2012 13th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing

# A Study of Student Experience Metrics for Software Development PBL

Umekawa Kohichi, Hiroshi Igaki, Yoshiki Higo, Shinji Kusumoto
*Graduate School of Information Science and Technology*
*Osaka University*
*Osaka, Japan*
*k-umekaw@ist.osaka-u.ac.jp, igaki@ist.osaka-u.ac.jp, higo@ist.osaka-u.ac.jp, kusumoto@ist.osaka-u.ac.jp*

***Abstract*—In recent years, the increased failure originated in the software defects, in various information systems causes a serious social problem. In order to build a high-quality software, cultivation of ICT(Information and Communication Technology) human resources like a software engineer is required. A software development PBL(Project-based Learning) is the educational technique which lets students acquire knowledge and skill spontaneously through practical software development. In PBL, on the other hand, it is difficult to evaluate not only the quality of the product but also the quality of the development process in the project. In this paper, we propose the student evaluation metrics to assess the development process in PBL. The student evaluation metrics represent LOC(Lines of Code) and development time for each product developed by a student. By using online storage, these metrics can be measured and visualized automatically. We conducted an experiment to evaluate the accuracy of the metrics about development time. As a result, we confirmed that development time metrics can be measured with approximately 20% of error.**

***Keywords*-metrics; PBL; process improvement; visualization;**

## I. Introduction

In recent years, software problems are increasing, and it is becoming a serious social problem. It is pointed out that quality and quantity of talented people for building good software are insufficient. Training of sophisticated ICT(Information and Communication Technology) talented people is required in the industrial world. In the university, including department of information, education and learning technique called PBL(Project-based Learning) based on actual software development are performed with various forms[1]. The PBL is an educational system which aims at advanced ICT talented people's training and provides an opportunity of autonomous study for students. Students form a group and tackle several practical tasks. In this paper, we focus on the software development PBL in which students develop actual software product. In such kinds of PBL, students can learn skills of project management or communication practically through the software development. Usually, multiple students in the same group cooperate in the software development. Then, it is difficult for a teacher to evaluate how much the individual student contributed to the development group. We propose a method which collects development logs of each student in the software development PBL with using an online storage service. Based on the method, we realized analysis and visualization environment which evaluate experiences of the individual student with little effort of students and teachers. Henceforth, in section 2, we describe the existing research on evaluation methods for the software development PBL. Section 3 denotes about our proposed method including several metrics for evaluating students' experience. Section 4 presents implementation of our system. We write an experiment and a discussion about our system in section 5 and 6.

## II. Preliminary

### A. Software Development PBL

In this paper, we call the software development PBL a SDPBL[2]. The following subjects may be given to the group of the students who develop a project, in addition to the feature of common PBL.

- T1: Can the product which fulfills given specification be implemented within a certain period deliberately?
- T2: Are the tasks appropriately shared within the project group?
- T3: Can the required tasks fully be carried out in the group?

The SDPBL is an educational approach which trains students devising Project Management skills for smooth software development process spontaneously. Therefore, the student group needs to manage their development process in the viewpoint of time for delivery, task assignment and software quality. T1 is related to time for delivery. The student group manages software development process and must protect the given time for delivery. T2 indicates that every student should have the same chance to acquire experience of software development in the SDPBL. The student group needs to progress software development efficiently, taking T2 into consideration. T3 is related to software product quality. In order to develop high-quality software, students must perform not only implementing but tests and reviews for products.

ITSpiral[3] contains the SDPBL program. In the SDPBL program, through the experience of group software development, the students learn the difficulty of working in a group and the importance of all-around skills such as communications, schedule management, and project management. Furthermore, through looking the development process back, the students obtain the further understanding of the development process and process improvement.

Shibaura Institute of Technology performs a class about PBL which is named advanced information practice 1B[4].

978-0-7695-4761-9/12 $26.00 © 2012 IEEE
DOI 10.1109/SNPD.2012.76

In the class, groups of students develop several kinds of products such as documents and source codes. This class aims at training of creativity and the problem-solving capability for advanced ICT talented people.

SDPBL environment differs for every school. Definitions of the environment which is used in many SDPBLs are shown below.

- P1: Development environment to work on the development project is used by the students.
- P2: Development support environment supports the development process.
- P3: Teaching support environment facilitates the analyses of the development process by instructors.

### B. Evaluation for Each Student and Group in SDPBL

Evaluation in SDPBL is performed for each group and each student. An instructor in SDPBL usually adopts the evaluation approach shown below.

- E1: Evaluation by an instructor's subjectivity
- E2: Evaluation by an examination or a report about SDPBL by each student
- E3: Evaluation by a questionnaire by each student and group

In E1, an instructor observes and estimates the progress of development of each group and student. The instructor evaluates subjectively, and load of the evaluation is so heavy. In E2, an instructor gives a report or an examination based on the group development in SDPBL to the students and evaluates by the result. Although the report and the examination are essential in the view point of estimation of comprehension about the group development, it is necessary to examine carefully whether the examination or the report are in agreement with the contents of the SDPBL. The E3 contains a group questionnaire and a student questionnaire. The questionnaires are significant in the meaning of grasp of students' subjective evaluation qualitatively.

As a result, almost all conventional evaluation approach aims at estimating students and groups qualitatively. The software management method PSP(Personal Software Process) is one of the method which evaluates a student quantitatively[5]. In the PSP, individual development data are recorded based on strictly-defined metrics. The development process based on the PSP enables evaluation using recorded individual development data. However, the development data collected by each developer manually may often be inaccurate. Then, we propose the metrics which enables evaluation of each student development experience in the SDPBL.

## III. Student Experience Evaluation Metrics for Software Development PBL

Our proposed student experience evaluation metrics consists of the following key ideas.

- K1: Automatic collection of development logs by using an online storage service.
- K2: Detailed development experience metrics for every student based on the development logs.
- K3: Contribution evaluation method by the comparison of development experience metrics in a group.

We use the online storage service "Dropbox"[6]. Dropbox shares data in a user's HDD between two or more PCs through the online storage service on the Internet.

We share students' working directories with an instructor using the online storage. The online storage has the function of Version Control System, and the storage records contents of files whenever a user saves files. Our system collects contents of files recorded by the storage as the user's fine-grained development log. Based on the collected development log, our system measures experience of each student such as development time and lines of code. Comparing an individual's measured quantitative experience within a group indicates contribution degree to a project for each student. Henceforth, we explain these key ideas in detail.


```
[
{
    "rev": "40000000d",
    "bytes": 0,
    "modified": "Wed, 20 Jul 2011 22:41:09 +0000",
    "path": "/hi2",
    "is_dir": false,
    "mime_type": "application/octet-stream"
},
{
    "rev": "10000000d",
    "bytes": 3,
    "modified": "Wed, 20 Jul 2011 22:40:43 +0000",
    "path": "/hi2",
    "is_dir": false,
    "mime_type": "application/octet-stream"
}
]
```


Figure 1. Version information examples by the API of the storage service

### A. Collecting Development Log using Online Storage Service

The online storage service "Dropbox" has the following features.

- F1: The data in a folder specified by a user are saved automatically at the storage on the service.
- F2: The storage service has a feature of Version Control System.
- F3: The folder specified by a user is sharable with other users.
- F4: APIs for accessing the storage service are exhibited.

By using the online storage service, whenever a user updates any file, time stamp, and the contents of files are recorded. All history of the recorded files can be accessed through the APIs by users. The API of the storage service returns version information for every file as shown in the figure1. In the version information, "rev", "bytes", "modified", "path", and "mime_type" indicate a version number, a size of the file, an updated time, path information of the file, and a type of the file respectively.

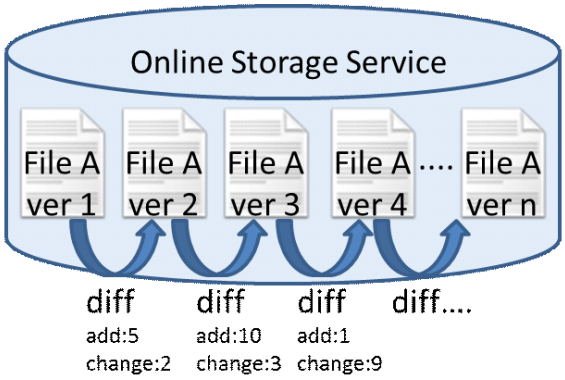


Figure 2. Example of the accumulated LOC

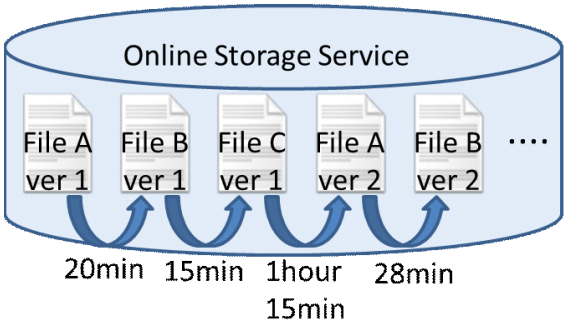


Figure 3. Example of the accumulated development time

By using such online storage service and APIs, the fine-grained development log than the existing version control system is collectable for each user. In the following section, we describe our metrics using this development log.

### *B. Development Experience Metrics of Each User*

Development Experience Metrics contains two kinds of metrics. The 1st metrics is an accumulated LOC of a product implemented by a student until a certain time. When a student implements a product, addition, deletion, and change to the file are performed. Here, we define the accumulated LOC as the total of the lines of code added by the student, after the file is newly created. In addition, the number of change lines is included in the accumulated LOC as the number of lines is added after deletion.

In order to measure the accumulated LOC, all the version information of the files implemented by each student is collected through the API of the online storage service. As shown in the figure 2, suppose that two or more versions of a certain file A exist in the online storage. Our system acquires all the version information of a certain file through the API and calculates the diff between adjoining version.

In this example, the diff between ver1 and ver2 of the file A is add 5(LOC) and change 2(LOC). Similarly, the diff between ver2 and ver3 is add 10 and change3. As a result, the accumulated LOC(aLOC) of the file A to ver3 is add 15 and change 5, or simply 20 aLOC.

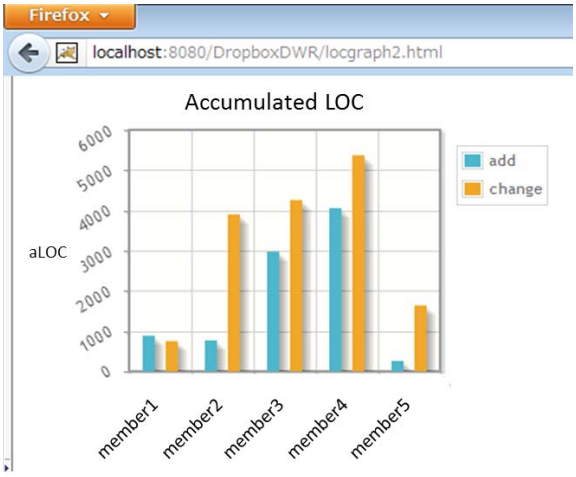


Figure 4. Example of comparison about aLOC within a group

The 2nd metrics is an accumulated development time for every file. The online storage records contents of the file information on the service, whenever a user saves a file. Then, we measure the accumulated development time based on the update time of all files saved by a user.

- Step1: Collects history of update time of all files which are saved by a certain user.
- Step2: Sorts all versions of all the files in order of update time.
- Step3: Calculates the diff of the update time between two continuous versions of files.
- Step4: Adds the diff of the update time to the file updated subsequently as the accumulated development time of the file, if the diff of the update time is below predefined threshold.

The figure 3 shows an example of the accumulated development time. In the figure, all versions of all files are sorted in order of update time. The diff of the update time between ver1 of file A and ver 1 of file B is 20 minutes. Here, suppose that threshold about the diff of update time is 1hour. The diff of the update time 20 minutes is added to the accumulated development time of file B. Similarly, 15 minutes and 28 minutes are added to the accumulated development time of file C and file B respectively. As a result, the accumulated development time(aDEV) of the file A, B, C in this example are calculated as 0 minutes, 48 minutes and 15 minutes respectively.

As the above example shows, the online storage service cannot record the start time of user's development. We have to take into consideration the error about the development start time, when we measure the accumulated development time.

### *C. Evaluating Contribution of Students in Development Group*

We calculated the accumulated LOC and the accumulated development time of each product file which each

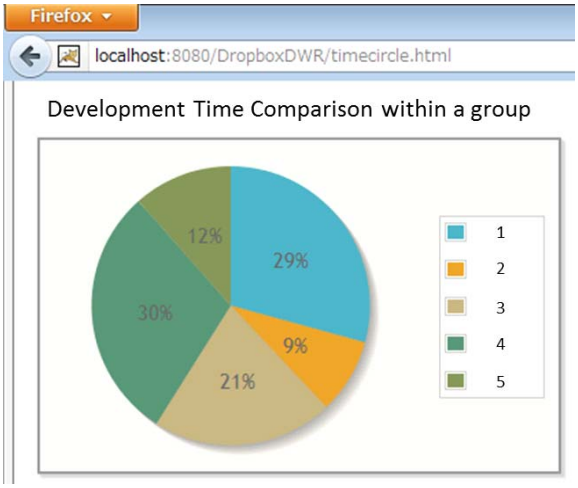


Figure 5. Example of comparison about aDEV within a group

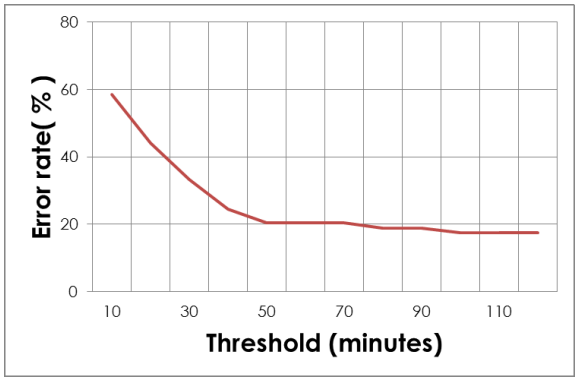


Figure 6. Magnitude of the error rate in application development

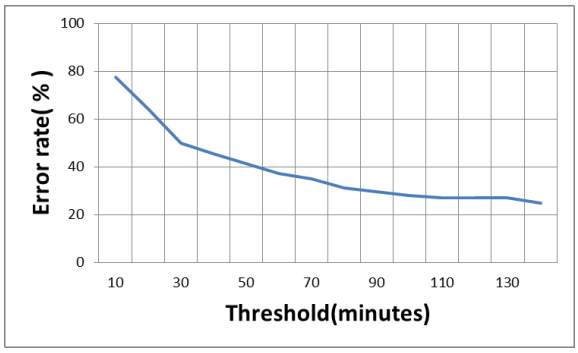


Figure 7. Magnitude of the error rate in paper writing

student has developed on the online storage. Visualization of comparison about our metrics within a group can make students recognize the deviation of the work load of group development. The figure 4 shows an example of the bar graph about aLOC for every member within a group. This example indicates that the difference of aLOC between member 1,5 and member 2,3 and 4 is very large. Considering that the SDPBL is the educational approach, it is desirable that the experience which students get to be qualitatively or quantitatively similar among several students.

Similarly, the figure 5 shows an example of the pie chart indicating the percentage of the accumulated development time in the group. Combining Figure 4 and Figure 5 makes it easy to grasp the situation of SDPBL. For example, although the aDEV of member 1 is long, there are few aLOC of the member. The monitoring result may imply that the member 1 tackles too difficult problem about his development.

## IV. Implementation

We adopted Java SE 6 update 20 and JavaScript as development languages for our system and developed on the Eclipse 3.7 and the NetBeans IDE 7.0.11. We used Dropbox4j[7] to access Dropbox API using OAuth, and db4o for saving proposed metrics as DBMS. Furthermore, DWR framework[8] was adopted for integrating Java and JavaScript to create our Ajax web application. We used jQuery and jqplot[9] of the JavaScript library for the purpose of visualizing our metrics.

## V. Experiment for estimating errors of the accumulated development time

Although we can collect the update time of each file, we cannot know a student's operation start time. Since exact operation start time cannot be obtained, the threshold or a work environment may affect the magnitude of the error on the aDEV. Therefore, in this research, we conducted an evaluation experiment for checking the magnitude of the error about the accumulated development time measured by our system.

In the experiment, 6 students developed products on the online storage service. We used an application called WorkTimer to record actual development log, i.e., operation start time, and finish time for each development task manually.

Based on the actual development log for about two days captured by each student with WorkTimer, we analyzed the error between aDEV and the actual development log. In addition, students conducted two kinds of development task. One is Java application development and another is paper writing. Figure 6 and 7 show the relation between the value of the threshold for calculating the aDEV, and the magnitude of the error in application development and paper writing, respectively. In the figure, a threshold is displayed on the horizontal axis, and the magnitude of the error of aDEV is displayed on the vertical axis.

The result of the figure 6 shows that the error decreased to about 20.5 %, when we set up the threshold to 50 minutes in application development. Another result of the figure 7 shows that the error decreased to about 31 %, when we set up the threshold to 80 minutes in application development. The magnitude of the error varied with the kinds of task. This difference is considered to originate from the difference of development environment. In this experiment, every student developed the Java application

on Eclipse IDE, and wrote the paper using a text editor. In the development on the Eclipse, the students have to save the source code files at the time of compile or execution of the Java application. In the development on the text editor, the students save files by their decision. That is, in the application development, files are saved more frequently.

## VI. Discussion

In this paper, we proposed two kinds of metrics to measure and evaluate the experience of the students on group development in SDPBL. Those metrics can be measured, merely by setting up the online storage service. In the experiment, we checked that the accumulated development time was measurable with about 30 In almost all the results of the experiment, values smaller than the actual development time measured by the student manually were calculated as aDEV. We consider that more exact aDEV can be measurable by adding the development time in consideration of the error than our current system.

The experimental result indicated that the adequate value of the threshold to calculate the aDEV should be determined based on the development environment of users. We are planning to propose automatic setting method of the threshold value, based on the type of files or a development environment.

Hackystat is a monitoring environment for collecting development log of each user[10]. This environment can collect a history of user control such as editing, opening, and closing of a file by adding a plug-in called "sensor" to a development environment, such as eclipse. Although the hackystat can collect development log more detailed than our system, there is a problem that the development environment which can be used, is restricted.

We are going to apply these metrics to the educational methods in SDPBL in order to improve PM technique.

## VII. Conclusion

In this research, we proposed two kinds of student experience metrics, an accumulated LOC and an accumulated development time to visualize and compare the effort of each student in SDPBL quantitatively. The online storage service made it possible to collect the development logs of each student for measuring such metrics at a low cost. We are planning to improve the accuracy of the accumulated development time with using an estimation method of operation start time based on the accumulated LOC.

## Acknowledgment

This research was partially supported by the Japan Ministry of Education, Science, Sports, and Culture [Young Scientists (B) (No.24700030)]

## References

[1] E. de Graaff and A. Kolmos, *Management of Change Implementation of Problem-Based and Project-Based Learning in Engineering*. Sense Publishers, 2006.

[2] N. Fukuyasu, S. Saiki, Y. Mizutani, H. Igaki, and Y. Manabe, "An adaptive teaching on software development PBL," *Japanese Society for Information and Systems in Education*, vol. 26, no. 7, pp. 169–176, 3 2012, (in Japanese).

[3] IT Spiral, "IT Specialist Program Initiative for Reality-based Advanced Learning," http://it-spiral.ist.osaka-u.ac.jp/index.html.

[4] S. Komiya, "Exercise Class for Software Development technique based on PBL," *http://www.grace-center.jp/downloads/itsp/talk7.pdf*, 2011, (in Japanese).

[5] M. Unterkalmsteiner, T. Gorschek, A. Islam, C. Cheng, R. Permadi, and R. Feldt, "Evaluation and measurement of software process improvement-a systematic literature review," *IEEE Transactions on Software Engineering,*, vol. 38, no. 2, pp. 398 – 424, 2011.

[6] Dropbox Inc., "Dropbox," https://www.dropbox.com/.

[7] "Dropbox4j," https://github.com/Frostman/dropbox4j.

[8] "Dwr - easy ajax for java," http://directwebremoting.org/dwr/index.html.

[9] "jqplot," http://www.jqplot.com/.

[10] P. Johnson, "Requirement and design trade-offs in hackystat: An in-process software engineering measurement and analysis system," in *Proc. of 1st Int. Symposium on Empirical Software Engineering and Measurement, IEEE Computer Society Press*, 2007.