특별기고 〈해외편〉

Technologies for Computing Everywhere Environments

Ken Sakamura[1] Noboru Koshizuka[2]

목 차

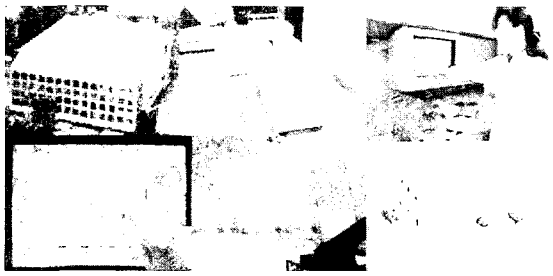
1. Introduction

The phrase "ubiquitous computing" has become popular ever since Mark Weiser used it in an article titled "The Computer for 21st Century" in the September 1991 issue of Scientific American. However, the concept itself is not new. We have proposed essentially the same idea in the early 1980's. In 1984, we have started a ubiquitous computing research and development project in Japan, TRON Project (The Real-time Operating system Nucleus)[1,7]. The goal of the project is to realize a computerized environment in which small computers are embedded and cooperate with each other. This environment will support any activities of our daily life. For the initial lack of better phrase and to distinguish our work from Mark Weiser's work, we have used "Computing Everywhere" to describe the environment which our project aims to achieve.

The TRON project has released several kinds of real-time and embedded operating systems such as ITRON, BTRON, JTRON and CTRON. These OSs have got the No. 1 market share of the real-time and embedded systems in Japan. They are used for most of Japanese mobile phones, digital cameras, automobiles, consumer electronic products, ISDN switching nodes and so on. The TRON Project has also developed a large-scale experimental ubiquitous computing environment as applications of these real-time and embedded systems. TRON-concept intelligent house is the most notable one, which consists of thousands of computer nodes with sensors and actuators.

1) YRP Ubiquitous Networking Laboratory
Interfaculty Initiative in Information Studies, The University of Tokyo
2) YRP Ubiquitous Networking Laboratory
Information Technology Center, The University of Tokyo

These nodes communicate and cooperate with each other, and provide comfortable environment for the resident persons (Figure 1).



(Figure 1) TRON-concept intelligent house, intelligent building, and intelligent city

In order to realize the ubiquitous computing environment, it is important to have very small electronic tag(called uID below), small hand-held terminals called ubiquitous communicator, and the network infrastructure. We explain the rational for the needs of the former two in the following.

From our active research in the TRON project, we have learned valuable lessons. To realize a computing everywhere environment, there are three important issues. First, the most essential basic technology for computing everywhere environment is context awareness, which enables computers to recognize the state and situation of the real-world automatically. Although there are many basic technologies for the context awareness, one of the most realistic methods is to embed ultra small tags in every object in the world, and computers get various kinds of real-world information by communicating with these tags. RFIDs, smart cards, and active sensor chip are examples of them.

Second, in building computing everywhere environment, it is necessary to develop huge number of computing nodes efficiently in a short period of time. Among the current embedded system manufacturers, there is a problem with individually developed kernels and middleware that are compatible with the different kinds of hardware for embedded systems. Development time for functional upgrade is prolonged, and development costs increase as debugging time increases. All of these are becoming a big problem among manufacturers.

Lastly, it is important to ensure security at the user side. In a computing everywhere environment, various devices will be connected to the network from houses and buildings. It is therefore necessary to ensure network security so that tapping, falsification and/or the disguising of identity by others can be prevented.

In 2002, we have established a new project, T-Engine Project [5] in order to promote an open hardware and software platform with the aim of achieving a computing everywhere environment where everything contains a computer and is connected to a network. This project consists of three subprojects: ubiquitous ID project[6], T-Engine platform project[3, 4], and eTRON project[2]. Each of them corresponds with

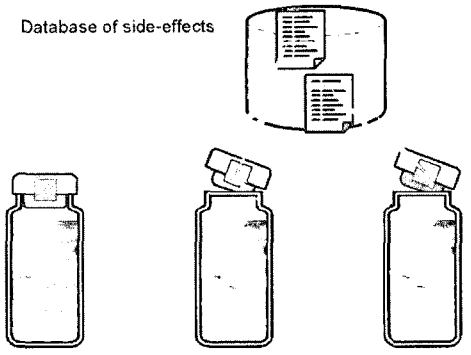
each problem mentioned above. The rest of this paper presents our philosophy and an overview of the T-Engine project, especially focusing on these three topics.

2. Ubiquitous ID

(Figure 2) Ultra small chips

Recently, with the progress of hardware technologies, computers have been becoming much smaller. Even a full.fledged computer equipped with CPU, memory and communication interfaces is implemented with a small single VLSI chip. A memory tag without CPU is smaller than 1mm x 1mm (Figure 2). These ultra small chips are driving research and development of computing everywhere environments.

In the computing everywhere environment, these small chips are embedded into all sorts of "things." They communicate and cooperate with each other to control our living environment or to provide rich information services to us. The computer automatically recognizes the condition of the real world, and conducts various information processing and operation according to the condition. For example, if we attach a ultra small tag to a medicine bottle, and moves two bottles closely, these computers exchange information one another. If the combination of these medicines is not good, they automatically recognize this inadequate condition, and warn users by calling their mobile phone (Figure 3).



(Figure 3) Warning from medicine bottles

An important technology for this kind of applications is to recognize conditions of the real world automatically(context-awareness). To recognize real world, computers must be able to distinguish "things." For this purpose, ubiquitous ID technology offers an id system which provides a unique identifier (uID: ubiquitous ID) to every "thing."

2.1 Ubiquitous ID

The most basic mechanism for context-aware computing in computing everywhere environment is to attach a real world object a unique ID (ubiquitous ID: uID), and to

implement the uID so as to be readable by computers.

There have been various systems of numbers to be attached to "things." Examples include bar code forms, such as the UPC code, and the EAN code, which are used in such fields as physical distribution. As codes that are attached to books and periodicals, ISBN and ISSN codes are famous. In these code systems, a number is allocated for a product type, not for each product. For example, all "Hamlet" books written by William Shakespeare published by a certain company have the same ISBN. The ISBN code does not distinguish each book. On the other hands, uID is a code allocated for each "thing." So, every book has its unique uID, and no two objects have the same uID. This unique object id system is effective especially for managing products each of which has its own characteristics such as its warranty period and best consume term/time.

uID is basically a 128-bit-long code, and uID also has a framework for extension in 128-bit units. Therefore, it can be 256-, 384-, 512-bit-long, and so on. uID is meta-code, which can absorb various existing ID codes such as UPC, EAN, and JAN in the barcode form, ISBN and ISSN for books, IP address given to computers connected to the Internet, and telephone number for telecommunication terminals. uID contains these id systems as much as possible, by using its large id space.

2.2 uID Carrier Devices

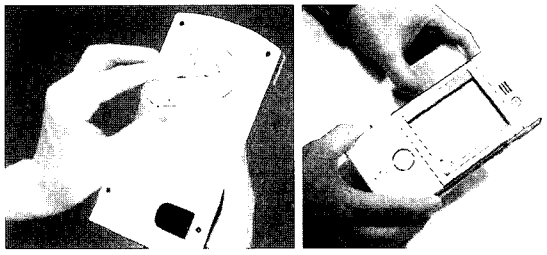
As the mechanism to attach uIDs to "things", very small memory devices with contactless communication capability are suitable. The most important device for this purpose is "RFID" (Radio Frequency IDentification). By storing uID into an RFID, and attaching it to "thing", we can combine the "thing" and the "uID." The main advantage of RFID is the capability to be read the stored data with no contact from remote position. Additionally, multiple RFIDs with anti-collision mechanisms allow to be read together at the same time. On the contrary, there are some disadvantages. First, the main disadvantage is the cost. RFIDs are much more expensive than barcode. Another disadvantage is data security. Unauthorized users might read the data in RFID. Additionally, some RFIDs do not work well with objects with much water or with objects made from metals. This is because radio waves in some frequency don't travel inside for water or metals.

So far, we have adopted various kinds of tags as uID carrier devices. For example, we use even barcodes as the cheapest uID tags. As for the most expensive device, we use smart card with cryptographic functions. uID framework supports various kinds of uID carrier devices and we can select the most suitable one among these according to requirement of applications (Table 1).

〈Table 1〉 Classification of uID Carrier Tags

Class	Name	Description
Class 0	Visible Tags	Barcode, two dimensional bar code,
Class 1	Low-level Passive RFIDs	Read-only RFID without battery
Class 2	High-Level Passive RFIDs	Writable RFID without battery
Class 3	Low-Level Smart Cards	Contactless smart cards without PKI security functions
Class 4	High-Level Smart Cards	Contactless smart cards with PKI security functions
Class 5	Low-Level Active Tags	RFID with battery and only uni-directional communication mechanism
Class 6	High-Level Active Tags	RFID with battery and bi-directional communication mechanism

2.3 Ubiquitous Communicator



(Figure 4) Ubiquitous Communicator

A ubiquitous communicator is a terminal for communication between the computing everywhere environment and persons (Figure 4). For this purpose, it provides a user a communication function in anywhere at anytime.

2.3.1 Multiple Communication Channels

Since ubiquitous communicator is designed as a communication machine, it is equipped with multiple communication channels as follows.

2.3.1.1 Short-distance communications function

Short-distance communication means a function of communicating with uID tags, including RFIDs, contactless smart cards and barcodes. Ubiquitous communicator contains some of these communication interfaces. Today, there are various standard of RFID and contactless smart card communication. Popularly used communication technologies have their reason to survive in the market and users need the features supplied, and so we must support each such popular communication technology unless it is deemed impossible. Thus, we are developing multi-modal system for short-distance communication which can communicate several kinds of RFIDs and contactless smart cards that are authorized by the Ubiquitous ID Center. Unauthorized devices are not supported.

2.3.1.2 Long-distance communications function

This is communications function to connect to a long-distance communications network, to receive information on "things" that have uIDs attached to them, and additional service on the "things." Examples are local area network as IEEE802.11b and Ethernet, personal area network as Bluetooth, and wide area digital telecommunication channel such as W-CDMA and PHS.

2.3.2 Seamless Communication

Seamless communication is an important feature for ubiquitous communicator, in which ubiquitous communicator can switch communication channels or underlying

communication protocols transparently from the upper communication layer. For example, a ubiquitous communicator is equipped with IEEE 802.11b interface and W-CDMA interfaces. It will use IEEE 802.11b interface for in-house communications. While it is using the IEEE 802.11b communication channels, its user walks out of the building. The wireless LAN communication channel will stop, but the communication will continue using W-CDMA communication channel. This channel transition must be done seamlessly.

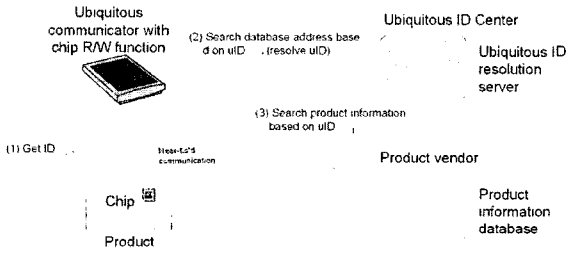
2.4 Security Problem in the uID Technologies

In a computing everywhere environment, it is necessary to ensure network security and to protect private data. For example, if malicious users are succeeded in sniffing networks communications for product database retrieval, they can know who bought which products.

To realize the network security, ubiquitous ID framework adopts the eTRON (Entity and Economy TRON) architecture, which provides a total framework for cryptography communications and online authentication. We explain eTRON in Section 4.

2.5 Ubiquitous Information Services

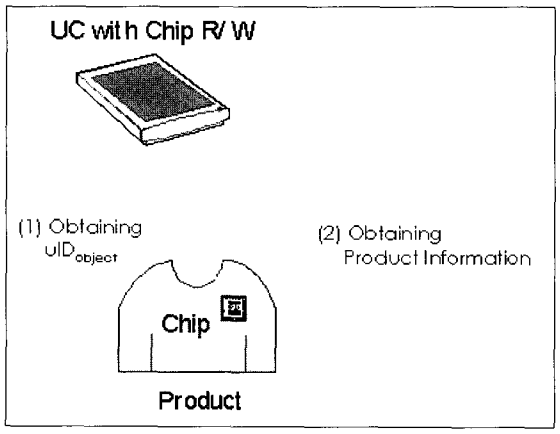
Much real-world context information can be obtained from the ultra small chip embedded in several objects. However, the amount of information stored in such ultra small chips is so limited that some backend system support is necessary for ubiquitous information service. Our ubiquitous ID framework provides a large-scale distributed system for this purpose. Figure 5 shows the mechanism which enables ubiquitous communicators to retrieve various information on "things" with uIDs and to receive services related to the "things."



(Figure 5) Ubiquitous information services using ultra small chips and ubiquitous communicators

First, a user moves the ubiquitous communicator over the "thing", then it reads uID from RFID attached to the "thing" using the short distance communication channel (Phase 1). Second, it sends the uID to "uID resolution database" located in Ubiquitous ID Center using the long distance communication channel, then it obtains the network address such as an IP address in which "product information database" of the "thing" is placed (Phase 2). Lastly, it retrieves the product information database" and obtains the information of the "thing"

(Phase 3). Reversing the flow of information, ubiquitous communicator can store or update the information on the product information database in the same mechanism.



(Figure 6) Ubiquitous information services using ultra small chips and ubiquitous communicators

If the RFID provides enough memory in it for the "thing", ubiquitous communicator retrieves the information from the RFID directly (Figure 6). Hybrid architecture is also possible, in which the information is stored in RFID chips and remote database in separation.

3.T-Engine Platform

The T-Engine platform consists of the hardware platform, T-Engine, and the software platform, T-Kernel.

3.1 T-Engine Hardware

The T-Engine platform is mainly suited for real-time embedded systems in ubiquitous computing environments. In addition, these systems are suitable for an extensive range of applications. To cover such a vast application range, the four established sets of hardware platform specifications shown in Figure 7 handle different sizes, power levels, and processing capabilities.

(Figure 7) T-Engine, μT-Engine, nT-Engine, and pT-Engine

3.1.1 T-Engine

The standard T-Engine platform is suited for mobile information appliances such as next-generation mobile phones and electronic-book readers. Suitable devices would have a graphic display and an input device, providing an advanced user interface; run on a battery; and have a wireless communication function. T-Engine physical specifications for this platform define the board size (75 x 120mm), some component positions, and the peripheral device's interface connector sizes. Multiple vendors has already released T-Engine product as shown in Table 2-1.

3.1.2 micro (μ) T-Engine

The μT-Engine is a standard platform for computers embedded in such units as home appliances and in measuring equipment. Its board size is 60 x 85mm, and it does not necessarily need a memory management unit(MMU). We apply the same specifications used for T-Engine to the expansion connector to enable the sharing of hardware components between both engines. Unlike T-Engine, a graphic display is not required because μT-Engine devices would use a simple user interface. This platform also provides an eTRON chip as standard. Multiple vendors has already released μT-Engine product as shown in Table 2-2.

3.1.3 nano (n) T-Engine

The nT-Engine is an inexpensive, coin-sized hardware platform intended for such nodes as lighting fixtures, sensors, and window controllers. It consists of a processor core, network interface, and peripheral functions. A hardware library integrates the necessary peripheral functions. This platform serves as a processor core, combining functions to create a target system.

3.1.4 pico (p) T-Engine

The pT-Engine is a chip-shaped platform with a sensor function and a wireless or optical communication function. Only several millimeters in size, pT-Engine fits into a large variety of unpowered objects, including clothing, desks, chairs, paintings (hanging on walls), dishes, and drug bottles. Rather than just a simple RFID device, pT-Engine is also a computer with processing capability. For example, this chip could attach to a wine bottle in transit to monitor and record temperature and vibration data for quality assurance. To run on unpowered objects, the chip might use electromagnetic energy from a communication medium or power generated by microelectromechanical systems using micropulsation.

〈Table 2〉 Current T-Engine and micro(μ) T-Engine Hardware Products

〈2-1〉 T-Engine Specification

Models	Vendors	CPU	Clock (MHz)	RAM (MB)	Flash ROM (MB)
h101	Renessas and Hitachi ULSI	SH7727 (SH3-DSP)	96	32	8
h102	Renessas and Hitachi ULSI	SH7751R (SH4)	240	64	8
n101	NEC	VR5500-400 (MIPS)	400	64	16
y101	YDC	S1C3800 (ARM720T)	72	32	8
y102	YDC	MC9328MX1 (ARM920T)	200	64	16
y103	YDC	ML7101 (ARM920T)	TBD	TBD	TBD
t101	Toshiba	TX4956 (MIPS4)	TBD	TBD	TBD

〈2-2〉 micro(μ) T-Engine Specification

Models	Vendors	CPU	Clock (MHz)	RAM (MB)	Flash ROM (MB)
h301	Renessas and Hitachi ULSI	SH7145 (SH2)	50	1	1
m301	Renessas	M32104 (M32R)	216	16	4
n301	NEC	VR4191-200 (MIPS)	200	32	16
f301	Fujitsu &YDC	MR93403 (FR400)	266	64.	16

3.2 T-Kernel

T-Kernel is the standard real-time operating system for (standard) T-Engine and mT-Engine. Neither nT-Engine nor pT-Engine use T-Kernel or its equivalents because both platforms have extremely small amounts of computing resources.

3.2.1 Design

As described previously, the T-Engine project seeks to apply a real-time embedded kernel to a large variety of systems. Even for just T-Engine and mT-Engine, applications range from a small, single-purpose embedded unit to a large-scale server system. T-Kernel is so scalable that it can cover-with a single kernel architecture-a wide array of systems. These systems range from lightweight, compact real-time systems that operate under severe cost and resource restrictions (light systems) to advanced server-oriented systems equipped with a virtual memory unit (advanced systems).

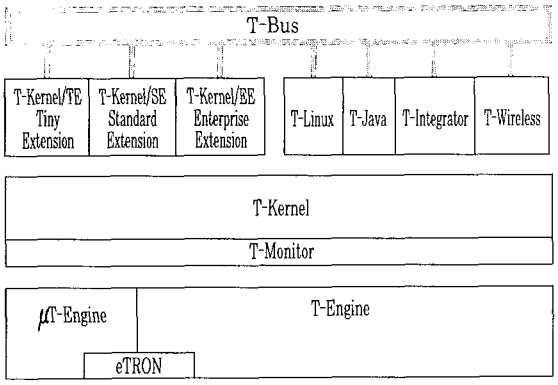
The T-Kernel design concept provides two advantages: It helps individual software applications implement common functions for the increasingly popular large server systems and small embedded systems. It can also manage small embedded systems in their initial development phase, before they evolve into larger systems.

To give T-Kernel high scalability, we introduced the concept of subsystem. The subsystem permits kernel extension and the construction of both light and advanced systems on the same kernel. T-Kernel also has a function not provided in ordinary real-time kernels: the dynamic management of kernel objects such as tasks and semaphores to facilitate the addition of miscellaneous middleware and device drivers.

3.2.2 Extensions

We have already been developing several middleware components upon T-Kernel as shown in Figure 8. These components can be classified into two classes. The first one is the native middleware, called T-Kernel native extensions. We have three native extensions: T-Kernel/TE (Tiny Extension), T-Kernel/SE (Standard Extension), and T-Kernel/EE (Enterprise Extension). The second one is a class of ported middleware, called ported extensions. They include T-Java, T-Linux, T-Integrator, and T-Wireless. T-Kernel alone permits the construction of lightweight and compact conventional real-time systems, whereas T-Kernel and these middleware components combined together can realize advanced systems.

To realize interoperability among these extensions, we need high-level communication and synchronization mechanisms. For this purpose, we have established T-Bus framework. T-Bus is a standard message passing mechanism similar to CORBA (Common Object Request Broker Architecture).

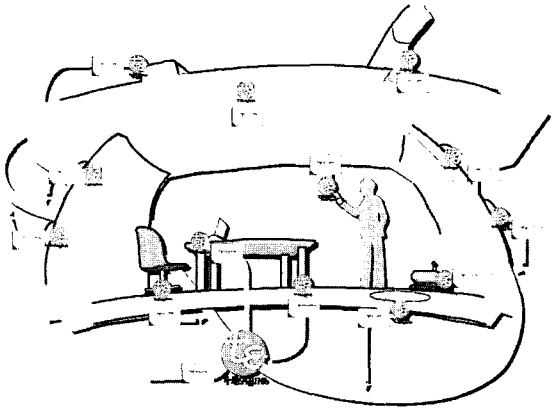


(Figure 8) T-Engine Software Architecture

4. eTRON

In a ubiquitous computing environment, various devices will be connected to the network from houses and buildings. It is therefore necessary to ensure network security and to protect private data so that tapping, falsification and/or the disguising of identity by others can be prevented. For example, if malicious users are succeeded in sniffing networks communications for product database retrieval, they can know who bought which products.

To realize the network security, ubiquitous ID framework adopts the eTRON(Entity and Economy TRON) architecture, which provides a total framework for cryptography communications and online authentication. With eTRON, the tamper-proof hardware is used as the base(eTRON nodes) and it works as a special data storage for valuable information. Private information is stored in the eTRON node. Two eTRON nodes can transfer the data with each other, and the all communications for the data transfer are authenticated and crypted communications. Thus, if anyone sniffs the communication data, he/she cannot know the contents of the communications. As for the data storage, eTRON node is manufactured as tamper-proof hardware. This protects stored data from malicious users. Ubiquitous communicators, object databases, and uID resolution database support eTRON. All communication among them are protected by the eTRON communication mechanism.



(Figure 9) Ubiquitous computing environment realized by T-Engine platforms and eTRON

5. Summary

This paper describes an introduction and overview of our computer everywhere environment technologies including T-Engine, T-Kernel, eTRON and ubiquitous ID.

Before concluding the article, we would like to offer our perspective on the acceptance of the ubiquitous computing environment by our society. Since the ubiquitous computing

is the computerization of the real world, we must respect the local culture, i.e., the local ways and customs with which people live. Otherwise, such computerization won't be accepted by the society. Such localization should be carried out by each local region. Internet has cut itself from the real world and has used the same technology all over the world. In contrast to this, due to the different cultures, we believe that ubiquitous computing will see variations based on the differences of human culture that surrounds it.

We also observe that many competing technologies have to be devised, tested and adopted until reasonably acceptable ubiquitous computing is realized. But during the transition, these competing technologies are often deployed simultaneously to offer services and so we don't expect cut-throat competition among the developers of competing technologies. For example, there are various types of RFID tags commercially available, and they use different frequencies such as 13.56 MHz UHF(860~950 MHz), and 2.45 GHz. Today, trade journal reporters seem to be interested in which tag will survive. To us, their interest look misplaced. We believe that all of these will survive since the objectives of these tags to solve are different, and interference from environment(water, and metal are two major interference sources) often dictate the preference of one tag over the other. With this in mind, our prototype Ubiquitous Communicator will support three frequency ranges to talk to different RFID tags.

To carrying out the R&D activity, standardization, and promotion of these technologies, we have established T-Engine Forum[5] (Chairman: Ken Sakamura). The forum has started with 22 member companies in June, 2002. Today, July 1, 2003, the number of forum members is more than 200 all over the world.

References

[1] K. Sakamura, "The TRON Project," IEEE Micro, vol. 7, no. 2, Apr. 1987, pp. 8-14.

[2] K. Sakamura and N. Koshizuka, "The eTRON Wide-Area Distributed-System Architecture for E-Commerce," IEEE Micro, vol. 21, no. 6, Dec. 2001, pp. 7-13.

[3] K. Sakamura and N. Koshizuka, "T-Engine: The Open Realtime Embedded Systems Platform", IEEE MICRO, vol. 22, no. 6, Dec. 2002.

[4] K. Sakamura and N. Koshizuka, "T-Engine: The Open, Real-Time Embedded-Systems Platform for Ubiquitous Computing", in Proc. VLSI Circuit Symposium, June 2003.

[5] T-Engine Forum Home Page, http://www.t-engine.org/

[6] Ubiquitous ID Center Home Page, http://www.uidcenter.org/

[7] TRON Project Home Page, http://www.tron.org/

Short Profile

Ken Sakamura

Born in Tokyo in 1951. He obtained his Ph.D. degree in Electrical Engineering from Keio University in 1979. In 1984, he organized The Real-time Operating system Nucleus (TRON) Project, and since that time has served as its leader. TRON is a total computer architecture for the ubiquitous computer networking society of the future. He has also promoted TRON concepts on his own in diverse ways by undertaking design projects in many fields, such as the design of furniture, industrial products, building architectures, and even cities. At present, he is a professor of the Interfaculty Initiative in Information Studies at the Graduate School of the University of Tokyo, director of the YRP Ubiquitous Networking Laboratory, and a fellow of the IEEE Computer Society.

Noboru Koshizuka

Born in 1966 at Tokyo, Japan. He is an associate professor at the Information Technology Center, University of Tokyo. His research interests include ubiquitous computing, secure systems, operating systems, and interactive systems. Koshizuka has BS, MS, and Ph. D, all in information science, from the University of Tokyo. He is a member of IEEE and ACM.