Task Classification with Chronological Action History for PSP Support

Ryouta Ohashi
Department of Information Engineering
Nara National College of Technology
Nara, Japan
Email: ryouta@info.nara-k.ac.jp

Hidetake Uwano
Department of Information Engineering
Nara National College of Technology
Nara, Japan
Email: uwano@info.nara-k.ac.jp

Takao Nakagawa
Department of Information Engineering
Nara National College of Technology
Nara, Japan
Email: takao@info.nara-k.ac.jp

***Abstract*—This paper proposes a method to support Personal Software Process (PSP) in a development organization by classify the operations on a computer into a purpose of the user. PSP requires the developers to record and analyze their activity during the development process. There are several methods and systems to support the PSP, they records the operations automatically and also records a purpose of the operations (task) which records manually by the developer. Such manual recording by the developers is a barrier to introduction of the PSP system, and the cause of inaccurate record histories. Our proposal method classifies the operations into the task automatically with the chronological operation history. The method hypothesize that the each task consists of successive operation. The method classify the each operation into the task with a machine learning algorithm, Random Forests. An Experiment result shows the proposal method with chronological operation history classify the operation into the tasks more accurately than the method without the chronological operation history.**

I. Introduction

Personal Software Process (PSP) is a method to improve a ability of developer and quality of development process [1]. In a software project with PSP, developer records the task history and analyze the time spent on each task such as coding, debugging, design, meeting, and others. Based on results of the analysis, the developer improves their work or development process. Several software tools has been proposed to support the PSP process, these systems automatically record the task time [2][3][4]. However, change of tasks must records manually by the developers, hence, some task changes should be unrecorded. Also manual recording of the tasks disturbs their concentration on the tasks.

To tackle this problem, Monden et. al. proposed TaskPit, a system that records task changes automatically [5]. The system identifies the task from a front most application name and their window name, then records the task time, the number of clicks and keystrokes. For example, when ”WINWORD.EXE” is on the top with window name ”design.docx”, the system record the task as ”Edit design document.”

However, design document is refered when a developer is on the coding task, hence, task name recorded by the system and task (coding or design) is not correspond one-to-one. To improve the development process with PSP, measuring the time that spend for the each task is required. Therefore, a system that records the task and the task purpose in same time is required; Also the system should adopt a mechanism that requires no user input to avoid interrupt the developer.

In this paper, we propose a method to classify the tasks into the each purpose. Our method focused on the chronological task history of developers, and classify each task to the purpose with one of the classification algorithm; Random Forests. In the experiment, we evaluate the classification accuracy of the method.

II. Related Work

A. PSP Support System

PSP is a process improvement method for developer by collect the activity history during the his/her task. In the PSP, the developer measures the time spend in each task and compares the time with an estimation. To support this work, support systems such as Process Dashboard [2], Task Coach [3] and Slim Timer [4] have been proposed. These systems measure the time spent in each task automatically, however, users are required to input their task purpose manually, when the task is changed. Therefore, it causes inaccurate records and loss of attention to their task itself.

TaskPit is a system to support the integration of PSP into a developer’s process by record and visualize his/her tasks [5]. Fig. 1 shows a screenshot of the system. The system considers consecutive operations to an application as one task, and records the number of keystrokes, task time, and size of files which are changed during the task. For examples developer operates Microsoft Word consecutively, and then operates Eclipse, the system records two tasks: ”Edit” and ”Coding.” Also the system can classify operations to an application as different tasks by window name. Distinction of the operations to an application which used on multiple purposes (e.g., Office suite, Internet browser) allows developers more fine-grained analysis.

On the other hand, for the privacy protection, TaskPit is not record details of keystrokes, application status, window names, file names, and contents of the files. In same reason, the system records only the operations for registered application, operations for other application are recorded as ”not registered”.

Fig. 2 shows an example of recorded history by the TaskPit. The history contains the following items: Task name, start

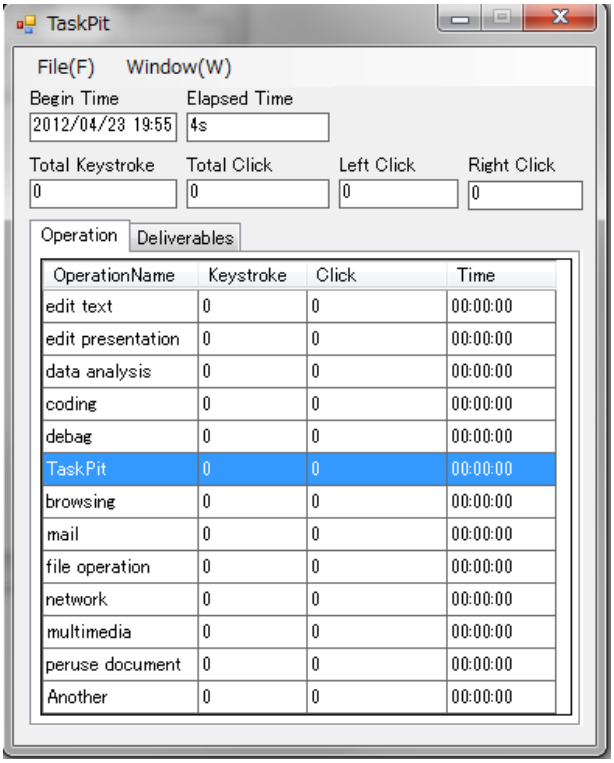

Fig. 1. Screenshot of TaskPit

```
OperationName,Start,Finish,Lclick,Rclick,Key
file operation,13:25:35,13:25:48,0,0,0
TaskPit,13:25:48,13:25:54,3,0,0
file operation,13:25:54,13:26:09,2,0,0
desktop,13:26:09:,13:26:27:,1,0,0
file operation,13:26:27:,13:26:36:,5,0,2
data analysis,13:26:36:,13:29:41:,24,0,64
desktop,13:29:41:,13:29:42:,2,0,0
TaskPit,13:29:42:,13:29:43:,1,0,0
data analysis,13:29:43:,13:29:45:,2,0,0
```

Fig. 2. Example Of Recorded History

time, end time, number of left/right click, and number of keystrokes. TaskPit detects the change of application currently operated, and records the operation history automatically. However, operation history without the purpose is hard to use for process improvement, because an analyst interprets the effectiveness of the process from the task of each operations. Our proposed method supports the process improvement with automatic classification of the operations without an user input.

B. Task Classification

Michael et. al. proposed the method to predict a task by user's context data [6]. Here, context data represents an application name, a window title and contents in the window. This method predict the task using machine learning from above features. Stumpf et. al. developed the TaskTracker and TaskPredictor [7]. TaskTracer records the operations and TaskPredictor predicts the task from the operations. Machine learning using the characteristics of a recorded operation is used for task prediction. Oliver et. al. proposed a system called SWISH [8]. The system classifies the tasks from a window title and order of traces among the windows. Also a system to improve the task effectiveness has been developed. TaskPose moves a window into a near of related window, or expands the size of an important window [9].

These systems use the title or contents of every windows as the feature. However, from the viewpoint of privacy protection, such systems must avoid recording the window title/contents which is not related with the process improvement. Basically, action history is reviewed by the developer him/herself in PSP. However, when the measurement is performed based on the instruction of the manager in an organization, action history is read by the other person. Recording the operation history without unrelated operations improves the privacy of the developers, and encourages the introduction of the PSP systems to development organizations. In this paper, we classify operations into the task using TaskPit without privacy invasion.

III. Proposal Method

This chapter describes a classification method using Random Forests to the chronological action history.

A. Operation and Task

TaskPit records the operation o toward the applications which the user permits to the system. Operation o is the all activity between the application get a focus (active window) and the application become a deactivated. Operation o composed of the following elements:

- application name
- window title
- the total nunber of left click
- the total nunber of right click
- the total nunber of keystroke

A task (purpose of the operation to an application) T is represented by a set of consecutive o:

$$T = [o_i, o_{i+1}, o_{i+2}, \cdots] \tag{1}$$

Fig. 3 shows an example of design task. Here, design task is composed of consecutive operations; "Edit a requirements specification" and "Edit a design document."

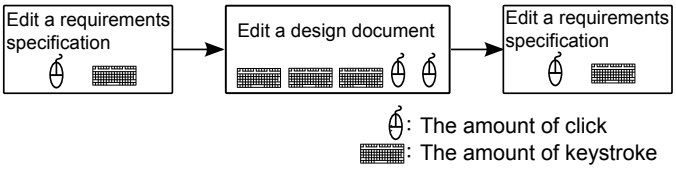

Fig. 3. Operations in a Design Task

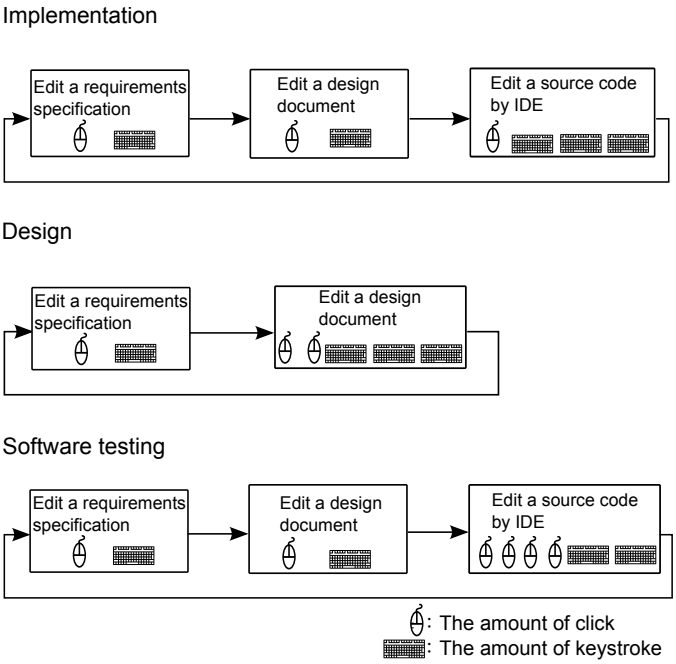

Fig. 4. Operation Pattern in Each Task

B. Random Forsts

Random Forests is a machine learning algorithm that improves the accuracy of the classification and make predictions by ensemble learning decision trees as weak learners make predictions in the conditional branch of the tree [10]. Ensemble learning is how to build a highly accurate classifier by combining several weak learners is not high-precision classifier.

It is possible to adjust the number of explanatory variables used during branch (*mtry*) and combined base classifier (*ntree*) in Random Forests. In our experiment, recommendation by Breiman [10] were used; *ntree* = 500, *mtry* = $\sqrt{M}$. Here, M is the total number of explanatory variables.

C. Task Classification with Chronological Action History

Our proposal method focuses on the chronological order of the operation history to classify the task. We hypothesize that each task have the specific pattern of operation order which consist of the task. Fig. 4 shows examples of operation pattern of three tasks; Design, Inplementation, and Software testing. In the figure, each rectangle represents an operation performed in the task. The number of mouse and keyboard describe the number of keystrokes and clicks is made at the operation. In the example, design task consists of the repetition of two operations; 1) Edit a requirements specification, and 2) Edit a design document. Because the difference of operation type and their order, the task can be classified from other two tasks.

Our method uses a characteristics of each operation to classification. In the example, both implementation task and software testing task consist of same operation type and their order. On the other hand, the characteristics of each operation in different task is different. Here, an operation “Edit a source code by IDE” in the implementation task characterized by the number of the keystroke compared with the same type operation in the testing task. Also the implementation task and design task include the same operations; edit a design document, and edit a requirements specification. In the design task, the operation can be identified as “Edit a design document” because of the number of keystrokes. In the implementation task, the operation can be identified as “Browse a design document” because a less keystrokes.

Our method classify the operations into the tasks by Random Forests algorithm using order of operation and the characteristics of each operation. Hence, characteristics of operations at before and after the target operation is used as input variables.

IV. Experiment

A. Environment

Five students of information technology department at Nara National College of Technology participated in an experiment which record a task history with TaskPit. Author installs the TaskPit into Windows PCs which subjects usually using at their laboratory. As mentioned in Section III-B, TaskPit relates the operations to the application with a task based on the task list provided from the user. In the experiment, author created the task list from the results of interview to the each subject. Applications that out of the list are recorded as ”not registered”. All task performed at five days on weekdays (from Monday to Friday) are recorded.

B. Task Purpose

In the experiment, we evaluate accuracy of our method by comparing the output of the method to inputs from the subjects. Therefore, we implemented a function that allow the users to input the task manually. Fig. 5 shows a screenshot of the TaskPit after the modification.

List of the tasks was prepared by the author based on the pre-interview to each subject. The task used in the experiment is as follows: Research (Thesis), Research (Presentation), Research (Coding), Research (Survey), Research (Other), Homework (Signal Processing: SP), Homework (Software Engineering: SE), Homework (Other), Break, Other. ”No select” is also prepared to prevent an incorrect record by the lack of user input. We divided the task about the ”Research” with finer-grain than other purpose, because the period of the experiment is close to the deadline of thesis submission. Fig. 6 shows an recorded operation history with task name.

We evaluate the accuracy of our method by comparing the user input and the classification by the method. Details of evaluation method described in Section IV-C.

C. Evaluation

We compare the accuracy between the classification with chronological task information and the classification without the information. Following four datasets are compared: 1) Target operation (not include the chronological information): target, 2) Target operation and a operation before and after the

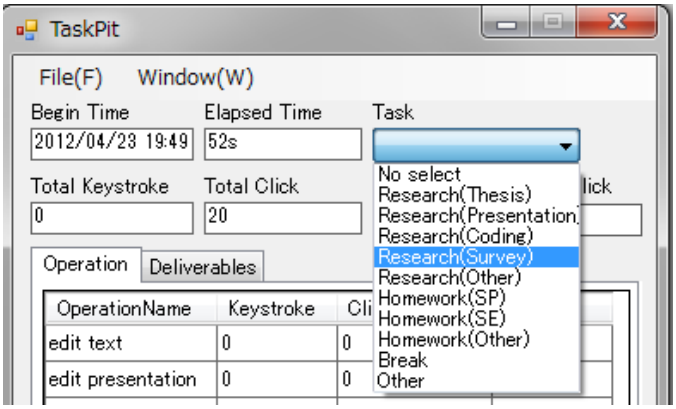

Fig. 5. Modified TaskPit

```
OperationName,Lclick,Rclick,Key,Task
TaskPit,19,12,0,Noselect
TaskPit,3,0,0,Homework(SE)
browsing,35,3,153,Homework(SE)
another,2,0,0,Homework(SE)
browsing,12,0,0,Homework(SE)
file operation,9,0,0,Homework(SE)
edit text,9,0,4,Homework(SE)
file operation,1,0,0,Homework(SE)
edit text,1,0,0,Homework(SE)
```

Fig. 6. Task Name in Operation History

target: target±1, 3) Target operation and two operation before and after the target: target±2, 4) Target operation and three operation before and after the target: target±3, Each operation includes the operation name, number of left/right click, and keystrokes. An example of the classification result is shown in Table I. Each column shows an task which subjects input to the system, and the each row describes the result of classification by the our method.

To evaluate the classification accuracy, we use three metrics, recall, precision, and F1-value. The classification result of task T expressed with four measurements shown in Table II.

TABLE I
EXAMPLE OF CLASSIFICATION RESULT

		Actual Task			
		SW	SP	Break	Thesis
Prediction	SW	167	0	41	33
	SP	0	14	1	0
	Break	56	7	282	54
	Thesis	7	0	35	140

TABLE II
MEASUREMENT FOR CLASSIFICATION ACCURACY

		Actual Task	
		T	Not T
Prediction	T	TP	FP
	Not T	FN	TN

- True Positive (TP): Number of tasks whose true is T correctly classified to task T by the classifier.
- False Positive (FP): Number of tasks whose true is not T misclassified to task T by the classifier.
- False Negative (FN): Number of tasks whose true is T misclassified to wrong task T the classifier.
- True Negative (TN): Number of tasks whose true is not T classified to task which is not T by the classifier.

Precision is the percentage of the task which belong to T that is classified to T by classifier. Precision can be expressed as the following expression:

$$precision = \frac{TP}{TP + FP} \tag{2}$$

Precision takes a range from zero to one, the higher the value means tasks are classified into T with lesser misclassification.

Recall is the percentage of the task that is classified to T by classifier, which belong to T. Recall can be expressed as the following expression:

$$recall = \frac{TP}{TP + FN} \tag{3}$$

Recall takes a range from zero to one, the higher the value means classifier classifies the tasks more correctly into T.

F1-value is the harmonic mean of precision and recall, it can be expressed as the following expression:

$$F1 - value = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \tag{4}$$

F1-value takes a range from zero to one, indicates high value when the both precision and recall are high.

In the evaluation, two-fold-cross-validation is adopted; a half of dataset is used for learning, and another half of dataset is used for evaluation. Also we repeat the experiment ten times for each dataset, then evaluate the average value of the result.

V. RESULT AND DISCUSSION

A. Overall Results

Table III, IV, and V shows a result of experiment. In the table, each value shows the average of metrics which are calculated on each subject. Fig. 7 describes the same metrics as a bar chart. The figure shows the all metrics increase when the number of operation at before and after the target operation is increased. That is, the operation at before and after the target operation is an useful information to predict the task purpose of the operation. Also the variance of each metrics were small.

In addition, tasks which observed frequently during the measure period were seen particularly high classification accuracy. During the experiment period, subject C performed $Coding$ (55.7%, 299/537), $Thesis$ (24.9%, 134/537), SP (4.4%, 24/537), and $Break$ (14.8%, 80/537) respectively. The F1-value of each task in this data shows $Coding$ (0.901), $Thesis$ (0.910), SP (0.771), and $Break$ (0.646) respectively. The results depict that the our method classifies the tasks which mainly performed with high accuracy. This is an useful feature to adopt the method to PSP, because the improvement

TABLE III
CLASSIFICATION RESULT PRECISION

	Target	Target±1	Target±2	Target±3
SubjectA	0.671	0.843	0.863	0.875
SubjectB	0.435	0.657	0.738	0.772
SubjectC	0.494	0.640	0.682	0.714
SubjectD	0.509	0.683	0.744	0.791
SubjectE	0.496	0.640	0.733	0.739
Average	0.521	0.693	0.752	0.778
Variance	0.006	0.006	0.004	0.003

TABLE IV
CLASSIFICATION RESULT RECALL

	Target	Target±1	Target±2	Target±3
SubjectA	0.612	0.756	0.769	0.773
SubjectB	0.375	0.538	0.559	0.570
SubjectC	0.477	0.628	0.654	0.646
SubjectD	0.447	0.647	0.691	0.723
SubjectE	0.419	0.574	0.612	0.644
Average	0.466	0.629	0.657	0.671
Variance	0.006	0.006	0.005	0.005

TABLE V
CLASSIFICATION RESULT F1-VALUE

	Target	Target±1	Target±2	Target±3
SubjectA	0.634	0.786	0.801	0.807
SubjectB	0.375	0.563	0.585	0.602
SubjectC	0.457	0.622	0.648	0.639
SubjectD	0.448	0.655	0.707	0.738
SubjectE	0.434	0.595	0.646	0.675
Average	0.492	0.659	0.701	0.721
Variance	0.008	0.006	0.006	0.006

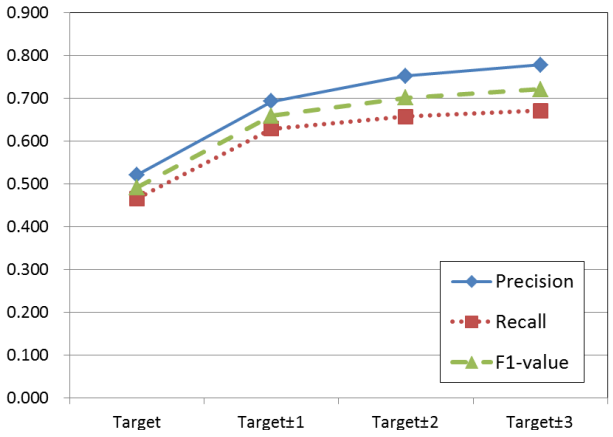

Fig. 7. Classification Result

of the main task is an effective approach in process improvement.

B. Subjects Shows C Different Result

One of the subjects (Subject C) shows a different result from other four subjects. Table VI shows a classification accuracy in the subject. The table shows a F1-value of Homework (SE) is low, compared with other tasks. In the Homework (SE), also the value of recall is low, while the value of precision is high. Detailed analysis reveals that 66.4% of the operations

TABLE VI
CLASSIFICATION ACCURACY IN SUBJECT C

	SW	Break	Thesis	Average
Precision	0.681	0.621	0.841	0.714
Recall	0.259	0.901	0.777	0.646
F1-value	0.376	0.735	0.808	0.678

```
OperationName,Task
Twitter,Break
browsing,Break
Twitter,Break
browsing,Break
Twitter,Break
browsing,Break
Twitter,Break
browsing,Break
Twitter,Break
Twitter,Homework(SE)
file operation,Homework(SE)
browsing,Homework(SE)
Twitter,Homework(SE)
browsing,Homework(SE)
file operation,Homework(SE)
coding,Homework(SE)
edit text,Homework(SE)
Twitter,Homework(SE)
```

Fig. 8. Operation History of Subject C

in the Homework (SE) task are classified into a Break task. Figure 8 shows the operation history of subject C recorded during the experiment. The figure shows the subject performed similar sequence of operation during the two tasks; Home work (Software Engineering) and Break. In these tasks, the subject uses Internet browser and Twitter client alternately.

The result suggests that the proposal method decrease the classification accuracy when the different tasks consist of similar operation histories. In this problem, additional features of operation would increase the accuracy. For example, measurement of the file sizes in the specific directories reveals what files are modified during the task. Improvement of the classification accuracy with additional operation features is a our future work.

C. Importance of Features

In the Random Forests algorithm, Gini's Index GI is used as contribution rate of each explanatory variables. GI is calculated with a following equation;

$$GI = 1 - \sum_{c \in C} p_c^2 \tag{5}$$

In this paper, the contribution rate is calculated for each data set. Fig. 9 describes an example of the contribution rate in each data set from Subject A. The figure depicts that the most important variable is the operation name in each task before and after the target operation. The total number of left click follows the operation name, then the total

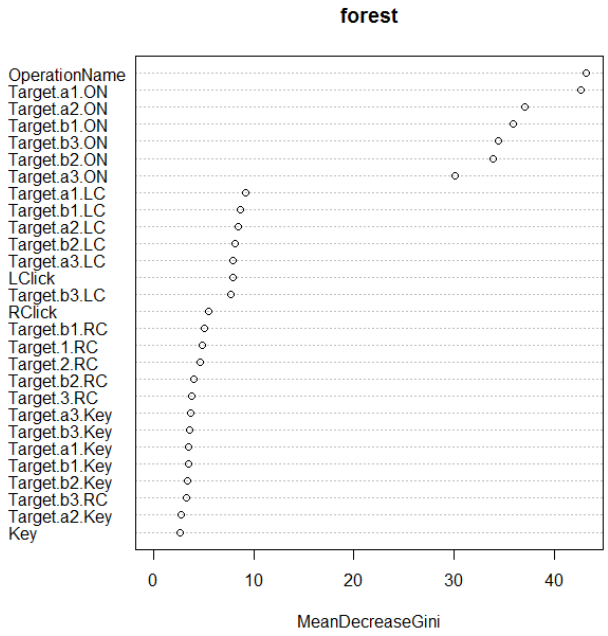

Fig. 9. Priority Index of Each Features

number of right click and the total number of keystrokes. This trend is also observed on other subjects. Hence, chronological operation history improves the prediction accuracy of target operation than the details of the target operation such as the total number of the keystrokes. Also the figure suggests that the classification accuracy may improve by eliminating the variables that have lower contribution rate.

VI. Summary

This paper proposed the method to classify the operation on the computer into the task with Random Forests. The method adopts the chronological operation history recorded by a PSP support system, TaskPit, and predicts the task of the target operation from the operations at before and after the target. The result of the experiment described the operation at before and after the target operation is an useful information to predict the task of the operation. The method had high classification accuracy when the target operation belongs to the task which was mainly performed during the experiment. Therefore, our proposal method is useful to measure the development activity for process improvement by the PSP.

As a future work, we aim to improve the classification accuracy by adding other explanatory variables and/or eliminate the variables which has low contribution rate. Also the evaluation with other data sets which include more number of the operation before and after the target operation clarify the usefulness of the chronological information. Classification with a data set which include only the operation before the target operation is also an interesting investigation for real-time task prediction.

Acknowledgment

This work is being conducted as a part of Grant-in-aid for Young Scientists (B), 24700038.

References

[1] W. S. Humphrey, “Introduction to the personal software process,” Addison-Wesley, 1996.
[2] Process Dashboard, http://processdash.sourceforge.net/pspdash.html.
[3] Task Coach Your friendly task manager, http://members.chello.nl/f.niessink/
[4] SlimTimer Time Tracking without the Timesheet, http://www.slimtimer.com/
[5] A. Monden, Y. Kamei, H. Uwano, and K. Matsumoto, “A Software Task Measurement System for Process Improvement,” In Proc. Workshop on Foundation of Software Engineering (FOSE), pp.123-128, 2008 (In Japanese.)
[6] M. Granitzer, A. S. Rath, M. Kroll, C. Seifert, D. Ipsmiller, D. Devaurs, N. Weber, and S. N. Lindstaedt, ”Machine Learning based Work Task Classification,” Journal of Digital Information Management, pp.306-313, 2009.
[7] S. Stumpf, X. Bao, A. Dragunov, T. G. Dietterich, J. Herlocker, K. Johnsrude, L. Li, and J. Shen, “Predicting User Tasks: I Know What You’re Doing!,” In Proc, the 20th National Conference on Artificial Intelligence (AAAI), 2005.
[8] N. Oliver, G. Smith, C. Thakkar, and A. C. Surendran, “SWISH: Semantic Analysis of Window Titles and Switching History,” In Proc. the 11th international conference on Intelligent user interfaces, pp.194-201, 2006.
[9] M. Bernstein, J. Shrager, and T. Winograd, “Taskpose: Exploring Fluid Boundaries in an Associative Window Visualization,” Evaluation, pp.231-234, 2008.
[10] L. Breiman, “Random Forests,” Machine Learning, Vol.45, No.1, pp.5–32, 2001.