IEICE TRANS. INF. & SYST., VOL.E90–D, NO.6 JUNE 2007

LETTER

Enhanced Framework for a Personalized User Interface Based on a Unified Context-Aware Application Model for Virtual Environments*

Youngho LEE[†], ***Student Member***, **Sejin OH**[†], **Youngjung SUH**[†], **Seiie JANG**[†,††], ***and*** **Woontack WOO**[†a)], ***Nonmembers***

SUMMARY In this letter, we propose a enhanced framework for a Personalized User Interface (PUI). This framework allows users to access and customize virtual objects in virtual environments in the sense of sharing user centric context with virtual objects. The proposed framework is enhanced by integrating a unified context-aware application for virtual environments (vr-UCAM 1.5) into virtual objects in the PUI framework. It allows a virtual object to receive context from both real and virtual environments, to decide responses based on context and if-then rules, and to communicate with other objects individually. To demonstrate the effectiveness of the proposed framework, we applied it to a virtual heritage system. Experimental results show that we enhance the accessibility and the customizability of virtual objects through the PUI. The proposed framework is expected to play an important role in VR applications such as education, entertainment, and storytelling.

key words: *user-centric context, personalized user interface, context-awareness, virtual reality*

1. Introduction

Many researchers have studied how users approach to virtual environments from real environments[1]–[6]. Forsberg et al. suggested that we need to facilitate seamless transition between technique and hardware when we design user interaction [1]. He also proposed a framework and showed several applications. However, without considering a user's situation, applications always provide the same interfaces to all users. Thus, Jang and Woo proposed a framework that provides users with a personalized interface by sharing user-centric context [7], [8]. However, it still has a limitation in its ease of access to and customization of virtual objects in NAVER [10]: The rv-interface, a bridge between real and virtual environments as shown in Fig. 1, supports the limited scope of customization of virtual events that influence the actions of an individual virtual object. On the other hand, a user is restricted to interact only with a whole virtual environment not with an individual virtual object.

Manuscript received November 10, 2006.

[†]The authors are with GIST U-VR Lab., Gwangju, 500–712, S. Korea.

[††]The author is with ITC, Nagoya University, Nagoya-shi, 464–8601 Japan.

*This research was supported by the Ubiquitous Computing and Network (UCN) Project, the Ministry of Information and Communication (MIC) 21st Century Frontier R&D Program in Korea.

a) E-mail: wwoo@gist.ac.kr

DOI: 10.1093/ietisy/e90–d.6.994

In this letter, we propose a enhanced framework for a personalized user interface (PUI). To enhance the previously proposed framework, we insert virtual objects, embedding a unified context-aware application model for virtual environments (vr-UCAM 1.5), into NAVER [10] as illustrated in Fig. 2. It allows each virtual object to receive user-centric context through its own communication channel individually by utilizing a network self-configuration manager (SCM). Then, the virtual object selects reactions based on if-then rules and user's context by itself. Additionally, it adopts a configuration scheme to SCM for solving the scalability problem while a user interacts with a large number of virtual objects.

The proposed framework enhances the accessibility and customizability of virtual objects through the PUI. A user can access a virtual object through PUI by applying his/her personal information into an individual virtual object. Moreover, a user can invoke personalized responses of a virtual object by applying user centric context. In addition, a user can experience the personalized responses of a large

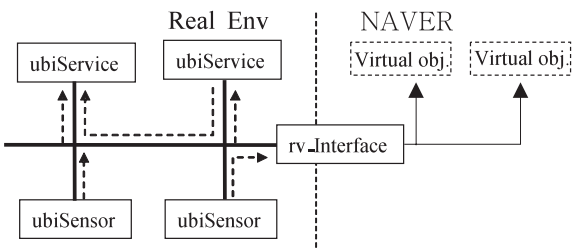


Fig. 1 Framework for a personalized user interface.

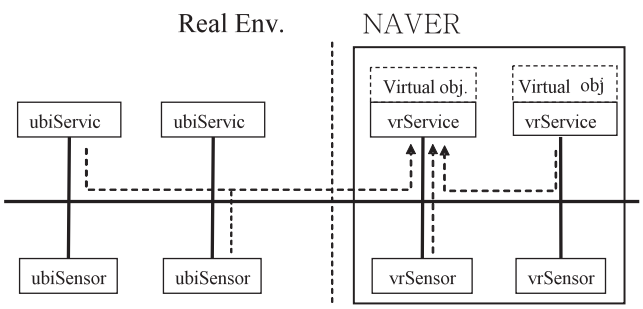


Fig. 2 Enhanced framework for a personalized user interface.

number of virtual objects distributed in NAVER.

For a demonstration, we implemented several virtual objects such as vrLocationSensors and vrMemo into a virtual heritage system developed in previous research [7]. Virtual objects were implemented in the form of dynamic link library (DLL). It was connected to XML parser of NAVER. Thus, using XML script language, we can add virtual objects to NAVER and assign parameters to them.

This letter is organized as follows: In Sect. 2, we explain the enhancement of the proposed framework. The implementation and experimental results are explained in Sects. 3 and 4, respectively. The conclusion and future works are discussed in Sect. 5.

2. Enhanced Framework for a Personalized User Interface

In this section, we explain the proposed framework and its functions in detail. We focus on the main enhancement part, integrating a unified context-aware application model for virtual environments (vr-UCAM 1.5) into PUI [9], [11]. vr-UCAM 1.5 is composed of vrSensor and vrService. vrSensor extracts the user's *preliminary context (PC)* containing the limited information about any changes in the virtual environments. vrService periodically integrates contexts containing meaningful contextual information, and compares the *integrated context (IC)* with *conditional context (CC)* defined by developers. Finally, the vrService decides on the state of a virtual object, and delivers a decision to a virtual object.

2.1 Model Definition

In the proposed framework, sensors and services in real and virtual environments are defined by a four-tuple $U = \langle I, S, E, C \rangle$, where I is a set of sensors, S is a set of services in both real and virtual world, C is a set of CC, and E ($\subseteq (I \times S) \cup (S \times I)$) is a set of connection lines between sensors and services. We do not distinguish between virtual sensors or services and real sensors or services in this model. The connections E between sensors and services are represented by an adjacency matrix (Fig. 3). It is a matrix with rows and columns labeled as services and sensors respectively, with a 1 or 0 in position (I_i, S_s) according to whether I_i and S_s are adjacent or not. Figure 4 shows a reaction model of a virtual object. Input from sensor I_i satisfying condition C_j activates a service S_j. In Fig. 4 (b), two inputs from sensor I_i, I_{i+1} satisfying condition C_j activate a service S_j.

$$\begin{array}{c} \\ I_1 \\ I_2 \\ \cdots \\ I_M \end{array}\!\!\begin{array}{c} \begin{array}{cccc} S_1 & S_2 & \cdots & S_N \end{array} \\ \begin{pmatrix} 0 & 0 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & 0 \\ 0 & 0 & \cdots & 0 \end{pmatrix} \end{array} \Rightarrow \begin{array}{c} \\ I_1 \\ I_2 \\ \cdots \\ I_M \end{array}\!\!\begin{array}{c} \begin{array}{cccc} S_1 & S_2 & \cdots & S_N \end{array} \\ \begin{pmatrix} 1 & 1 & \cdots & 0 \\ 0 & 0 & \cdots & 0 \\ \cdots & \cdots & \cdots & 1 \\ 1 & 0 & \cdots & 0 \end{pmatrix} \end{array}$$

Fig. 3 Adjacency matrix of connections E.

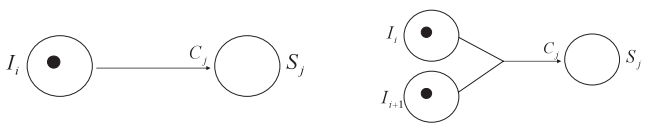
2.2 Accessing Virtual Objects

We need to support seamless connection between sensors or services in real and virtual environments. The network self-configuration manager (SCM) has the role to compose a multicasting group dynamically and to share a context among members of the group. It is embedded in each sensor and service. SCM applies a publisher-subscriber mechanism. When a sensor generates a *PC*, it notifies all services in the same active area. If a service needs to receive the *PC*, it responds to the notification. Then, the sensor can make a multicast group list. Finally, the sensor sends the *PC* to the services in the list. Figure 5 illustrates this mechanism.

The self-configuration manager is composed of a context publisher, a configuration manager, and a context subscriber. The configuration manager checks active area, type, and state of vrService and vrSensor. The context provider composes a multicasting group whenever a vrSensor generates a *PC*. After composing the group, it delivers the *PC* to each member over the network. The context subscriber decides to join or not to join the multicasting group according to the information in the configuration manager. After it decides to join, it receives the *PC* from vrSensors.

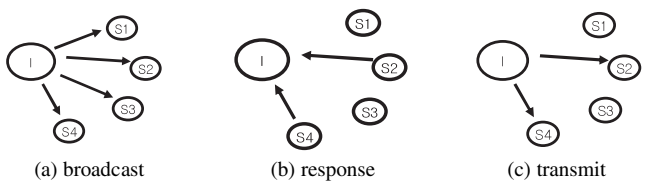
2.3 Action Customization of a Virtual Object

To customize a reaction of virtual objects, we describe a *CC* in a vrService. *CCs* are if-then rules which trigger events. For example, we can write a *CC* with a meaning of "if the location of a user A is 'where', then trigger the action A". We should know what kind of user-centric context can be captured from sensors. Then, vrService in vr-UCAM decides the action of a virtual object. Getting a *PC* from vrSensors, it starts to compare an *IC* with the *CCs*. Firstly, it checks whether any sub-context (5W1H) of *IC* and *CC* matches or not. Secondly, it finds all contexts with the same template through 'OR' operation. Thirdly, it compares sub-fields between contexts with the same template through 'AND' operation. Lastly, it selects an action for the virtual object de-



(a) one input case (b) two inputs case

Fig. 4 Reaction model.



(a) broadcast (b) response (c) transmit

Fig. 5 Publisher-subscriber mechanism.

scribed in the then-part of a matching *CC*. In this moment, *CC* should be included in the *PC* logically. When the comparison result is true, service provider executes actions of the virtual object.

3. Implementation

To demonstrate the usefulness of the proposed framework, we applied it to the virtual heritage system implemented in the previous work. It adopts a PDA as a personal user interface and several virtual objects in NAVER. We implemented a graphic user interface in the PDA to acquire user-centric context and to allow users to navigate the virtual environment. We implemented virtual sensors and virtual services as listed in Table 1 and Table 2. Every virtual object was implemented as a dynamic link library (DLL) and included in XML parser provided in NAVER.

The vrLocationSensor generates the location information in a virtual environment and broadcasts it to other services. We defined a position and sensing area in XML script. It observes a specific area from the position. When a user enters the specific area, it sends the position and orientation of user in the virtual environment. For example, when a user closes to a temple, it broadcasts the name of temple.

Table 1 Sensors and preliminary context.

Sensor	vrLocationSensor	PDA button
Sensing information (*PC*)	User ID	forward, backward, left, right
	Avatar appears	
	Avatar disappears	

Table 2 Virtual object and conditions.

Virtual object	Current state	Condition	Next state
vrMemoViewer	Wait	Avatar appears	Memo display
	Memo display	Avatar appears again	Next memo
		Avatar disappears	wait
vrMemoWriter	Wait	Avatar appears	Board display
	Board display	After a user write	Send it and wait
vrNavagation	Stand	forward, backward, left, right	Move

```
<module>
  <name>vrmemo</name>
  <class>vrmemo</class>
  <dso>vrmemo.dll</dso>
  <data>
  <vrmemo>vrmemo</vrmemo>
        <coord> 23, 40, 3</coord>
        <rot> 38 </rot>
        <port>8000</port>
  </data>
</module>
```

Fig. 6 Example of XML script of a virtual object.

We utilized location information to trigger services in real and virtual environments.

We made vrMemoWriter and vrMemoViewer in a PDA and a virtual environment respectively. vrMemoWriter was implemented with JAVA: It allows a user to write a note on the PDA and delivers the image file to a content management server. It has three modes; waiting, downloading, displaying. Firstly, it shows a note written by previous persons. When a new user writes a note, it downloads the image and displays it. We added it into NAVER by using XML-script as shown in Fig. 6. Both vrMemoWriter and vrMemoViewer are triggered when vrLocationSensor sends location information.

4. Experimental Results and Analysis

To evaluate the proposed enhanced PUI framework, we compared the previous approach (case A) with our proposed framework (case B). In case A, we applied a user's input and profile to NAVER though rv-interface. In case B, we applied a user's input and profile to individual virtual objects in NAVER. In our experimental setup, we allowed users to navigate the virtual environment. At a specific location, they wrote a note on a graphic user interface displayed on PDA. The notes written by persons who visit this virtual heritage were shown on the virtual object in NAVER. The memo was selected according to user's profile, such as user group. Figure 7 shows the virtual heritage system.

4.1 Performance Measurement of Accessibility

We estimated the accessibility of the self-configuration manager (SCM) of vr-UCAM 1.5. We implemented two network programs for a comparison and counted the number of sockets in each case. In the first case, we used a conventional socket communication. In the second case, we applied our SCM to the system. Figure 8 shows the results. The dotted line is the first case, and the solid line is the second. We found that we saved a network resource by using SCM. In the first case, if we add N virtual objects, it needs N network ports theoretically. Our SCM reduces the network resource less than $N(> |E|)$ based on an active area of a vrSensor. Thus, we can say that SCM enhances accessibility of the applications connected with a large number of sensors and services. However, there was a network delay due to the publisher-subscriber protocol used.

(a) (b)

Fig. 7 Virtual heritage. (a) A user writes a note on PDA, (b) a user reads a note displayed on the board in front of user.

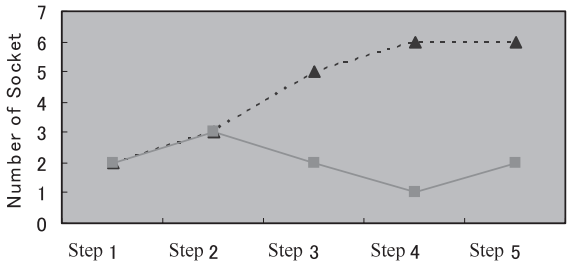


Fig. 8 The number of sockets on each step.

4.2 Customizability of Enhanced PUI

The reaction of virtual objects is diversified according to rules given by developers. If we assume that there are N virtual objects, each object has M actions, and U is a number of user contexts. The total number of services $|S|$ is $N \times M$. Then we can expect $U \times (N \times M)$ reactions in a whole NAVER in maximum. Moreover, if we apply joint rules, the number of possible reactions is increased. In addition, each virtual object can show different reactions on the same input since they may have different conditional context.

In out experiment, we implemented three virtual objects (vrMemoViewer, vrMemoWriter, vrNavigation) and 3, 2, 4 actions respectively. The sensors in real and NAVER acquired 7 inputs (two user IDs, one location, four directions). Thus, the maximum number of possible actions is $3 \times (3+2+4) \times 7$ totally. However, since each virtual object utilized a thread function to decide actions, we needed more computational power. This problem was partially solved by the SCM.

5. Conclusion and Future Works

In this letter, we proposed a enhanced framework for personalized user interfaces. It makes a feature of a symmetric structure for real and virtual environments by inserting virtual objects embedding vr-UCAM into NAVER. Compared to the previously developed framework, we provide enhancements for the ability of accessibility and customizability of PUI. This allows users to access and customize a large number of virtual objects. By implementing a virtual heritage system based on the proposed enhanced framework, we verified that PUI helps users interact with virtual objects. In future work, we will investigate technical issues to realize our proposed framework in ubiquitous virtual reality.

References

[1] A.S. Forsberg, J.J. Laviola, L. Markosian, R.C. Zeleznik, "Seamless interaction in virtual reality," IEEE Comput. Graph. Appl., vol.17 no.6, pp.6–9, Nov. 1997.
[2] R. Kadobayashi and K. Mase, "Seamless guidance by personal agent in virtual space based on user interaction in real world," PAAM98, pp.191–200, 1998.
[3] A. Schmidt, "Implicit human-computer interaction through context," Personal Technologies, vol.4, no.2, pp.191–199, 2000.
[4] A.K. Dey, "Context-aware computing: The cyberDesk project," Proc. AAAI'98 Spring Symposium, Technical Report, SS-98-02, pp.51–54, 1998.
[5] S. Kim, Y. Suh, Y. Lee, and W. Woo, "Toward ubiquitous VR: When VR meets ubiComp," International Symposium on Ubiquitous Virtual Reality, pp.1–4, 2006, CEUR Workshop Proc., ISSN 1613-0073.
[6] S. Agamanolis and V.M. Bove, "Multilevel scripting for responsive multimedia," IEEE Multimedia, pp.40–50, Oct./Dec., 1997.
[7] S. Jang and W. Woo, "Framework for personalized user interface by sharing user-centric context between real and virtual environments," IEICE Trans. Inf. & Syst., vol.E89-D, no.5, pp.1694–1701, May 2006.
[8] S. Jang and W. Woo, "ubi-UCAM: A unified context-aware application model," Context (LNAI/LNCS), pp.178–189, June 2003.
[9] S. Jang and W. Woo, "Unified context representing user-centric context: Who, where, when, what, how and why," ubiComp Workshop(ubiPCMM), pp.26–34, 2005, CEUR Workshop Proc., ISSN 1613-0073.
[10] C.H. Park, H. Ko, H. Ahn, and J. Kim, "NAVER: Design and implementation of XML-based VR Framework on a PC cluster," The 8th International Conference on Virtual Systems and MultiMedia 2002, Gyeongju, Korea, Sept. 2002.
[11] S. Lee, Y. Lee, S. Jang, and W. Woo, "vr-UCAM: Unified context-aware application module for virtual reality," ICAT04, pp.495–498, 2004.