IEICE TRANS. INF. & SYST., VOL.E101–D, NO.7 JULY 2018

PAPER Special Section on Knowledge-Based Software Engineering

Usability Evaluation Method of Applications for Mobile Computers Using Operation Histories

Junko SHIROGANE[†a)], ***Member***, **Misaki MATSUZAWA**[††], ***Nonmember***, **Hajime IWATA**[†††], ***and*** **Yoshiaki FUKAZAWA**[††], ***Members***

SUMMARY Various applications have been realized on mobile computers such as smart phones and tablet computers. Because mobile computers have smaller monitors than conventional computers, strategies to develop user interfaces differ from conventional computer applications. For example, contents in a window are reduced or divided into multiple windows on mobile computers. To realize usable applications in this situation, usability evaluations are important. Although various usability evaluation methods for mobile computers have been proposed, few evaluate applications and identify problems automatically. Herein we propose a systematic usability evaluation method. In our method, operation histories by users are recorded and analyzed to identify steps with usability problems. Our method automatically analyzes usability problems, allowing usability evaluations in software development to be implemented easily and economically. As a case study, the operation histories were recorded and analyzed when 20 subjects operated an application on a tablet computer. Our method automatically identified many usability problems, confirming its effectiveness.

key words: *usability, evaluation, mobile, operation history*

1. Introduction

Mobile computers, such as smart phones and tablet computers, are ubiquitous. Because users interact with applications via graphical user interfaces (GUIs), usability of GUIs strongly impacts applications. Usability depends on users' preferences and experiences. Thus, iterations involving the development of GUIs, user evaluations, and improvement of GUIs are essential to realize highly usable applications [1].

Usability evaluation methods are classified into two main types: tests by subjects [2] and evaluations by usability specialists [3]. For tests by subjects, subjects use the target application, their operation histories are recorded, and their operations are observed. After operating the target applications, users answer interviews or questionnaires. Then the data from the recorded operation histories, observations, and answers are analyzed. In evaluations by usability specialists, experts search for usability problems from the target application and its documentation based on experience. They also assume the user characteristics of target users, operate the target applications in terms of the target users, and identify usability problems.

Various usability evaluation methods have been provided from conventional computer applications to mobile applications [4]–[6]. However, the GUI structure of mobile applications differs drastically from conventional computers because mobile computers have much smaller monitors. Hence, applying usability evaluation methods for conventional computer applications to mobile applications may be inappropriate.

Many evaluation methods for mobile applications are performed by usability specialists. For example, they observe users' operations, conduct interviews or questionnaires, and analyze the results [7]–[9]. This work is time consuming and very challenging for non-usability specialists. In an effort to realize an automatic evaluation, some methods have proposed recording the operation histories [10]–[12], but concrete usability problems are not identified. Thus, a usability evaluation method that can identify usability problems easily and systematically is required.

To resolve the aforementioned problems, we have developed a method to evaluate the usability of mobile applications and identify usability problems automatically [13]. In our method, developers, who are usability specialists, initially define tasks to evaluate usability. Then subjects operate mobile applications along with the defined tasks. During these operations, operation histories are recorded. Finally, the operation histories are analyzed. This analysis, which is based on the 10 usability heuristics proposed by Nielsen [14], identifies usability problems as steps (e.g., tapping a button defined by developers) with the elements of the 10 usability heuristics. Unsatisfied steps are determined. That is, our method shows steps with usability problems and the origins of the usability problems.

Previously our method was applied to a small sample application as a case study [13] but the effectiveness of our method was not evaluated. Herein we apply our method to another sample application and evaluate its effectiveness. This paper describes our entire method and the evaluation results.

The rest of the paper is organized as follows. Section 2 discusses related research on compares other research on usability evaluations for mobile devices. Section 3 describes the contributions of our usability evaluation method for mobile devices, while the concepts and techniques used by our method are summarized in Sect. 4. Section 5 describes the

Manuscript received November 8, 2017.
Manuscript revised March 13, 2018.
Manuscript publicized April 20, 2018.

[†]The author is with the Tokyo Woman's Christian University, Tokyo, 167–8585 Japan.
[††]The authors are with the Waseda University, Tokyo, 169–8555 Japan.
[†††]The author is with the Kanagawa Institute of Technology, Atsugi-shi, 243–0292 Japan.
a) E-mail: junko@lab.twcu.ac.jp

DOI: 10.1587/transinf.2017KBP0022

acquisition of the operation histories in our method, while detailed the usability evaluation strategy of our method is are shown in Sect. 6. Section 7 evaluates the proposed method. Finally, Sect. 8 concludes this paper.

2. Related Works

Various works have investigated usability evaluations of mobile applications. For automatic evaluations, Lettner et al. developed a toolkit to evaluate the usability of mobile applications [15]. Their toolkit analyzed the operation histories of users statistically using metrics for the number and rate of user errors. The evaluation results were shown by a diagram of the application's window switching. In this diagram, colors identified the severity of problems in the windows, allowing developers to visually recognize problems.

Ma et al. also proposed a method to evaluate the usability of mobile applications by comparing the operation sequences of non-experts with an expert [11]. First, their method obtained the operation histories of the target tasks by an expert. The expert's sequences in the histories were represented by state charts. Windows were states, and transitions were operations in the state chart. Next, the operation histories of non-experts were also obtained, and their sequences were traced on the state charts. Finally, the differences between expert and non-expert sequences such as transitions to states not depicted in the state chart were analyzed. Although both Lettner's and Ma's studies contributed to the reduction of time and costs for usability evaluations, the analysis targets were mainly user errors, and other factors such as a large time consumption were not targeted.

New heuristics for mobile applications have been proposed. Gomez et al. proposed heuristics for mobile applications for non-experts to evaluate usability [16]. First, they listed the characteristics of mobile devices. Second, existing highly abstract heuristics were rearranged into 13 heuristics considering the characteristics of mobile devices. Third, low abstract heuristics were classified into the 13 heuristics as subheuristics. Inappropriate subheuristics for mobile devices were excluded. Fourth, extra subheuristics were added considering mobile devices. Then the heuristics were formatted.

SMART (Smartphone Mobile Application heuRisTics) has also been proposed as heuristics for mobile applications [17], [18]. These heuristics were based on existing heuristics, specialists of HCI (Human Computer Interaction) evaluated them, and finally, 13 heuristics were developed. These heuristics were more suitable to evaluate the usability of mobile applications, but support to apply them was not considered.

Preparations of usability evaluations have also been supported. Ferre et al. proposed a method to record the operation histories of native mobile applications [19]. In their method, target evaluation tasks were specified by use case descriptions. Annotations to record user interface elements were added to the use case descriptions. Operation histories were recorded based on GAMA (Google Analytics for Mobile Applications) [20] using the annotated use case descriptions. Then a usability expert analyzed the recorded operation histories and evaluated usability.

SwiftHand has been used to generate test input sequences for Android applications [10]. This learned models of the target application employing machine learning techniques. Then it used the learned models to generate user input that transitioned to unexplored states. The application was executed to refine models. These methods reduced the preparation of the usability evaluations, but concrete analysis of the obtained data was not supported.

In addition, usability evaluations by questionnaires, observations, and video analyses have been performed. Moumane et al. evaluated some mobile applications empirically [21]. The evaluation criteria, which were used for objective measures, were usability characteristics of ISO/IEC 9126 standards [22], and ISO/IEC 25062 [23] and ISO 9241 [24] standards. To evaluate the satisfaction, QUIS 7.0 (Questionnaire for User Satisfaction Interaction) [25] was used. Dunn et al. evaluated various tasks available in smart phones in terms of importance and frequency of use [26]. First, users ranked important tasks. Next, the listed tasks were limited using the Delphi method [27]. Then the importance and frequency of tasks were analyzed, showing the relationships. Although these evaluations reported the results for existing applications and tasks, automatic evaluations were insufficient.

3. Features of Our Method

The features of our method are as follows:

Automatic identification of usability problems

Various usability evaluation methods exist, but many are implemented by usability specialists. Usability evaluations are burdensome and expensive because developers must recruit subjects to use the target application and then developers must analyze the subjects' operations and opinions. In addition, developers who are not usability specialists have difficulty executing usability evaluations. Consequently, a method to efficiently perform usability evaluations would be extremely useful.

In our method, users' operation histories are recorded and analyzed to determine usability problems. Our method is automatic, reducing the burden of analysis while improving ease of use. Thus, non-usability specialists can easily analyze usability problems.

Clarification of steps with usability problems

To improve GUIs, detailed steps with usability problems must be analyzed. Although existing methods record operation histories in an effort to realize an automatic usability evaluation, many analyze only window switching by user operations. However, window structures (layout and arrangement of widgets) are diverse, and window switching may be insufficient to identify detailed usability problems. It is necessary to record and analyze detailed operations of widgets in windows.

Our method records and analyzes detailed operations of widgets such as taps, swipes, and flicks using statistic methods. Usability problems are identified by steps, allowing developers to recognize which steps of GUIs require improvement.

Clarification of origins of usability problems

To improve GUIs appropriately, the origin of a usability problem must be understood. Our method analyzes operation histories in terms of the 10 usability heuristics by Nielsen [14], providing not only usability problems but also their origins.

4. Preliminary

4.1 10 Usability Heuristics

This section defines the 10 usability heuristics and the criteria of the usability evaluation in our method.

4.1.1 Definition

The 10 usability heuristics by Nielsen [14] are basic strategies that can be used to define the criteria of usability evaluations. Concrete elements of the heuristics are:

Heuristics-1 Visibility of system status
The system should always provide information about the user's status.

Heuristics-2 Match between system and the real world
The system should use terminologies, words, and concepts familiar to users.

Heuristics-3 User control and freedom
The system should support undo and redo to avoid system behaviors by mistake.

Heuristics-4 Consistency and standards
The system should maintain consistency of words, situations, and actions.

Heuristics-5 Error prevention
The system should avoid reoccurring mistakes rather than displaying the appropriate error message.

Heuristics-6 Recognition rather than recall
The system should make information visible and not require users to remember information.

Heuristics-7 Flexibility and efficiency of use
The system should allow users to adjust frequently used functions.

Heuristics-8 Aesthetic and minimalist design
The system should not show irrelevant or unnecessary information.

Heuristics-9 Help users recognize, diagnose, and recover from errors
The system should explain error messages with understandable words, present problems appropriately, and suggest solutions.

Heuristics-10 Help and documentation
The system should provide simple tutorials and documentations that users can easily search and understand.

4.1.2 Criteria for usability evaluation in our method

Usability definitions include several elements such as efficiency, effectiveness, and satisfaction [28]. Automatic usability evaluations can be performed using objective data. Typical usability elements that can be evaluated by objective data from users are efficiency and effectiveness [29]. Efficiency includes whether users can achieve their purpose in a short time with a few resources, while effectiveness includes the degrees of accurately and correctly achieving users' purposes and error rates. Considering the operations of mobile computers, we define five evaluation criteria by referring to the literature [30], [31] to evaluate the efficiency and effectiveness by analyzing the operation histories. Usability problems are operations that have statistically significant differences between the operation histories of novice and expert users. The criteria and their correspondences to efficiency and effectiveness are:

Operation time (efficiency) Time that a user operates a step

Number of single tap selections (efficiency) Number of single taps to select GUI widgets

Scroll length (efficiency) Time that a user scrolls

Number of unnecessary double taps (effectiveness) Number of double taps unrelated to an application operation

Use of back button (effectiveness) Number of times that a back button is used by the mobile computer (not the application)

Although other operations are also available in mobile computers such as flick and pinch in/out, they are treated as kinds of single taps and scrolls. Thus, our method evaluates usability with these five evaluation criteria.

When a usability problem is identified, the reason must be clarified to resolve it. We consider that the 10 usability heuristics are sufficient to identify the origin of a usability problem. Thus, the five evaluation criteria correspond to the 10 usability heuristics [30]–[32] (Table 1). Table 1 clarifies the reason of the usability problems, allowing GUI improvement strategies to be easily determined. In addition, when a step has usability problems in terms of the evaluation criteria, overlapping heuristics are considered to be more critical reasons for the usability problems.

Considering the evaluations of efficiency and effectiveness, heuristics-1, 2, 4, 6, 9, and 10 are elements of application understandability. Thus, they contribute to both efficiency and effectiveness. Heuristics-5 requires the adoption of strategies to prevent errors. Hence, it contributes to both efficiency and effectiveness. Heuristics-7 and 8 require operation flexibility and reduction of useless content. Therefore, they contribute to efficiency. However, heuristics-3 requires support to recover from errors. Because efficiency requires reducing the operation time and resources while effectiveness requires reducing the number of errors, heuristice-3 does not correspond to the evaluation criteria for either effi-

ciency or effectiveness.

How to resolve usability problems employing only these heuristics is difficult. Thus, concrete and possible problems are associated with these heuristics by referring to [30]–[32]. These assist developers in identifying a path to resolve usability problems. Table 2 shows examples of concrete and possible problems for the "scroll length" criterion. It should be noted that for our current target (Android as described in Sect. 5.2) we use references [30], [31] to define evaluation criteria, the correspondences between the evaluation criteria and heuristics, and the correspondences between heuristics and concrete and possible problems. Although these references are described for iPad, we don't consider to adjust their descriptions for Android because the scope of our method is not limited to Android. Actually, their descriptions do not have to be adjusted in the current scope of our method when we implement our system.

4.2 Statistical Analyses in Our Method

The data from the usability evaluation are analyzed statistically. Our method employs the Shapiro-Wilk test [33], parametric test [34], and nonparametric test [35] for statistical analyses. The Shapiro-Wilk test is used to analyze whether the obtained data are normal distributions.

A parametric test is used for data with a certain distribution. Although this test has various analysis methods, we employ the F test [36] and the T test [37]. These tests require that the data to be analyzed follow normal distributions. The F test is used to analyze whether the populations of two types data are the same. That is, the variance (equal or unequal) for the two types of data is analyzed. The F test is used before T test. The T test is used to determine whether the average between the populations of two types of data differs. If the populations have equal variance, student's t-test [38] is used. If populations have unequal variance, Welch's t-test [39] is used.

A nonparametric test is used for data that do not satisfy a certain distribution. This test can be used for a small dataset or for an unspecified distribution type. Compared to the parametric test, it is difficult to demonstrate a significant difference using the nonparametric test. Consequently, our method employs the Wilcoxon rank sum test [40], which is a type of non-parametric test. It determines whether there is a difference between the medians of the two types of data when the data are not normal distributions. The results are similar to the T test.

Table 1 Correspondences between criteria and the 10 usability heuristics [13]

Evaluation criteria	Elements of 10 usability heuristics
Operation time	Heuristics-2 Heuristics-5 Heuristics-6
Number of single tap selection	Heuristics-1 Heuristics-2 Heuristics-4 Heuristics-5 Heuristics-7 Heuristics-8
Scroll Length	Heuristics-2 Heuristics-6 Heuristics-10
Unnecessary double taps	Heuristics-1 Heuristics-2 Heuristics-4 Heuristics-6 Heuristics-9
Use of back button	Heuristics-4 Heuristics-5 Heuristics-9

Table 2 Examples of problems corresponding to heuristics (Scroll length)

Heuristics	Concrete and possible problems
Heuristics-2	• Unclear words and difficult to understand • Object intentions not intuitive • Existence of uncommon words • Existence of atypical operations
Heuristics-6	• Selectability of objects unclear • Object intentions not intuitive • Window operability unclear • Operations too diverse • Inappropriate size of texts and objects
Heuristics-10	• Lack of tutorials and documentations • Complex application structure • Difficult to understand tutorials and documentations

5. Acquisition of Operation Histories

5.1 Strategies to Record Operation Histories

To evaluate the usability of mobile applications, operation histories must be recorded. Figure 1 shows a method to record operation histories.

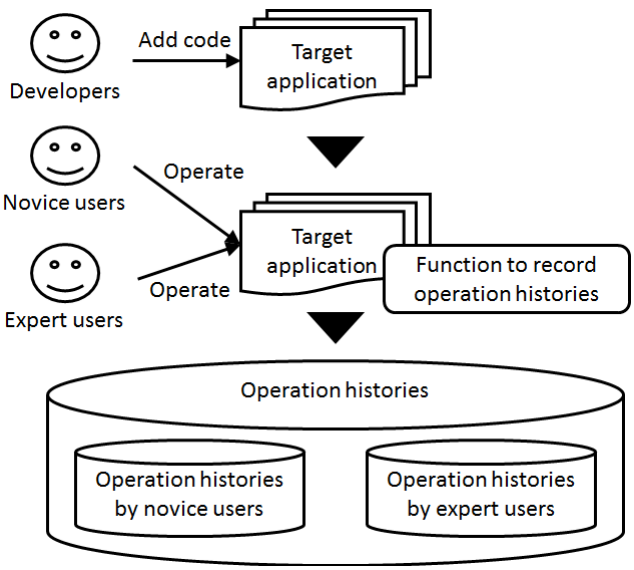

Fig. 1 Operation history acquisition

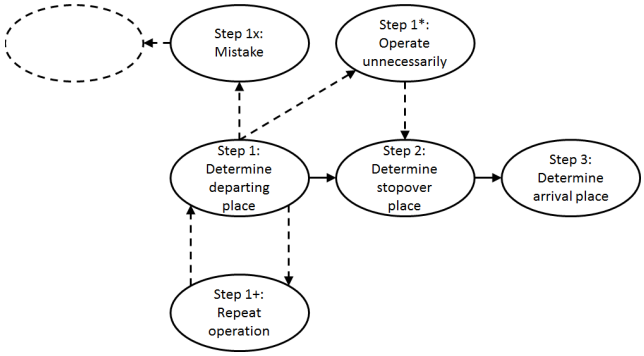

Fig. 2 Example of a task (revised from [13])

5.1.1 Preparation of Recording Operation Histories

To obtain operation histories, a function to record the operation histories must be added to the target application. Users' operations to the GUIs and application reactions from the GUIs by the operations are recorded as operation histories. The concrete items of operation histories are as follows:

- Target widget of an operation
- Operation type
- Window switching
- Time of operation or window switching

5.1.2 Definition of Tasks

Developers define tasks for usability evaluations. We assume these developers are usability specialists. In our method, a use case is considered as the task. A task is defined as operations to achieve a certain purpose after starting the target application. A task consists of some steps. A step is an operation by a user or a reaction to/from the target applications such as tapping a button, scrolling, or switching windows. Developers prepare some tasks to evaluate usability and define steps by dividing a task. Then, developers operate the target application and specify desirable steps to complete a task using the operation histories because usability problems may occur if users perform unnecessary operations. If a user operates different steps from the specified steps, the operations are detected as inappropriate steps by our method.

Figure 2 shows an example of a task in a map application. A developer can set steps to specify the departure, stopover, and arrival. The application shows how to proceed. In this example, a task is defined as three steps: determine a departure location (step 1), determine a stopover place (step 2), and determine an arrival place (step 3). In this figure, a solid oval indicates a step. A broken oval indicates an unnecessary step to perform the task and the only way to return to an appropriate step is to use the back command. Solid lines indicate appropriate flows from a step to a step in the task, while broken lines indicate unnecessary step flows by a user's mistake. Thus, step 1x indicates a mistake and a user cannot finish the task successfully. Step 1+ and step 1* indicate that the user can finish the task successfully. However, the user repeats unnecessary operations in step 1+ and performs an unnecessary operation between step 1 and step 2. In case that the user performs step 1x, 1+, or 1*, usability problems are identified.

This figure shows only an image to represent the operation histories and examples of appropriate/inappropriate step flows. Our method does not use this figure to set tasks and steps.

5.1.3 Record of Operation Histories

Users operate the target application with a function to record operation histories along with the defined tasks. Our method considers two types of users: novice and expert users. Operations by novice users are compared with those of expert users. This technique is common in usability evaluations [41].

Novice users are the target users of the target application. Expert users are familiar with the operations of the target application and possess significant knowledge about the target application. Examples of expert users are evaluators and developers. If the results differ significantly between the two groups, the operation has a usability problem.

5.2 Implementation

Currently, our method supports Android applications. Two types of development methods for GUIs are prepared: XML-based and code-based. For XML-based, the GUI program is written in XML. Widgets are identified by resource IDs. For code-based, the GUI program is written in Java. The current implementation target is XML-based. However, analysis of the operation histories is independent of the operating system and programming languages. That is, by recording the operation histories, usability evaluations of applications on various operating systems can be performed.

Our method uses LogCat [42] because it can record operation histories, application reactions, and user's operations. The following items can be recorded by LogCat: single tap, double tap, scroll, flick, long tap, long hold, use of back button, resource ID, window switching, and time.

These items must be recorded along with the operated widgets. Because the current implementation target of our method is XML-based, widgets are identified by their resource IDs. That is, user operations must be recorded along with the resource IDs of the operated widgets. To record these operation histories by LogCat, developers must add additional code to application programs.

6. Usability Evaluation

The obtained operation histories are analyzed to identify usability problems. Figure 3 shows the flow of the usability evaluation in our method.

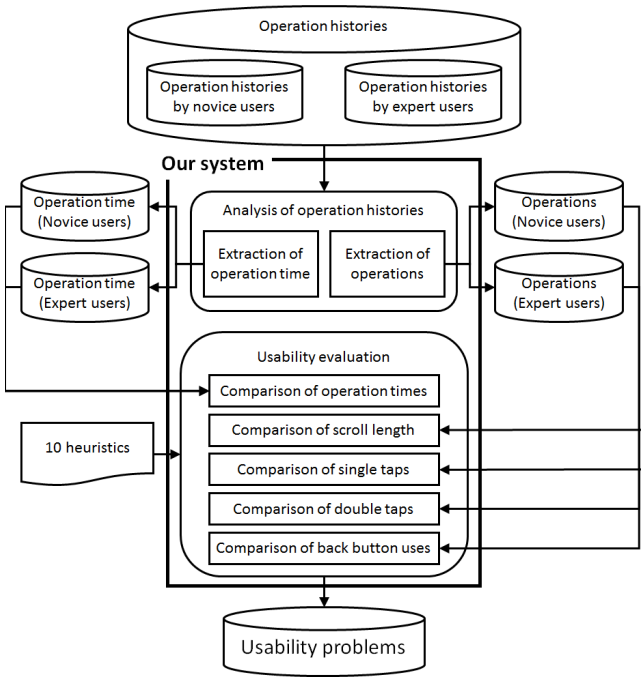

Fig. 3 Flow of a usability evaluation

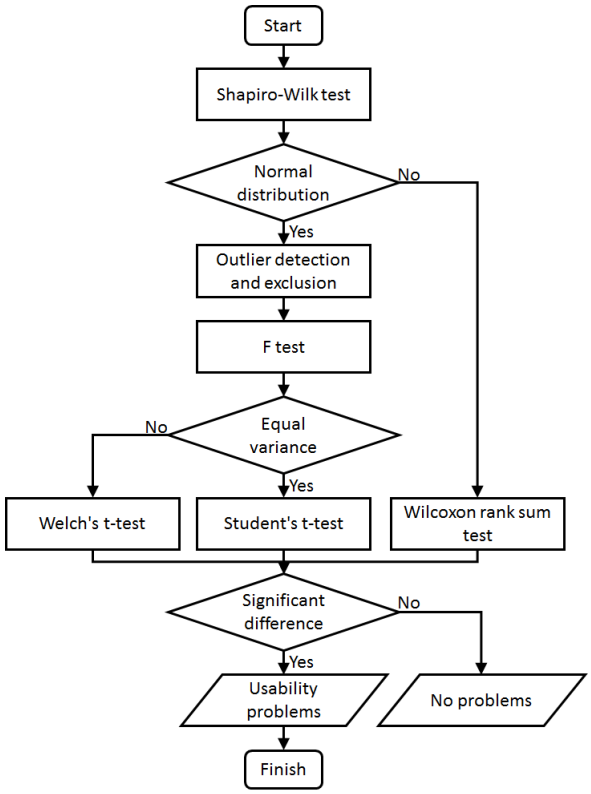

Fig. 4 Flow of statistical analysis (revised from [13])

6.1 Extraction of Time and Operations

As described in Sect. 4.1.2, the usability evaluation is performed in terms of five criteria: operation time, number of single taps, scroll length, number of unnecessary double taps, and use of back buttons.

A task consists of some steps. First, the operation histories are divided into steps. Then the operation time of each step is calculated as the difference between the time of the first operation and the time of the last operation in a step.

Second, steps are analyzed in terms of operation type (single tap, double tap, scroll, etc.) and resource ID. For "number of single taps", "number of unnecessary double taps", or "use of back buttons", the number of entries with single taps, double taps, and back buttons in a step is counted, respectively. The "scroll length" is calculated based on the number of scroll, drag, and flick entries in a step. The operation history records the number of scroll occurrences but not the scroll length. Meanwhile, when a user traces a monitor with his/her finger over 20 ms, a scroll entry is recorded. Thus, scroll length is calculated by multiplying the number of scroll entries by 20 ms.

6.2 Statistical Analysis

For the extracted individual data for steps, usability problems are identified by comparing novice user data with expert user data statistically. Figure 4 shows the flow of this statistical analysis.

First, the Shapiro-Wilk test verifies whether the novice user data and expert user data follow a normal distribution. For a normal distribution, the presence of outliers is also determined. Outliers are excluded because they may strongly influence the final results. To analyze outliers, the average (μ) and standard deviation (σ) are used [43]. If a data point is less than $\mu - 2\sigma$ or larger than $\mu + 2\sigma$, it is identified as an outlier. This outlier analysis has a confidence interval of 95%.

Second, if novice user data and expert user data are normally distributed, the F test determines the variance [36]. For equal variance and unequal variance, student's t-test [38] and Welch's t-test [39] are applied, respectively. Meanwhile, if novice user data or expert user data are not normally distributed, the Wilcoxon rank sum test [40] is applied. Consequently, our method can analyze whether novice user data and expert user data significantly differ.

6.3 Presentation of Results

Finally, the statistical analysis reveals steps with usability problems according to the evaluation criteria. Evaluation criteria where novice user data and expert user data significantly differ at the 95% confidence interval are identified as usability problems. Usability evaluations in our method are performed by step in terms of each evaluation criterion. Thus, evaluation criteria that a step has usability problems are identified. Heuristics corresponded to evaluation criteria are identified based on Table 1. Concrete and possible problems to each heuristic are associated previously by referring to [30]–[32] as described in Sect. 4.1.2, thus concrete and

possible problems for the target heuristics are determined based on the associations. Table 2 shows examples of usability problems for a certain step in terms of the evaluation criterion "scroll length" identified by this process.

In addition, the numerical values of the statistical analysis results can be shown. This allows developers to recognize the evaluation criteria with more critical usability problems.

7. Evaluation

To confirm whether our method can appropriately and efficiently identify usability problems, we performed a usability evaluation using a mobile application.

7.1 Appropriateness of Our Method

7.1.1 Usability Evaluation Performance

The target application was household accounts that we developed. This application consists of a date display window (the first window when this application was started), entry addition window, date selection window, and monthly entry display window. We prepared this application with a function to the record operation histories.

Twenty subjects participated in the usability evaluations. Eight were expert users, and twelve were novice users. Expert users were familiar with the application structure and operations. All of the subjects were university students who were familiar with the operations of mobile applications such as single tap and scroll.

We prepared eight tasks (Table 3). Each task consisted of two steps. Tasks 5, 6, and 7 were similar to 1, 2, and 3, respectively. Subjects performed these tasks on an Android tablet computer.

In the evaluation, the subjects performed the eight tasks using the target application. The operation histories were recorded and analyzed. In addition, the novice subjects answered questionnaires about steps with usability problems and the origin of the problems. These questionnaires were not included in our method. To evaluate our method, we compare the usability problems by our method to those by the questionnaires and discuss the differences.

7.1.2 Usability Evaluation Results

Table 4 shows the steps identified by our method with usability problems and their correspondence to the 10 usability heuristics by Nielsen [14], while Table 5 shows the steps identified by subjects' questionnaires with usability problems, the reasons for the problems, and the corresponding 10 usability heuristics. As described in Sect. 7.1.1, each task consists of two steps. Steps 1–16 in these tables indicate the steps in the task. Comparing these two tables, the usability problems of steps 8, 10, 11, 12, 14, 15, and 16 were found by only subjects' questionnaires and not by our method.

Table 3 Tasks of usability evaluation

Task	Purpose
Task 1	Add an entry to a specific date
Task 2	Confirm whether an entry is appropriately added by date display mode
Task 3	Confirm whether an entry is appropriately added by month display mode
Task 4	Delete a specific entry by the date display mode
Task 5	Add an entry to another specific date
Task 6	Confirm whether an entry is appropriately added by date display mode
Task 7	Confirm whether an entry is appropriately added by month display mode
Task 8	Delete a specific entry by the month display mode

Table 4 Identified usability problems by our method

Step	Heuristics by Nielsen
Step 1 (Task 1)	Heuristics-1, 2, 4, 5, 7, 8
Step 2 (Task 1)	Heuristics-2, 6, 10
Step 3 (Task 2)	Heuristics-2, 5, 6
Step 4 (Task 2)	Heuristics-2, 5, 6
Step 5 (Task 3)	Heuristics-1, 2, 4, 5, 6, 7, 8, 10
Step 6 (Task 3)	Heuristics-1, 2, 4, 5, 6, 7, 8
Step 7 (Task 4)	Heuristics-1, 2, 4, 5, 7, 8
Step 9 (Task 5)	Heuristics-2, 6, 10
Step 13 (Task 7)	Heuristics-2, 6, 10

7.2 Efficiency of Our Method

Additionally, we evaluated the efficiency of our method. Usability evaluation methods can be roughly classified into usability tests and heuristic evaluations [1]. Usability tests indicate evaluations by the subjects. Subjects operated the target application. The operations and the subjects' aspects were recorded. Then the subjects answered the questionnaires. The recorded data and questionnaire responses were analyzed. Heuristic evaluations were performed by usability specialists, who identified usability problems based on their experiences and the guidelines. Thus, our method is included in usability tests in this classification.

Usability tests consist of test plan creation, test preparation, test performance, and test result analysis [1], [44]. Test plan creation includes determining the purpose of the usability evaluation and defining tasks. Test preparation includes arranging the test environment such as a room, test device, required applications, and subject recruitment. Test performance includes subjects operating the target application while recording the operations and subjects' aspects as well as completing questionnaires. Test result analyses include analyzing the recorded data and questionnaire responses to identify the usability problems. Table 6 compares our method to existing usability evaluation methods. In this table, "$\surd$" indicates the phases that developers must conduct.

7.3 Discussion

As described in Sect. 7.1.2, Table 4 shows the usability problems found by our method, while Table 5 shows the usabil-

Table 5 Identified usability problems by subjects' questionnaires

Step	Subjects' opinions	Heuristics by Nielsen
Step 1 (Task 1)	• Difficult to understand GUI objects intuitively • Difficult to distinguish designs and buttons • Unnecessary GUI objects	Heuristics-1, 2, 4, 5, 6, 8
Step 2 (Task 1)	• Small size of input items	Heuristics-2, 5, 6, 10
Step 3 (Task 2)	• Difficult to distinguish designs from buttons	Heuristics-1, 2, 5, 6
Step 4 (Task 2)	• Inconsistent operations • Difficult to understand operation situations • Lack of tutorials and manuals	Heuristics-1, 2, 4, 5, 6, 9, 10
Step 5 (Task 3)	• Difficult to understand operation situations • Lack of tutorials and manuals • Difficult to understand sentences • Unnecessary GUI objects • Difficult to understand display change	Heuristics-1, 2, 3, 4, 5, 6, 8, 9, 10
Step 6 (Task 3)	• Too many sentences • Difficult to understand scroll availability	Heuristics-1, 2, 4, 5, 6, 8
Step 7 (Task 4)	• Difficult to understand operation situations • Difficult to understand scroll availability	Heuristics-1, 2, 4, 5, 6, 8
Step 8 (Task 4)	• Inconsistent operations • Too many sentences	Heuristics-3, 5, 8
Step 9 (Task 5)	• Difficult to understand GUI objects intuitively • Unnecessary GUI objects • Difficult to distinguish designs from buttons	Heuristics-1, 2, 5, 6
Step 10 (Task 5)	• Small size of input items	Heuristics-4, 6
Step 12 (Task 6)	• Difficult to understand operation situations	Heuristics-1, 5, 6
Step 13 (Task 7)	• Difficult to understand sentences	Heuristics-1, 2, 5, 6, 8
Step 14 (Task 7)	• Difficult to understand scroll availability	Heuristics-1
Step 15 (Task 8)	• Difficult to understand scroll availability	Heuristics-1
Step 16 (Task 8)	• Inconsistent operations	Heuristics-4, 5

Table 6 Phase comparison of our method with existing methods

	Our method	Existing usability evaluations
Test plan creation	√	√
Test preparation	√	√
Test performance	√	√
Test result analysis		√

Table 7 Number of identified usability problems

	Number of identified problems (All steps)	Number of identified problems (steps without 9–15)
Our method	42 (6)	36 (4)
Questionnaire	63 (27)	47 (15)
Total	69	51

ity problems found by subjects' questionnaires. Because method to find usability problems were different, the found usability problems were also different. In Table 4, heuristics were corresponded systematically to evaluation criteria based on Table 1, and steps had usability problems in terms of evaluation criteria. Thus, heuristics were corresponded to steps as usability problems systematically. Meanwhile, usability problems in Table 5 were identified based on subjects' questionnaires. In the questionnaires, subjects answered steps with usability problems, targets of usability problems, and detailed problems. Heuristics were corresponded to steps based on the analysis of the answers, and these correspondences were not systematic. Because the usability problems in Table 5 were derived directly from subjects' opinions, they are considered to represent more appropriate usability problems. Below we discuss the differences between these two methods.

First, steps with usability problems by our method were compared to the questionnaires. Our method reveals usability problems for steps 1, 2, 3, 4, 5, 6, 7, 9, and 13 (Table 4), while the questionnaires reveals usability problems in steps 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 12, 13, 14, 15, and 16 (Table 5). All the steps identified by our method are included in the steps identified by the questionnaires. Thus, our method can identify appropriate steps with usability problems to be modified.

Next, comparing the number of identified usability problems by our method to the questionnaire results assessed the effectiveness of our method to identify usability problems. For the 16 steps, steps 9–15 are similar to steps 1–7. Because steps 9–15 were performed after steps 1–7, it is possible that subjects were familiar with steps 9–15 prior to performing them. Thus, we counted the number of usability problems (heuristics) in terms of all steps and the steps without 9–15. Table 7 shows the results, where the number in parenthesis indicates identified problems by only the corresponding method (i.e., "Our method" problems are identified by only our method, while "Questionnaire" problems are identified by only the questionnaires).

Of the usability problems found by our method, 36 problems in all steps and 32 problems in steps without 9–15 are the same problems as the questionnaires. That is, our method found 86% of the usability problems in all steps and 89% of usability problems in steps without 9–15 that should be modified assuming that the questionnaires revealed usability problems that should be modified. Thus, our method

can identify usability problems appropriately. The reason why the problems in steps without 9–15 have a higher percentage than problems in all steps is attributed to that the fact that the subjects were familiar with steps 9–15 prior to executing them. The results confirm that our method can identify usability problems for different operations.

However, our method did not identify 39% of the usability problems in all steps and 29% in steps without 9–15 in "Total". Because our method did not identify these usability problems, we hypothesize that these problems did not greatly impact the subject's ability to perform the steps. As described in Table 5, most of the usability problems by the subjects are difficulty understanding and distinguishing texts and objects. To identify these problems, the evaluation criteria must be increased and the operation histories must be analyzed in more detail.

In our method, usability problems correspond to the 10 usability heuristics by Nielsen [14]. Unnecessary heuristics are identified as the origin of the usability problems. For example, heuristics-7 (Flexibility and efficiency of use) is observed in steps 1, 5, 6, and 7. Because all subjects performed the same steps by the same operations, this heuristic is inappropriate. To address this problem, the correspondences between the evaluation criteria and the 10 usability heuristics should be customized by considering the characteristics of the target application and usability evaluations.

As described in Sect. 6.3, every evaluation criterion indicates heuristics that are not satisfied in a step. The types of usability problems such as operation types (single tap, double tap, or use of back button) and time (operation time or scroll time) are clarified by the evaluation criteria, while parts of tasks with usability problems are clarified by every step. Heuristics summarize the origins of usability problems. Detailed origins of usability problems that the heuristics are not satisfied are shown by concrete and possible problems similar to Table 2. Concrete and possible problems provide sufficiently detailed origins for developers to understand such as "Unclear words and difficult to understand" and "Inappropriate size of texts and objects". In addition, because the numerical values of the statistical analysis results can be shown, evaluation criteria that denote more critical usability problems can be clarified for a step. Due to this, heuristics that are not significantly satisfied are clarified, and concrete and possible problems with high priorities can be determined.

Meanwhile, if only heuristics that are not satisfied are shown, it may be difficult for developers to determine appropriate strategies to modify the target application GUIs, because one step has many applicable heuristics. However, prioritizing the evaluation criteria can resolve these problems.

In terms of usability evaluation efficiency, our method can eliminate the time and burden of the test result analysis phase, according to Table 6. Although our method requires steps for our system to be set in test preparation phase, and this is not performed in existing usability evaluation methods, if tasks and steps are already determined, the settings can be completed by only operating our system. Tasks and steps must be defined in not only our method but also existing usability evaluation methods. Thus, our method can reduce time to test result analysis phase.

Our method appropriately identified most of the steps and usability problems, our method can eliminate the burdens of the test analysis phase of usability evaluations. Although our method needs to be further refined, this study confirms that it can effectively identify usability problems and efficiently perform usability evaluations.

7.4 Threats to Validity

As described in 7.1.1, the experiment was performed on an Android tablet computer. Because tablet computers normally have larger monitors than smart phones, operations of tablet computers may differ from those of smart phones. For example, fat finger problems [45] are more likely to occur on smart phones than tablet computers. Thus, repeating the experiment using smart phones may produce different results.

All subjects of the experiments were university students familiar with operations of mobile computers. However, usability depends on users' preferences and experiences. If subjects have different preferences and experiences, the identified usability problems may differ. It is possible that our method detects different types of usability problems as the demographics vary.

In addition, we used 95% as the confidence interval for statistical analysis. Although 95% is a commonly used value, if another value is used as the confidence interval, the identified usability problems may not be the same as this experiment.

8. Conclusion

Although several studies have proposed usability evaluation methods for mobile computers, they are time consuming and burdensome. The GUI structures of mobile computers differ drastically from conventional computers due to the smaller monitor size. Consequently, usability evaluation methods for conventional computers cannot be directly applied to mobile computers. Herein we propose a method to evaluate usability of mobile computers.

In our method, the operation histories of subjects are recorded in usability evaluations. The subjects are classified as novice or expert users. Then the operation histories are analyzed statistically between the two groups. Large variations between novice and expert users indicate usability problems. Usability problems are identified by the criteria corresponding to the 10 usability heuristics by Nielsen [14]. Our method automatically analyzes the data obtained in usability evaluations. Thus, usability evaluations are performed easily and effectively even by non-usability specialists in terms of time and costs.

We performed an experiment to evaluate our method. During the experiment, 20 subjects operated a mobile ap-

plication and their operation histories were analyzed. These results were compared to the usability problems identified by questionnaires. Although our method did not identify usability problems exhaustively, our method automatically found most of the usability problems commonly identified via a questionnaire. Thus, the effectiveness of our method is confirmed.

Future work aims to improve our method based on experiments. Improvements include:

- Increase the evaluation criteria and improve operation history analysis
- Realize a customizable correspondence between the 10 usability heuristics and the evaluation criteria
- Develop usability evaluations for mobile applications except the native application

References

[1] J. Nielsen, "Usability Engineering," Morgan Kaufmann, 1994.
[2] C.M. Barnum and S. Dragga, "Usability Testing and Research," Pearson Education, 2001.
[3] J. Nielsen, "Usability Inspection Methods," John Wiley & Sons Inc., 1994.
[4] A. Fernandez, E. Insfran, and S. Abrahão, "Usability evaluation methods for the web: a systematic mapping study," Information and Software Technology, vol.53, no.8, pp.789–817, 2011.
[5] T. Carta, F. Paternò, and V.F. de Santana, "Web usability probe: a tool for supporting remote usability evaluation of web sites," Procs. of 13th IFIP TC.13 International Conference on Human-Computer Interaction (INTERACT 2011), vol.6949, pp.349–357, 2011.
[6] P. Palanque, E. Barboni, C. Martinie, D. Navarre, and M. Winckle, "A model-based approach for supporting engineering usability evaluation of interaction techniques," Procs. of the 3rd ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS'11), pp.21–30, 2011.
[7] B. Biel, T. Grill, and V. Gruhna, "Exploring the benefits of the combination of a software architecture analysis and a usability evaluation of a mobile application," The Journal of Systems and Software, vol.83, no.11, pp.2031–2044, 2010.
[8] A.L. Salgado and A.P. Freire, "Heuristic evaluation of mobile usability: a mapping study," Procs. of The 16th International Conference on Human-Computer Interaction (HCI International 2014), vol.8512, pp.178–188, 2014.
[9] R. Inostroza, C. Rusu, S. Roncagliolo, and V. Rusu, "Usability heuristics for touchscreen-based mobile devices: update," Procs. of the 2013 Chilean Conference on Human - Computer Interaction (ChileCHI'13), pp.24–29, 2013.
[10] W. Choi, G. Necula, and K. Sen, "Guided GUI testing of android apps with minimal Restart and approximate learning," Procs. of the 2013 ACM SIGPLAN International Conference on Object Oriented Programming Systems Languages & Applications, (OOPSLA'13), vol.48, no.10, pp.623–640, 2013.
[11] X. Ma, B. Yan, G. Chen, C. Zhang, K. Huang, J. Drury, and L. Wang, "Design and implementation of a toolkit for usability testing of mobile apps," Mobile Networks and Applications, vol.18, no.1, pp.81–97, 2013.
[12] B.K. Dunn, D.F. Galletta, D. Hypolite, A. Puri, and S. Raghuwanshi, "Development of smart phone usability benchmarking tasks," Procs. of 2013 46th Hawaii International Conference on System Sciences (HICSS 2013), 2013.
[13] M. Matsuzawa, H. Iwata, J. Shirogane, and Y. Fukazawa, "Support method of usability evaluations for android applications based on operation histories," Procs. of The Fifth International Conference on Digital Information Processing and Communications (ICDIPC 2015), 2015.
[14] J. Nielsen, "10 usability heuristics for User interface design," https://www.nngroup.com/articles/ten-usability-heuristics/, 1995 (Accessed 4 November, 2017).
[15] F. Lettner and C. Holzmann, "Automated and unsupervised user interaction logging as basis for usability evaluation of mobile applications," Procs. of the 10th International Conference on Advances in Mobile Computing & Multimedia (MoMM'12), pp.118–127, 2012.
[16] R.Y. Gómez, D.C. Caballero, and J.-L. Sevillano, "Heuristic evaluation on mobile interfaces: a new checklist," The Scientific World Journal, vol.2014, pp.1–19, 2014.
[17] G. Joyce, M. Lilley, T. Barker, and A. Jefferies, "Mobile application usability: heuristic evaluation and evaluation of heuristics," Procs. of 8th International Conference on Applied Human Factors and Ergonomics (AHFE 2016), vol.492, pp.77–86, 2016.
[18] G. Joyce and M. Lilley, "Towards the development of usability heuristics for native smartphone mobile applications," Procs. of Third International Conference on Design, User Experience and Usability, Held as Part of the HCI International 2014 (DUXU 2014), vol.8517, pp.465–474, 2014.
[19] X. Ferre, E. Villalba, H. Julio, and H. Zhu, "Extending mobile app analytics for usability test logging," Procs. of The 16th IFIP TC.13 International Conference on Human-Computer Interaction (INTERACT 2017), vol.10515, pp.114–131, 2017.
[20] "Google analytics for mobile apps," https://developers.google.com/analytics/solutions/mobile (Accessed on 6th Nov. 2017).
[21] K. Moumane, A. Idri, and A. Abran, "Usability evaluation of mobile applications using ISO 9241 and ISO 25062 standards," SpringerPlus 2016, 5:548, 2016.
[22] ISO/IEC 9126-1:2001, "Software engineering – Product quality – Part 1: Quality model," 2001.
[23] ISO/IEC 25062:2006, "Software engineering – Software product Quality Requirements and Evaluation (SQuaRE) – Common Industry Format (CIF) for usability test reports," 2006.
[24] ISO 9241-11:1998, "Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability," 1998.
[25] A. Hussain and M. Kutar, "Usability evaluation of SatNav application on mobile phone using mGQM," International Journal of Computer Information Systems and Industrial Management Applications, vol.4, pp.92–100, 2012.
[26] B.K. Dunn, D.F. Galletta, D. Hypolite, A. Puri, and S. Raghuwanshi, "Development of smart phone usability benchmarking tasks," Procs. of 2013 46th Hawaii International Conference on System Sciences (HICSS 2013), 2013.
[27] A.L. Delbecq, A.H.V. Ven, and D.H. Gustafson, "Group techniques for program planning: a guide to nominal group and delphi processes," Green Briar Press, 1976.
[28] ISO/IEC 25010:2011, "Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – System and software quality models," 2011.
[29] ISO/IEC 25022:2016, "Systems and software engineering – Systems and software quality requirements and evaluation (SQuaRE) – Measurement of quality in use," 2016.
[30] R. Budiu and J. Nielsen, "Usability of iPad Apps and Websites First Research Findings 1st edition," Nielsen Norman Group, 2010.
[31] R. Budiu and J. Nielsen, "Usability of iPad Apps and Websites 2nd edition, Nielsen Norman Group," 2011.
[32] J. Nielsen and R. Budiu, "Mobile Usability," New Riders, 2012.
[33] S.S. Shapiro and M.B. Wilk, "An analysis of variance test for normality (complete samples)," Biometrika, vol.52, no.3-4, pp.591–611, 1965.
[34] S. Geisser and W.O. Johnson, "Modes of Parametric Statistical Inference," Wiley-Interscience, 2006.
[35] E. Brodsky and B.S. Darkhovsky, "Non-Parametric Statistical Diagnosis: Problems and Methods," Springer, 2000.

[36] L.K. Edwards, "Applied Analysis of Variance in Behavioral Science," Marcel Dekker Inc., 1993.
[37] S. Hartshorn, "Hypothesis Testing: A Visual Introduction To Statistical Significance," Independently published, 2017.
[38] Student, "The probable error of a mean," Biometrika, vol.6, no.1, pp.1–25, 1908.
[39] B.L. Welch, "The generalization of 'student's' problem when several different population variances are involved," Biometrika, vol.34, no.1/2, pp.28–35, 1947.
[40] F. Wilcoxon, "Individual comparisons by ranking methods," Biometrics Bulletin, vol.1, no.6, pp.80–83, 1945.
[41] H. Urokohara, K. Tanaka, K. Furuta, M. Honda, and M. Kurosu, "NEM: "novice expert ratio method" a usability evaluation method to generate a new performance measure," Procs. of CHI'00 Extended Abstracts on Human Factors in Computing Systems (CHI EA'00), pp.185–186, 2000.
[42] "Logcat command line tool," https://developer.android.com/studio/command-line/logcat.html (Accessed on 27th Oct. 2017).
[43] W. Bryc, "The Normal Distribution Characterizations with Applications," Springer-verlag New York Inc., 1995.
[44] B. Shneiderman, C. Plaisant, M. Cohen, S. Jacobs, N. Elmqvist, and N. Diakopoulos, "Designing the User Interface: Strategies for Effective Human-Computer Interaction, Global Edition," Pearson Education Limited, 2017.
[45] K.A. Siek, Y. Rogers, and K.H. Connelly, "Fat finger worries: how older and younger users physically interact with PDAs," Procs. of the 2005 IFIP TC.13 international conference on Human-Computer Interaction (INTERACT 2005), vol.3585, pp.267–280, 2005.

Junko Shirogane received the B.E., M.E. and D.E. degrees in information and computer science from Waseda University, Tokyo, Japan, in 1997, 1999 and 2002, respectively. She joined Media Network Center of Waseda University as a Research Assistant in 2000 and Department of Communication of Tokyo Woman's Christian University as a Lecturer in 2003. She is now an Associate Professor of School of Arts and Sciences, Tokyo Woman's Christian University. Her research interest includes support tools for development of software with graphical user interface. She is a member of IPSJ, IEICE Japan, JSSST, HIS, IEEE and ACM.

Misaki Matsuzawa received B.E. and M.E. degrees in information and computer science from Waseda University, Tokyo, Japan, in 2014 and 2016, respectively. His research interests include support tools for evaluating software usability.

Hajime Iwata received the B.E., M.E. and D.E. degrees in information and computer science from Waseda University, Tokyo, Japan, in 2002, 2004 and 2008, respectively. He joined Media Network Center of Waseda University as a Research Assistant in 2005, Department of Network and Communication of Kanagawa Institute of Technology as a Assistant Professor in 2008 and Associate Professor in 2017. His research interest includes support tools for learning operating method of applications. He is a member of IPSJ, IEICE Japan and ACM.

Yoshiaki Fukazawa received the B.E., M.E. and D.E. degrees in electrical engineering from Waseda University, Tokyo, Japan, in 1976, 1978 and 1986, respectively. He joined Department of Computer Science of Sagami Institute of Technology as a Lecturer in 1983 and Department of Electrical Engineering of Waseda University as an Associate Professor in 1987. He is now a Professor of Department of Information and Computer Science, Waseda University. His research interests include software engineering, program optimization and computer aided design. He is a member of IPSJ, IEICE Japan, JSSST, ACM and IEEE.