coordinates the various sensors and legacy applications. It manages the SAM and SA and also controls devices such as the robot platforms and home appliances.

IV. Context-Aware Active Services

To understand the proposed context-aware active services including the content recommendation services, we first illustrate a scenario that is likely to occur in a ubiquitous computing environment. This example scenario represents a TV application domain.

Example 1. Suppose there is a TV in the living room and a TV in the bedroom. The TVs are controlled by voice commands. The users want to execute seamless services in their locations. The applications should take advantage of the contextual information, such as the user's location, to offer greater services.

Scenario.

1) The user enters the living room and sits on the sofa.
2) The TV in the living room is turned on and set to the high preference channel automatically.
3) The user controls the volume and channels with voice commands.
4) The user moves to the bedroom.
5) The TV in the living room is turned off automatically.
6) The TV in the bedroom is turned on if the user leaves the TV in the living room on.
7) The user moves to the child's room.
8) The TV is not available in the child's room.

To complete this scenario, the context-aware active services in CAMUS consist of content recommendation, the actions of control service agents, and the context-aware task. The content recommendation service agent recommends personalized items by comparing the contents and the user preference context. The control service agent is responsible for controlling properties in the physical world, such as a TV. For example, it turns up the volume on the TV according to the user's voice commands and gestures. The context-aware task executes the actual operation of the service agents considering the context.

1. Content Recommendation Service Agent

The system architecture of a content recommendation SA is shown in Fig. 3. This SA consists of a recommendation engine and a service agent interface. The recommendation engine has three components: a content information manager, a recommendation manager, and a user model manager.

First, the content information manager acquires the items for recommendation from the content web server and stores them in the content database. For example, the TV program content database can be gathered from the broadcasting stations' web servers.

Secondly, the user model manager models a user's

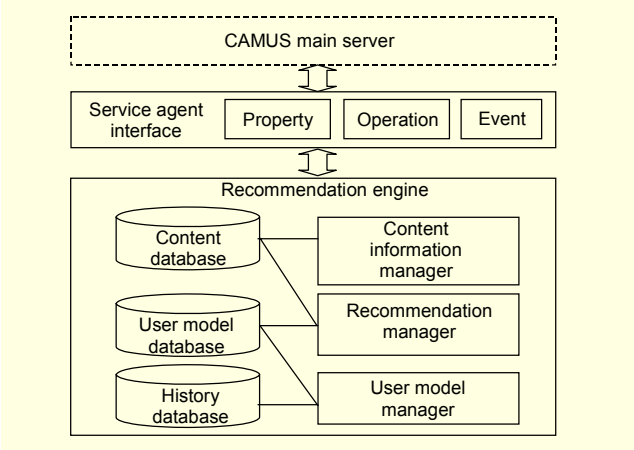


Fig. 3. Components of a recommendation engine and its service agent.

preferences from the information recorded in the user's profile and updates the user model database according to the information gathered from the user's behavior, which is stored in a history database. The user model database consists of two parts: the common and the specific model parts. In particular, the common model influences other recommendation service agents because it contains a pair of keywords and a preference value. This is also updated according to the history database which stores the implicit feedback of the user and adapts to changes in the user's interests. The service agent generates a list of items that the user has selected and stores it in the history database. This user model update can be made by using various aggregative algorithms. The method of updating the preference value of the keywords (V_k^u) is expressed as

$$V_k^u = \frac{F_k^u}{\max|F_k^u|}, \quad (1)$$

where u, k, and F_k^u are the user, the keyword, and the frequency of the keyword as shown in the history table, respectively. The preference value of a user about the keyword k ($Preference_k^u$) is computed as follows; hence, the user model is updated related to the user's preference:

$$Preference_k^u = Preference_k^u + (1 - Preference_k^u) \times V_k^u. \quad (2)$$

Finally, the recommendation manager plays an important role in the content recommendation service agent because it includes a recommendation module. The recommendation manager is implemented according to the content-based approach [22]. This approach matches the user's profile to the content and recommends content with the highest user preference to generate recommendation lists for the user. The method of computing the preference value of each item for each user can be expressed by the two functions of the specific model and the common model given in (1). The common model is computed using the cosine similarity between keyword vectors from the user model and the keyword vectors extracted from the contents in (2). The function of the specific model is related to the application domain; therefore, it can best be described in an implementation example.

$$Preference^{u,i} = \alpha \times Preference_{specific}^{u,i} + \beta \times Preference_{common}^{u,i}, \quad (3)$$

u: user, i: items,
$Preference^{u,i}$: The preference value of user u about item i,
$Preference_{specific}^{u,i}$: The preference of the specific model,
$Preference_{common}^{u,i}$: The preference of the common model,
α: The weight of the specific model,
β: The weight of the common model.

$$Preference_{common}^{u,i} = \text{COSINE\_SIM}(\text{Vector}_{keyword}(c), \text{Vector}_{keyword}(u)). \quad (4)$$

2. Context-Aware TV Task

The task is a set of work items that are taken in a specific context or from the user's command. Each work item, which is a unit of action, is described using the ECA convention. The action part describes the operations to be executed by the SA when the incoming event meets specific conditions. The SA is accessed by a task through the SAM shown in Fig. 1.

Figure 4 shows the state transition diagram of a context-aware TV task designed to implement the scenario represented in example 1. The proposed task uses the two events of UserEntered and SpeechReceived provided by the user to CAMUS. UserEntered is generated by the RFID sensor or image recognition application and is related to the user's location; SpeechReceived is related to the user's voice command. This task consists of four states: Starting, TVNotAvailable, TVON, and TVOFF. As the user moves, the task checks the user's current location, checks the status of the TVs, and executes the related operations of the service agents.

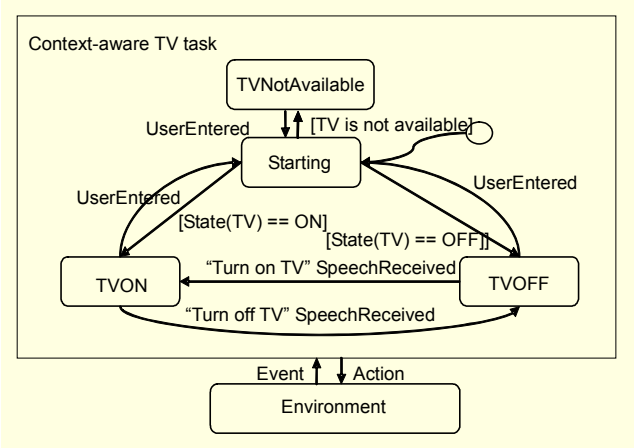


Fig. 4. State transition diagram of a context-aware TV task.

A. Starting State

This is the initial state. It branches to the TVON, TVOFF, and TVNotAvailable states depending on the states of the TVs and the user's current location.

B. TVNotAvailable State

This state indicates that the TV is not available at the user's current location. If the user's location changes, the process returns to the starting state.

C. TVON State

This state means that the TV at the user's current location is turned on. On entering this state, the task executes the recommendation operation of the TV content recommendation service agent, and it changes to a high preference TV channel using the control service agent. Also, the user can control the TV with voice commands. For example, if the user says, "Turn off TV," the TV is turned off and the state transitions to the TVOFF state. If the user location changes, the state transitions to the starting state and turns off the TV. In this state, when the user moves to another location with an available TV, the starting state executes operations of the control service agent and turns on TV in the location.

D. TVOFF State

This state indicates that the existing TV at the user's current location is turned off. If the user's location changes, this state transitions to the starting state. If the user says, "Turn on TV," then it turns on the TV through the control service agent and transitions to the TVON state.

V. Implementation

To evaluate the active services, we applied them to a TV application domain. Therefore, we implemented the TV content recommendation and control service agents and a context-aware TV task was implemented. Finally, we evaluated the example scenario mentioned above.

1. TV Content Recommendation and Control Service Agent

To implement a context-aware service based on CAMUS, the SA defines the property. The property is represented by a name and value pair. A task can access the service agent's property through the service framework and check the state of a variety of sensors and devices in ubiquitous computing environments. Furthermore, by changing the value of a property, the task can control the devices and software applications. In the design of the TV content recommendation service agent, we considered the properties shown in Table 1.

Table 1. Properties of the TV content recommendation service agent.

Property	Definition
tv_recommendation _number	Number of programs that the user wants to be recommended
tv_obtain_interval	Number of days between information updates: the interval between times when the broadcasting information manager acquires the TV broadcasting information from the broadcasting station's web server
tv_program_start_limit	Starting time limitations of a recommended TV program
tv_ program _end_limit	Ending time limitations of a recommended TV program.
tv_channel_code	Local TV channel code

The content of TV programs can be represented by these items: identification information (ID/title), category information (genres/subgenres), broadcast information (channel/starting time/ending time), content ratings, and keywords. The content ratings and keywords are particularly important because the content ratings prevent users from being recommended inappropriate TV programs, (for example, TV programs with violent or sexual content being recommended to teenagers) and the keywords affect the user model of other recommendation service agents. Suppose a user views programs related to the FIFA World Cup regularly. In that case, the preference value of the FIFA World Cup keyword increases. As a result, the recommendation probability of news related to the FIFA World Cup increases in the web news recommendation service agent.

To compute user preferences in the recommendation manager depicted in Fig. 3, the TV specific model is divided into three types of TV content information: genre, channel, and person. To collect this information about the user's interests, the system can ask the user to manually indicate their interests through a graphical user interface (GUI). The function of the TV specific model consists of computing the preference of genre, channel, and person, and multiplying each result by weights, as follows:

$$Preference_{specific}^{u,i} = W_g \times Preference_{genre}^{u,p} + W_c \times Preference_{channel}^{u,p} + W_p \times Preference_{person}^{u,p}, \quad (5)$$

u: user, i: items, p: TV broadcasting program,
$Preference_{specific}^{u,i}$: preference of the specific model of user u about item i,
W_g: genre weight, W_c: channel weight, W_p : person weight.

In addition, we utilized the user's viewing pattern based on

```
initialize U-TBL
while(has more H-TBL)
  program information obtains next history from H-TBL
  d is index of day
  t is index of time
  g is index of genre
  increase U-TBL[d][t][g]
end of while
for i∈day
  for j∈time
    for k∈genre
      F-TBL[i][j][k] += U-TBL[i][j][k]
    end of for
  end of for
end of for
```

Fig. 5. Updating algorithms of the F-TBL.

the time context using the day and the time. To acquire this information, the recommendation manager registers the following frequency table (F-TBL) information. The F-TBL is represented by the three-dimensional matrix that includes day, time, and genre, and registers the frequency of the user viewing specific genres according to days and times into the user model database. The F-TBL is computed by a history table (H-TBL), which is stored in the history database. Figure 5 shows the detailed algorithm used to update the F-TBL. The updating table (U-TBL) stores the temporary information used to compute the F-TBL.

The TV control SA defines the IP and port properties used to communicate with digital TVs or set top boxes (STB). The TV control service agent's operations are the following three types as shown in Table 2.

Table 2. Operations of the TV control service agent.

On/off	boolean getPower() void setPower(boolean power)

Channel	void setChannel(int channel);
	void channelUp();
	void channelDown();

Volume	void volumeUp(); void volumeDown();

2. Context-Aware TV Task

The CAMUS system supports the programming language for ubiquitous environment (PLUE) which allows the programmer to develop context-aware tasks. Essentially, PLUE is an extension of the Java programming language and its compiler is a pre-processor of the Java compiler. The following program code is an example of a TV task for the TVON state as described by PLUE. The ECA rules in PLUE can be augmented with an event expression so that they start only when an expected event is received. Specifically, they are used intensively in the domain where the applications need to react to environmental changes quickly. On entering the TVON state, the task executes a TV content recommendation operation and selects a TV channel with a high preference TV program using the TV control service agent.

In the program code shown in Fig. 6, $place indicates the service agents existing in the user's current location and $owner identifies the specific user. Also, the user can control the TV using voice commands. For example, if the user says, "Turn off TV," the TV is turned off and moves to the TVOFF state. As shown in the example above, the expression "on event SpeechReceived (e)" describes the event that starts this rule and the expression "if (e.symbol == volume up)" describes the condition of the rule. Finally, the remainder of the rule explains the action to be taken. The rule in the example is read as, "Whenever the user says volume up, turn up the volume of the TV." Conventional Java method calls are allowed in the rule expression: volumeUp() is an operation of the TV control service agent. If the location of the user changes, the state transitions to the starting state and turns off the TV.

Figure 7 shows the results of this task. In the case in which

```
State TVON {
  entry {
    TVProgram info =
          $place.tv.service.getRecommendation($owner);
    $place.tv.serviceagent.setChannel(info.channel);
    }
    Exit {$place.tv.service.setPower(false); }
    on event SpeechReceived(e)
    condition(e.symbol == 'volume up')
          $place.tv.volumeUp();

    on event SpeechReceived(e)
    condition(e.symbol == 'volume down')
          $place.tv.volumeDown();
}
Transition StateTVON-> StateTVOFF {
    on event SpeechReceived(e)
    condition(e.symbol == "tv off")
          $place.tv.setPower(false);
}
Transition StateTVON -> Starting {
    on event UserEntered(e)
    condition($owner.lastLocation()!= $owner.currentLocation()) {
          $owner.tv = true;
          $place.tv.setPower(false);
    }
}
```

Fig. 6. Pseudo program code of the TVON state.

the user enters a room and turns on the TV, the selected program ranks first on the recommendation list. These results are presented on a PDA and consist of TV programs broadcast by Korean broadcasting stations.

Figure 8 shows the results of the experiments based on the scenario in example 1. The user's location is recognized by an RFID sensor and a camera sensor. When the user enters the living room, the TV is turned on and automatically set to the channel with the highest preference (Fig. 8(a)). When the user moves to the bedroom, the TV is turned off in the living room and at the same time the TV in the bedroom is automatically turned on (Fig. 8(b) and (c)). However, if the user moves to the child's room, the TV is turned off in the bedroom because no TV is available in the child's room (Fig. 8(d) and (e)).

no.	프로그램명	방송시간
1	2004 아테네 올림픽	22:00-24:00
2	풀하우스	
3	바람난 가족	21:50-23:50
4	베트맨	20:20-22:00
5	풀하우스	19:00-24:00
6	2004 아테네 올림픽	18:00-24:00
7	풀하우스	18:00-23:00
8	바람난 가족	17:50-23:50
9	베트맨	17:20-22:00
10	바람난 가족	16:50-23:50

Fig. 7. Result of the task.

VI. Conclusion

In this paper, we proposed context-aware active services using CAMUS. The proposed content recommendation SA and context-aware task were developed based on the service framework and task development methodology of CAMUS. The proposed content recommendation SA is more intelligent than previous content recommendation applications because it provides services even when there is no explicit request and it divides the user preferences into common and specific models; it also has a strong influence on other recommendation applications.

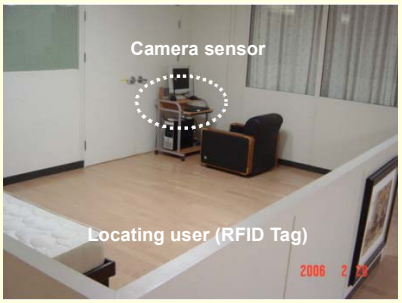
To evaluate our system, we applied the proposed active services to a TV application domain. Also, in the implementation of the TV content recommendation SA, we registered content ratings. This is a strong point of our system because the content ratings prevent users from being recommended inappropriate TV programs. We utilized the user's viewing pattern behavior based on the time context and implemented a TV control SA and context-aware task based on CAMUS. Finally, the implemented context-aware TV task responded in a timely fashion to changing contexts, such as the

(a) The user is in the living room.

(b) The user moved to the bedroom.

(c) The user in the bedroom.

(d) The user moved to the child's room.



(e) The user in the child's room.

Fig. 8. Snapshots of the experiment.

user's location and voice commands.

References

[1] A. Ranganathan and R.H. Campbell, "A Middleware for Context-Aware Agents in Ubiquitous Computing Environments," *LNCS* (2672), 2003, pp. 143-161.

[2] K. Min, K. Nam, and J. Kim, "Multilevel Location Trigger in Distributed Mobile Environments for Location-Based Services," *ETRI Journal*, vol.29, no.1, 2007, pp.107-109.

[3] H. Kim, Y. Cho, and S. Oh, "CAMUS: A Middleware Supporting Context-Aware Services for Network-Based Robots," *IEEE Workshop on Advanced Robotics and Social Impacts*, 2005.

[4] Y. Ha, J. Sohn, Y. Cho and H. Yoon, "Towards Ubiquitous Robotic Companion: Design and Implementation of Ubiquitous Robotic Service Framework," *ETRI Journal*, vol. 27, no. 6, 2005, pp. 666-676.

[5] L. Ardissono, et al., "Personalized Recommendation of TV Programs," *LNCS* (2829), 2003, pp. 474-486.

[6] W. Lee and T. Yang, "Personalizing Information Appliances: A Multi-agent Framework for TV Programm Recommendations," *Expert Systems with Applications*, vol. 25, 2003, pp. 331-341.

[7] Y. Blanco-Fernández, et al., "AVATAR: An Advanced Multi-agent Recommender System of Personalized TV Contents by Semantic Reasoning," *LNCS* (3306), 2004, pp. 415-421.

[8] J. Xu, L. Zhang, H. Lu, and Y. Li, "The Development and Prospect of Personalized TV Program Recommendation Systems," *Proc. IEEE Symp. Multimedia Software Engineering*, 2002, pp. 82-89.

[9] M. Weiser, "The Computer of the 21st Century," *Scientific American*, vol. 265, no. 3, 1991, pp. 66-75.

[10] R. Want, A. Hopper, V. Falcão, and J. Gibbons, "The Active Badge Location System," *ACM Transactions on Information Systems*, vol. 10, no. 1, Jan. 1992, pp. 91-102.

[11] G.M. Voelker and B.N. Bershad, "Mobisaic: An Information System for a Mobile Wireless Computing Environment," *IEEE Workshop on Mobile Computing Systems and Applications*, 1994.

[12] A. Asthana, M. Cravatts, and P. Krzyzanowski, "An Indoor Wireless System for Personalized Shopping Assistance," *IEEE Workshop on Mobile Computing Systems and Applications*, 1994.

[13] R. Want, B.N. Schilit, N.I. Adams, R. Gold, K. Petersen, D. Goldberg, J.R. Ellis, and M. Weiser, *Mobile Computing*, Kluwer Academic Publishers, 1996.

[14] G.D. Abowd, C.G. Atkeson, J. Hong, S. Long, R. Kooper, and M. Pinkerton, "Cyberguide: A Mobile Context-Aware Tour Guide," *Wireless Networks*, vol. 3, no. 5, Oct. 1997, pp. 421-433.

[15] A.K. Dey, M. Futakawa, D. Salber, and G.D. Abowd, "The Conference Assistant: Combining Context-Awareness with Wearable Computing," *Symposium on Wearable Computers*, Oct. 1999.

[16] A.K. Dey and G.D. Abowd, "The Context Toolkit: Aiding the Development of Context-Aware Applications," *Workshop on Software Engineering for Wearable and Pervasive Computing (SEWPC)*, 2000.

[17] S. Jang and W. Woo, "Ubi-UCAM: A Unified Context-Aware Application Model," *Context*, *LNAI* (vol. 2680), 2003, pp. 178-189.

[18] S.S. Yau, F. Karim, Y. Wang, B. Wang, and S. Gupta, "Reconfigurable Context-Sensitive Middleware for Pervasive Computing," *IEEE Pervasive Computing*, vol. 1, no. 3, 2002.

[19] A. Daftari, N. Mehta, S. Bakre, and X.H. Sun, "On the Design Framework of Context Aware Embedded Systems," *Software Engineering for Embedded Systems: From Requirements to Implementation*, 2003.

[20] T. Gu, H.K. Pung, and D.Q. Zhang, "A Middleware for Building Context-Aware Mobile Services," *IEEE Vehicular Technology Conference (VTC)*, 2004.

[21] G. Biegel and V. Cahill, "A Framework for Developing Mobile, Context-Aware Applications," *IEEE Int'l Conf. on Pervasive Computing and Communications* (*PerCom*), 2004.

[22] M. Balabnovic and Y. Shoham, "Content-Based, Collaborative Recommendation," *CACM*, vol. 40, no. 3, 1997, pp. 66-72.

Aekyung Moon received the MS and PhD degrees in computer engineering from Yeungnam University, Korea, in 1992 and 2000, respectively. In 2000, she joined Electronics and Telecommunication Research Institute, Korea. Currently, she is a senior researcher in the Software Robot Research Team. Her research interests include distributed systems, context-aware middleware, sensor networks, and recommendation systems.

Hyoungsun Kim received the MS degree in computer engineering from Kwangwoon University, Seoul, Korea, in 1991, and the PhD degree in computer engineering from Daejeon University, Daejeon, Korea, 2003. He had worked for Systems Engineering Research Institute (SERI) from 1986 to 1998. He joined Electronics and Telecommunications Research Institute (ETRI) in 1998 and has worked on software development related to intelligent robots. Currently, he is senior researcher in the Software Robot Research Team, ETRI. His research interests include context-aware computing, networked robots, software robots, distributed computing, Information security, and distributed database.

Hyun Kim received the BS, MS, and PhD degrees in the Department of Mechanical Design and Manufacturing from Hanyang University, Seoul, Korea, in 1984, 1987, and 1997, respectively. He worked for Systems Engineering Research Institute (SERI) from 1990 to 1998. He joined ETRI in 1998 and has worked on software development related to intelligent systems. Currently, he is a project leader in the Intelligent Robot Research Division, ETRI. His research interests include networked robots, context-awareness and ubiquitous computing, distributed computing, and virtual engineering.

Soowon Lee is an associate professor in the School of Computing at Soongsil University, Korea, and director of its Mining Lab. He received his PhD in computer science from the University of Southern California in 1994. His current research interests include data mining, personalization, machine learning, and agent systems.