EPISODE: AN EXTREME PROGRAMMING METHOD FOR INNOVATIVE SOFTWARE BASED ON SYSTEMS DESIGN AND ITS PRACTICAL STUDY

Takaaki Goto[1], Kensei Tsuchida[2] and Tetsuro Nishino[1]

[1]Graduate School of Informatics and Engineering, The University of Electro-Communications, Japan
[2]Faculty of Information Science and Arts, Toyo University, Japan

ABSTRACT

In software development, the waterfall model is commonly used, especially for large-scale software systems. For smaller-scale software development, agile software development approaches such as extreme programming or scrum are used. Traditional software development methodologies are mainly targeted toward customer-centric development, and therefore, new software methodologies are often not well received in the industry. In this study, we propose a new software development methodology that is aimed at developing innovative software using artificial intelligence (AI), idea creation, value engineering, and systems design. The name of our method is named as EPISODE (Extreme Programming method for Innovative SOftware based on systems DEsign). EPISODE supports the efficient and creative development of open source software (OSS) by small groups. Moreover we describe an evaluation of EPISODE in a class.

KEYWORDS

Software Development Methodology, Value Engineering, Open Source Software

1. INTRODUCTION

In software development, the waterfall model is commonly used at present. Recent years have, however, witnessed the increasing use of agile software development methodologies, especially for small- or mid-scale projects and even in some large-scale projects.

Both the waterfall and agile software development models feature developers and clients, where the developers obtain the project requirements from the clients. After requirement analysis, the developers design and develop systems according to the obtained requirements.

Nowadays, there is an increasing demand for software and applications for tablets and other small smart devices. This software is developed by not only established software companies but also individual or small groups of developers who may not have clients. In the latter case, the developers have to create new ideas by themselves for software requirements instead of obtaining requirements from clients. However, the waterfall and agile development models do not facilitate new and innovative software development methods. Moreover, small software development groups have limited resources available to themselves, and therefore, there is a strong need for efficient development methods for such groups.

DOI : 10.5121/ijsea.2014.5501

In this study, we propose a new software development method for developing innovative software using artificial intelligent (AI), idea creation, value engineering, and systems design. Moreover we describe a practical study of our method on graduate school course.

2. BACKGROUND

The software development process plays an important role in software development. Some software development processes have been proposed thus far.

2.1 Software Development Process

2.1.1. Waterfall model

The waterfall model for software development involves six steps: “requirement analysis,” “external design,” “internal design,” “implementation,” “test,” and “operation and maintenance.” Figure 1 shows the flow of the waterfall model.

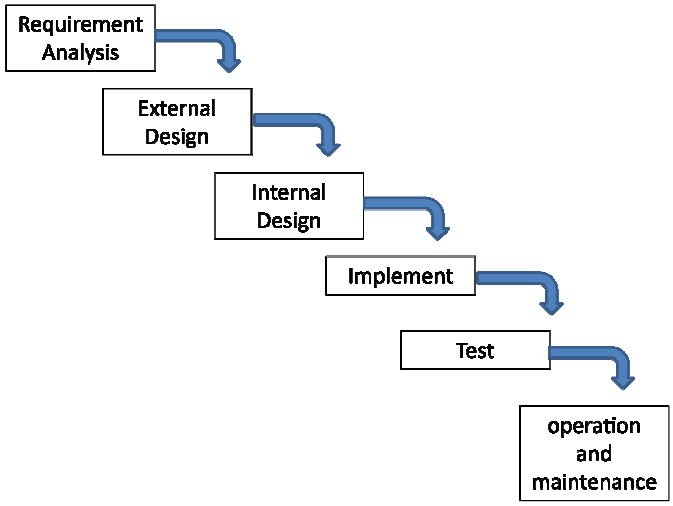

Figure 1 Waterfall model

2.1.2. Agile development

Agile development is a lightweight software development methodology. In this approach, developers develop the desired systems by dividing the overall project into small functions. In agile development, the development cycle from the design to the test phase is performed for one small function. Then, the developers shift to the next function. By using this approach, developers can present the software to their clients from the early stage of the project.

A popular agile development approach is extreme programming (XP). Figure 2 shows a software development cycle in XP. The cycle consists of the following stages: ”planning,” ”design,” ”coding,” and “test.” This cycle is used for extracting one function from the desired system, and it is repeated until the system development is completed.

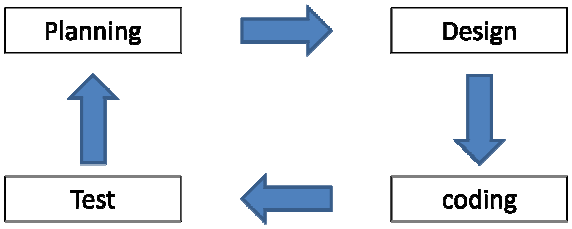

Figure 2 Extreme programming

2.2. Methods for innovating

2.2.1. Brainstorming

Brainstorming was first proposed by Alex Faickney Osborn in 1938. Brainstorming is a method used to generate new ideas by a team composed of around 10 members. To generate new ideas efficiently, the number of participants, constructing groups, implementation method, and so on are defined in this method. Brainstorming, as defined by Osborn, is performed under the following principles [1].

1. Withhold criticism
2. Welcome abandon
3. Focus on quantity
4. Combine and improve ideas

2.2.2. Affinity Diagram

A method for analyzing obtained ideas by brain storming using grouping is the affinity diagram [2]. This method enables groups to formulate new ideas that cannot be found from one idea by constructing groups of obtained ideas.

2.2.2 Value graph

Ishii et al.[3] proposed the concept of value graphs. A value graph is a tool used for the managing of finding value and requirement functions. It can be applied to the generation concept and its selection. Figure 3 shows an example of a value graph.

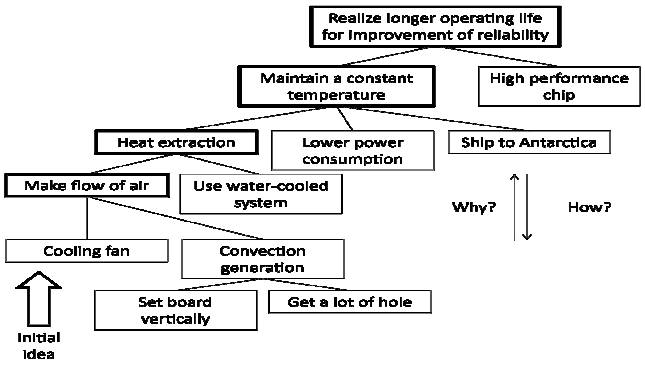

Figure 3 Example of value graph **Error! Reference source not found.**]

A value graph can be used to analyze the value of some object by constructing a graph from one idea with thinking such as “why?” or “how?” In Figure 3, “Cooling fan” is the initial idea, and then, an analysis is performed by thinking “why” or “how.” In this case, the cooling fan is needed to promote the flow of air. An alternate method to promote the flow of air is to generate convection.

2.3 Practical Software Development Education and UEC Software Repository

2.3.1 Practical Software Development Education

Nowadays, many attempts are being made to implement PBL (project-based learning) in software development education. At the University of Electro-Communications, we are working on a practical form of software development education that provides students with a more rounded set of practical skills and self-motivation for software development. By developing teaching materials for self-motivated practical education, with a focus on open-source software development and implementing an education program that incorporates these materials, our goal is to provide practical education using the UEC software repository [4] as an autonomous practical educational resource to cultivate highly creative developers with excellent research and development skills.

2.3.2 UEC Software Repository

At the University of Electro-Communications, a great deal of software is written every year through the course of student research and educational activities. Normally, most of this software is managed and stored on servers in the research laboratories; however, as the software is generally lacking in documentation and is not developed with third-party use in mind, it tends to be used only by the original developers. Once the developers have graduated, there are very few opportunities for this software to be reused. As one approach to addressing these problems, we are working on the construction of a UEC software repository.

Figure 4 Screenshot of UEC software repository

The UEC software repository is a database for the centralized in-house management of software developed at the university. Ordinary users can download and use software stored in this repository by accessing it through site searches. The repository system has the following features:

1. Software registration, search, and download functions
2. Research paper registration, search, and download functions
3. Download ranking functions
4. User management functions
5. Development support functions

This repository supports three types of users with different privileges:

1. Software project leader
2. Software users
3. System administrators

Table 1 lists the features available to each type of user. Software project leaders are permitted to submit software to the repository. To qualify as a software project leader, a user must have studied for and completed a graduate course in the practical software development course, which is described above. Software project leaders can also search for and download research papers.

Table 1 Features available to each user type

Software project leader	Registering, searching, and downloading software, Searching and downloading research papers
Software users	Searching and downloading software, Searching and downloading research papers
System administrators	Managing software, Managing research papers, Managing users

3. ISSUES ON APPLYING EXISTING SOFTWARE DEVELOPMENT METHODOLOGY TO OUR TARGET

3.1 Issues on applying waterfall model to our target

When developing software by a small group based on existing software development methodologies, two main problems are encountered.

1. In this study, we target the development of new software including innovative ideas proposed by the developing team members. Because clients do not exist, there will be some possibility to add or modify functions in the development process. Therefore, if we apply the waterfall model to our target (developing innovative software), the development cost will increase because of backtracking during development.
2. Creating a software document such as a requirement definition document or external design document is too heavy and time-consuming a workload for developers because of its volume. Creating many documents with only a small number of developers is difficult.
3. In the waterfall model, it is common for persons with professional skills to be associated with each development phase (requirement definition, design, test, etc.). However, it is

difficult to assign team members to such development phases because of the limited number of team members available.

4. The waterfall model basically targets large-scale development. Therefore, human wave tactics can be applied. However, it is not applicable to development with a small group. A small group needs to decide priorities for developing functions and to develop efficiently using development support tools

3.2 Issues faced in applying agile development to our target

The Information-Technology Promotion Agency (IPA) reported a survey of application examples on the practice of agile development [5]. In their report, they showed application examples on the practice of agile development. The following nine situations are described in their report:

1. Short-term development
2. Project that has a lot of ups and downs in scope
3. Requirement of quality is high
4. Requirement of cost is difficult
5. Skills of team members are undeveloped
6. Teams treat new technical field or new business knowledge
7. Teams have some newcomers
8. Teams develop using distributed development methodology
9. Teams use agile development for the first time

Our software development methodology mainly targets small groups in universities. Therefore, “Skills of team members are undeveloped,” “Teams treat new technical field or new business knowledge,” “Teams have some newcomers,” and “Teams use agile development for the first time” can be true. To solve these problems, an agile coach is required to train the team. However, it is difficult to acquire a sufficient number of agile coaches in universities.

On the other hand, software development projects in universities change project members every year. However, the agile development method does not mention how to share knowledge and know-how and how to maintain software documents.

The waterfall and agile development models involve another problem when applied to our target. Our target project does not have a client. However, both these models assume a client in order to obtain the system or software specifications. Therefore, a method for generating new ideas is not considered in these development processes.

4. PROPOSED METHOD

4.1 Abstract of EPISODE

In this study, we propose a new software development methodology that targets developing innovative software using artificial intelligence (AI), idea creation, and value engineering. This new method supports the efficient and creative development of open source software (OSS) by small groups. The name of our method is named as EPISODE (Extreme Programming method for Innovative SOftware based on systems DEsign).

4.2 Development cycle of EPISODE

The development cycle of our proposed method is shown in Figure 5. In this cycle, development is performed by the following process: "Planning" → "Design" → "Coding" → "Evaluation." To finish developing the software, the cycle is continued. We adopt the idea creation and value engineering methods in XP.

In the planning phase, story extraction is performed. First, team members create ideas for target software using the brainstorming method (divergence). Then, the KJ method is used to group the obtained ideas and to derive insights from the constructed groups (convergence).

When the goal of the target software is decided by the brainstorming and KJ methods, then the developers extract the story of the target software. At that time, the related software is needed for analysis. Then, we obtain information about the related software from OSS. Many OSS do not have appropriate documents. In this case, an automatic document generation tool is used to obtain documents from the source code of OSS.

In the design phase, task division is performed using the story obtained in the planning phase. A brainstorming tool is also used in this phase to divide the story and design. A digital book tool can be used when developers want to record their insight or know-how. The digital book includes not only software structure information and algorithms but also developer notes.

In the coding phase, developers implement tasks (prototyping) that are obtained in the design phase. Programming is performed using the "pair programming" method. Developers note their findings using the digital book tool.

In the evaluation phase, the developers themselves evaluate their developed software. After evaluating the software, the developers also check the value of the obtained software in terms of whether it satisfies the requirements by using a value graph.

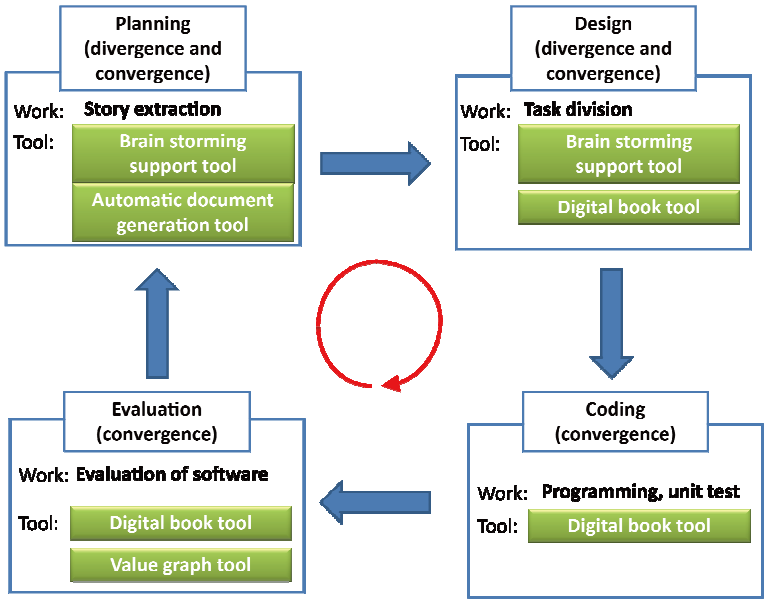

Figure 5 Development cycle of EPISODE

EPISODE can be used for two software development patterns such as "development from scratch" and "alter existing software." In the process of "development from scratch," students develop their software following the steps: Planning → Design → Coding → Evaluation. On the other hand, analysis of existing software should be done first of all on "alter existing software," that is the steps Evaluation → Planning → Design → Coding → Evaluation are needed for such development.

4.3 Development support tool

Herein, we describe some proposed tools that are used in each phase. In our method, the following four tools are used:

1. Brainstorming support tool
2. Automatic document generation tool
3. Digital book tool
4. Value graph tool

These tools are under development, and therefore, we explain their concepts.

4.3.1. Brainstorming support tool

The brainstorming support tool is used in the planning and design phases. Developers can discuss their new ideas by using this tool. This tool includes two functions: brainstorming (divergence) and KJ method grouping (convergence). This tool can be used by one or many developers.

4.3.2. Automatic document generation tool

The automatic document generation tool is used in the planning phase. If there already exist some software related to the target software, the developers have to check and understand this software. Most OSS has only source code. This tool generates useful software documents from the OSS's source code.

This tool targets an object-oriented language for analysis. This tool consists of the following two components: (1) UML generation module and (2) source code summarization module based on a natural language processing method.

4.3.3. Digital book tool

A digital book tool is used for understanding software and sharing notes and insights regarding the target software. The digital book generated using this tool also includes an index function and a writing annotation function to support efficient development. This tool is used in the design, implementation, and evaluation phases.

4.3.4 Value graph tool

The value graph tool is used in the evaluation phase. This tool draws a value graph function and adjusts the layout function. This tool can be used by one or many developers.

5. A CURRICULUM FOR A CLASS IN OUR UNIVERSITY BASED ON EPISODE

We have started to apply our method EPISODE to a graduate course practical software development education class called "Fundamentals of Practical Software Development." In this class, students develop OSS and experience the software development process from the requirement definition to the test phases based on EPISODE. We prepared four small OSS which will be base systems for software which students develop. The four OSS are "Calculator," "Support soft for dyschromatopsia," "Ledger sheet tool," and "breakout game with physical controller." These OSS include source code, binary code, and software design documents.

Table 2 explains the syllabus of this class. During the early part of the class, students will overview the outline of software engineering and agile development. From the seventh class, students start developing their own software based on the provided OSS. In this class, students modify the OSS, therefore analysis of existing software should be done first of all on "alter existing software," that is, Evaluation → Planning → Design → Coding → Evaluation is needed for such development.

Table 2 Syllabus for "Fundamentals of Practical Software Development"

No.	Contents	No.	Contents
1	Guidance	9	Coding 1 Programming, unit test
2	Introduction to software engineering 1 (Summary of software engineering, requirement)	10	Coding 2 Programming, unit test
3	Introduction to software engineering 2 (design, coding)	11	Coding 3 Programming, join test
4	Introduction to software engineering 3 (Test) Summary of agile development	12	Coding 4 Programming join test
5	Extreme programming (XP) 1 (Life cycle of XP, Concept of XP, Story card, Task card)	13	Evaluation (Validation and analysis of obtained software)
6	Extreme programming (XP) 2 (Pair programming, test-driven development)	14	Preparation of publishing software Documentation
7	Evaluation, Planning Extraction of story for target software	15	Presentation
8	Design Task division		

6. PRACTICAL STUDY OF INNOVATING METHODOLOGY TO AGILE DEVELOPMENT

In this section we describe an evaluation of applying innovative methodology to agile development based on the proposed EPISODE in our class. There are 20 students in our class, and they formed six groups (one "support soft for dyschromatopsia" group, three "calculator" groups, and two "breakout game with physical controller" groups).

Students develop their software based on following procedure: Evaluation → Planning → Design → Coding → Evaluation. In the first evaluation phase, students choose an OSS based on their request, then they analyze values of the chosen OSS (such as "Calculator") using a value graph. The theme of making value graph work is "xx application" (xx means target software).

Table 3 Results of value graph analysis

Group	Theme	Obtained value	Insight
1	Support soft for dyschromatopsia	Make dyschromatopsia person's life easier	Lack adequate support for designer to design in consideration of dyschromatopsia persons
2	Calculator	Obtain correct calculation result, everyone can obtain the same calculation result	Need for high-function calculator which can be used easily
3	Calculator	Calculate rapidly and easily; visualize calculation result	Need for calculator which can calculate complicated formulas and visualize mathematical formulas
4	Calculator	Make calculation easier with more precision	Want to shorten the time of calculation
5	breakout	Have fun, gain a feeling of accomplishment, to kill time	Incorporate elements of sports
6	breakout	Have fun, gain a feeling of accomplishment, to kill time	Get rid of stress, get an exhilarating feeling

From Table 3, each group could analyze essential qualities for their target software. Students could find higher layer values of target software, and obtain some insights from the obtained value graph.

In the next phase (planning phase), students did brainstorming based on the subject "When did you find the application useful?" Students then made affinity diagrams from the obtained ideas by brainstorming. They made groups in terms of "situation." Table 4 indicates the result of the brainstorming and affinity diagram.

Table 5 Results of brainstorming and affinity diagram

Group	Theme	Obtained group name (situation)	Keyword (partially)
1	Support soft for dyschromatopsia	In setting up	Allow detail settings, fast and easy access
		Lost	Ability to be used anywhere, can get some information
		Web browsing	Privacy mode, favorite, tag
2	Calculator	Pack light	Do not bring books, cloud service, substitutable for dictionaries
		To give shape	Listen to music, studying English words
		Use in train	Search train connections, Use twitter
		life-or-death matter	Get lost, expenditure memo
3	Calculator	Use in drinking party	Check final train, find coupons, find common topics
		Writing a report	Translate, get up on time, suggest a candidate character
		Use in play tag	Search current location, check forecast, chronograph
		In shopping	Manage banking on the Internet, dealings on the stock exchange, price survey
4	Calculator	Job hunting	Alarm clock, forecast, check news
		Escape from reality	SNS, exercise, listen to music
		Preparation	File sharing, take photos, studying
5	breakout	Live	Cook dinner, make schedule, check schedule, get up
		Something unexpected	Get home late, earthquake occurrence, have no money
		What one tries to do	Search shops, calculate something
6	breakout	Play a role in an emergency	Arrive at a destination, unable to make a quick decision
		Long for the days as a child	Find old photos, play old games
		No money at the end of the month	Use free tablet application, use free call application

From the result of the value graph analysis and brainstorming and affinity diagram, each group found the development goal for their software as shown in Table 6.

Table 6 Development goal for each group

Group	Theme	Development goal
1	Support soft for dyschromatopsia	To develop a function that supports designer
2	Calculator	To develop a calculator with graph plot function
3	Calculator	To develop a calculator which can calculate from history
4	Calculator	To develop a calculator with memo function
5	breakout	To develop a match type breakout
6	breakout	To develop a breakout game with fun by breaking chain

Students learned how to do brainstorming, making affinity diagrams, and generating value graphs by lecture before they analyzed their software. However, we saw that some students experience difficulty in using these methodologies. More effective guides for these methodologies are needed. We will improve our methods of support.

6. RELATED WORKS

So far, a great deal of research has been done on software development education.

Turnu et. al. [6] report the effects of agile practices on open source software development. They evaluated the effects of Test Driven Development (TDD). We also use TDD in EPISODE. TDD is a useful method for developing; therefore, we adopt TDD into our method.

Koch [7] reports the comparison of agile and open source software development. The study discusses differences between open source software and agile development. Our proposed method targets developing open source software based on agile development with the method of generating innovative ideas. Our method allows us to benefit from the merits of both open source development and agile development.

Aaen [8] proposes an agile development method. The method includes ways to facilitate creativity and innovation by using brainstorming. We also take in brainstorming for our proposed develop method. However we adopt not only brainstorming but also use a value graph. Our method targets developing software without customers. Therefore the value graph plays the important role of helping us discuss and judge whether something is innovative or not.

7. CONCLUSIONS

In this study, we propose a new method that targets the development of innovative software by small groups by using artificial intelligence (AI), idea creation, value engineering, and systems design. We find the correspondence between the cycle of XP and processes for innovating (divergence and convergence). Then, we propose a method for developing innovative software using the brainstorming and value graph methods. We also present a practical study of our proposed methodology.

We will develop the proposed tool and evaluate the effectiveness of our method through our class.

REFERENCES

[1] Alex F. Osborn, Applied Imagination. Creative Foundation Inc., 1963.
[2] Lou Cohen. Quality Function Deployment: How to Make QFD Work for You. Addison-Wesley, 1995.
[3] Kousuke Ishii, Kenji Iino, Sekkeinokagaku kachidukurisekkei. Yokendo, 2008, (in Japanese).
[4] “UEC Software Repository,” https://www.repository.uec.ac.jp/.
[5] IPA:INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN, “Report and reference guide for survey of application example on practice of agile development,” http://www.ipa.go.jp/sec/softwareengineering/reports/20130319.html, (in Japanese).
[6] Ivana Turnu, Marco Melis, Alessandra Cau, Alessio Setzu, Giulio Concas, Katiuscia Mannaro, “Modeling and simulation of open source development using an agile practice, ” Journal of Systems Architecture, Volume 52, Issue 11, November 2006, Pages 610-618.
[7] Stefan Koch, “Agile Principles and Open Source Software Development: A Theoretical and Empirical Discussion,” Extreme Programming and Agile Processes in Software Engineering, Lecture Notes in Computer Science, Vol. 3092, pp. 85-93, 2004
[8] van Aaen, “Essence: Facilitating Agile Innovation”, Agile Processes in Software Engineering and Extreme Programming, Lecture Notes in Business Information Processing, pp.1-10, 2008.

Authors

Takaaki Goto graduated with a Doctor of Engineering degree from Toyo University in 2009. He is a Project Assistant Professor at the Graduate School of Informatics and Engineering at The University of Electro-Communications. His main research interests are applications of open source software development, software development education, graph grammars, visual languages, and software development environments. He is a member of IEICE Japan, IPSJ, and IEEE.

Kensei Tsuchida received M.S. and D.S. degrees in mathematics from Waseda University in 1984 and 1994, respectively. He was a member of the Software Engineering Development Laboratory, NEC Corporation in 1984-1990. From 1990 to 1992, he was a Research Associate of the Department of Industrial Engineering and Management at Kanagawa University. In 1992 he joined Toyo University, where he was an Instructor until 1995 and an associate professor from 1995 to 2002, and a Professor from 2002 to 2009 at the Department of Information and Computer Sciences, and since 2009 he has been a Professor of Faculty of Information Sciences and Arts. He was a Visiting Associate Professor of the Department of Computer Science at Oregon State University from 1997 to 1998. His research interests include software visualization, human interface, graph languages, and graph algorithms. He is a member of IPSJ, IEICE Japan and IEEE Computer Society.

Tetsuro Nishino graduated from department of Mathematics, Waseda University, and continued his research and obtained a D.Sc. in Mathematics in 1991. He was a researcher at Tokyo Research Laboratory, IBM Japan during 1984-1987. He was an Associate Professor at School of Information Science, Japan Advanced Institute of Science and Technology, Hokuriku during 1992-1994. In 1994, he joined at Department of Communications and Systems Engineering, The University of Electro-Communications as an Associate Professor and in 2006 he became a Professor. He received The Funai Information Technology Prize (2003) and IBM Faculty Award (2008).