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

Experimental Report of the Exercise Environment for Software Development PBL

Naoki Fukuyasu*, Sachio Saiki†, Hiroshi Igaki‡ and Yuki Manabe‡
*Faculty of Systems Engineering, Wakayama University
930 Sakaedani, Wakayama-shi, Wakayama, Japan
Email: fukuyasu@sys.wakayama-u.ac.jp
†School of Information, Kochi University of Technology
185 Miyanokuchi, Tosayamada-cho, Kami-shi, Kochi, Japan
Email: saiki.sachio@kochi-tech.ac.jp
‡Graduate School of Information Science and Technology, Osaka University
1-5 Yamadaoka, Suita-shi, Osaka, Japan
Email:{igaki,y-manabe}@ist.osaka-u.ac.jp

***Abstract*—This paper summarized experiences of practical software development exercise in PBL style activities from organizer perspective. The object of this PBL is nurturing advanced knowledge as advanced information and communication technology (ICT) engineers.**

A main pillar of this report is trace the 5-year history of three sub environments such as development, development support and teaching support environment which are badly need to hold our software development PBL, from problem and its solutions viewpoint.

***Keywords*-PBL; exercise environment; software engineering education;**

I. Introduction

In recent years, with the increase of the scale and complication of the information system, shortage of the information system engineer is pointed out. At first, industrial trend was toward increasing the number of engineers. Nowadays that trend changes to the quality of the engineer instead of quantity. Therefore, nurturing advanced engineers of information and communication technology is recognized as an important issue [1]. Against that requirement, there are many trials by collaboration between industry, academia and government.

Industry requires that undergraduate and graduate students should learn not only coding skills but also management and communication skills which are necessary for project management through practical exercises of software development. One way to react to the requirement of industry, practical software development exercises in PBL [2] style (here after referred as "software development PBL") has been implemented on various educational organizations [3]–[5]. It is difficult to master skill required for employment of a project only by the usual lecture and exercise. On the other hand, in the PBL style exercise, it is said that it might be possible to acquire such skills because an attendance student performs actual project management actively [5].

Against this background, we have also implemented a software development PBL in IT Spiral, a program for developing advanced IT specialists [6]. In this paper, we summarize our experiences of software development PBL activities.

The organisation of this paper is as follows. Details of IT Spiral and its PBL style exercise are presented in section II, then our improvements on prepared exercise environment for the PBL are reported in section III. In section IV, we describe future works of our environments, and finally we present the conclusion of this report.

II. IT Spiral and Software Development PBL

A. IT Spiral

IT Spiral is an education program for nurturing advanced software engineers by 9 graduate schools and 4 private corporations under the leadership of Osaka University [7]. At first, it was started as one of the pioneering IT specialist development promotion programs led by the Ministry of Education, Culture, Sports, Science and Technology (MEXT). Even after the support by MEXT ended, the program is being held continuously for 5 years until now including the supported period.

Its one-year curriculum features a series of basic software courses, advanced software courses, and practical software development courses. Especially, the series of practical software development courses is held at one exercise room and about 30 or 40 students from 9 graduate schools meet in the room. The series consists of 16-day lectures and exercises for a total of 6 course credits. A software development PBL is held a part of the series. Figure 1 shows a scene of activity in our PBL.

B. Practical Software Development Exercise

IT Spiral holds the software development PBL named "practical software development exercise." The PBL consists of 3-day exercises and 1-day final presentation which is opportunity to report the activities of students' project and its improvement. It spends about one and a half months from the first exercise day to the final presentation day. A

978-0-7695-4761-9/12 $26.00 © 2012 IEEE
DOI 10.1109/SNPD.2012.56

Figure 1. Practical scene of the software development PBL in IT Spiral

team consists of 5 or 6 students from different universities. Students of each team implement the web application which specifications are given by teachers. The web application is based on the Java EE (Enterprise Edition). The each team implements it using JSP and Java, such as business logic, DAO, Action of Struts framework and so on. Students should also implement some test cases which test Java classes implemented by themselves. The roles, for example who implements a JSP and who implements a test case, are decided by the team meeting. The practical software development exercise aims to attain the following targets:

- Through the experience of software development from implementation to integration testing by a team, the students learn the difficulty of working in a team and the importance of all-around skills such as communication, schedule management, and project management.
- Through looking back the development process, the students obtain the further understanding of development process and process improvement.

III. Improvement of Exercise Environment

We prepared and improved the exercise environment for the software development PBL in IT Spiral. It consisted of some sub environments. In this section, we describe our improvements of three sub environments, such as development, development support and teaching support environments, for 5 years.

Figure 2 shows the transitions of three sub environments. The details of the transitions of each sub environment are described in following.

A. Development Environment

The development environment consists not only of tools for describing and compiling programs but also environment for testing the implemented system. In software development PBL, the system is developed by a team of more than one member. It is important that all members prepare same development environment. The testing results may be different, if the members use different environment. It is complex to find out the cause of the problem.

In our PBL, students implement a Java EE (Enterprise Edition) based web application system by using the development environment in which JDK (Java Development Kit) / JRE (Java Runtime Environment), Apache Tomcat, Eclipse, and so on are installed. In the 1st year, the exercise was held by using students' laptop PC. We made students install some software tools specified by us on their own PC. Because of the difference of Operating System such as Windows XP and Vista and the difference of pre-installed software especially JRE which have different version numbers, it has occurred that one student got the correct result on his environment and another got the wrong result from same implementation. In the 2nd year, then we have tried to prepare laptop PCs installed same possible Operating System by cleaning each PC before the exercise. Moreover, in the 3rd year, we have tried standardization of the development environment by distributing common image of virtual machine to students.

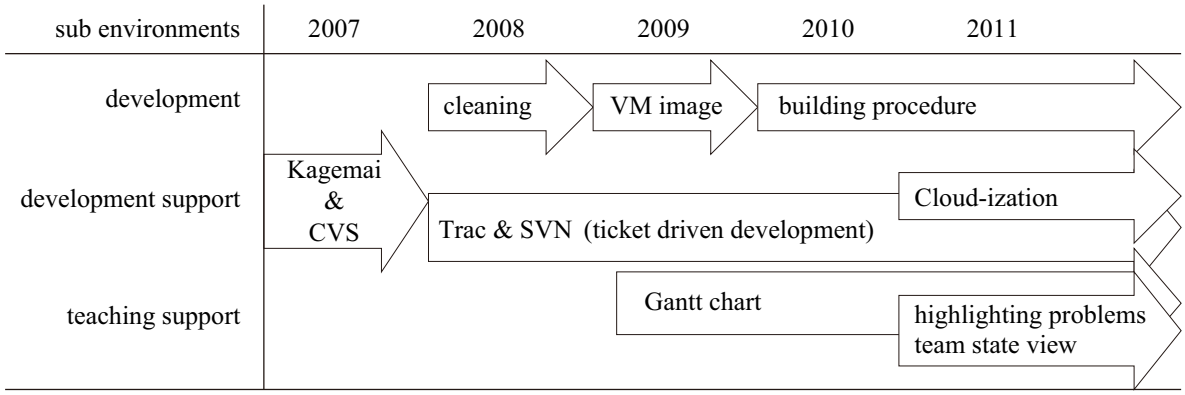

Figure 2. The transitions of three sub environments for 5 years

On the other hand, it became clear that hardware performance was important for execution of that virtual machine. Especially the students who used laptop PC with less than 2 GBytes RAM complained about slow performance and said that it was impossible to implement by their own way. So, by showing the procedures of building that development environment for students, we considered that they could prepare the environment whose versions of installed software corresponded, even if they did not use virtual machine.

B. Development Support Environment

Communication in system development by a team is generally important, so development is usually carried out by using some support tools for that communication. It is also applied to software development PBL. The typical support tools of communication in system development are for example, a versioning system, a bug tracking system, etc.

In our original PBL, students carried out the development by using CVS as a versioning system and Kagemai as a bug tracking system for development support environment. These support tools were prepared by us. Then we tried two improvements in support environment. One of that improvements was to introduce Trac and SVN, and the other was the Cloud-ization of support environment.

As already reported in [6], the usability of Kagemai was low for our PBL because it also required inputting data which were beyond the scope of the exercise. There were some cases that students fixed bugs without to report these bugs through Kagemai. Teachers analyzed development status of students from Kagemai and CVS data, so more exact input was important. Therefore we introduced Trac and SVN as a support environment which was more suitable for our PBL. Trac and SVN were prepared for ticket driven development. Under ticket driven development, a ticket is generated for each task which is performed for one product by one developer. Generated tickets are grouped by a unit of development named milestone which corresponds to use cases and so on. By using Trac, students can manage not only reporting bugs but all tasks of the development on the same system. Teachers can also recognize the development status by analyzing these logs, as it will be described in III-C.

In the 5th year, we tried the Cloud-ization of the server running Trac and SVN to manage it easily. We used edubase Cloud[1], one of the Cloud Computing services for education provided by GRACE center in National Institute of Informatics, as an infrastructure of the Cloud-ization. The summary of the new environment is illustrated in Figure 3. Edubase Cloud is based on Eucalyptus, and booting images of OS (machine images) are saved on the S3 (Simple Storage Service) storage. As a characteristic of S3 storage, if the instance of a machine image is stopped, the runtime data is cleared. It is necessary to use EBS (Elastic Block Store) volumes to keep Trac and SVN data persistently. Therefore, we have planned to store constant parts, such as the web server for running Trac and SVN, the main software and plugins of Trac and SVN, and basic configurations of the server, as the machine image on the S3 storage and to keep variable data at runtime, such as Trac project data, SVN repository data, and access logs, persistently by the EBS volumes. We have prepared an EBS volume and an Elastic IP address for each team, and an EBS volume and an IP address have been always assigned to an instance in the pair. Even if the server instance was changed, the students could access the team data by the always same URL.

It became more easy to supply the server running Trac and SVN for each team independently by the Cloud-ization. The operations needed for booting two or more servers which had similar settings only copying the machine image and changing a few configurations. When the server trouble occurred, we could minimize the extent of the impact. The I/O error of the file system has occurred in the server of a team as an actual trouble, we have tried to restore the server

[1]http://edubase.jp/cloud/

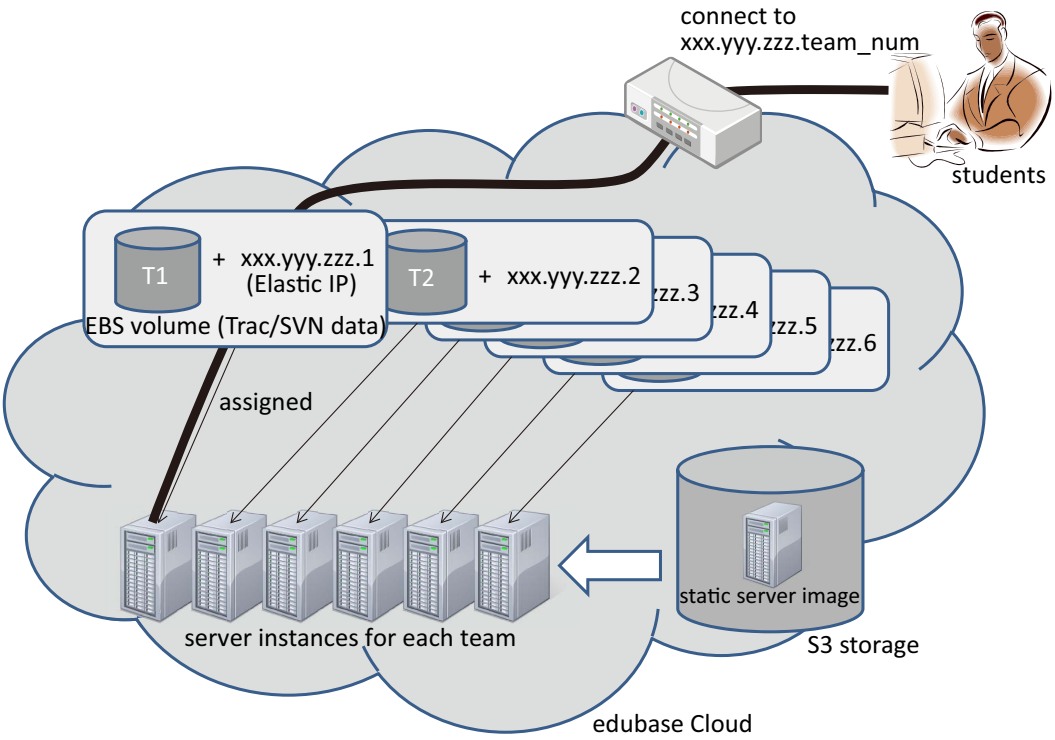

Figure 3. Cloud-ization of the development support environment using edubase Cloud

by a new instance. We have assigned the copy of the EBS volume and the Elastic IP address which was assigned to the troubled instance to the new instance. By this operation, we have been able to restore the same environment as before the trouble for students in about 12 minutes from recognition (about 50 minutes from occurrence) of that.

In addition, we have prepared two systems against other troubles. One is a server for duplicating data on EBS volumes using rsync, a tool for synchronizing file systems over the network. The other is a system that checks aliveness of web servers providing Trac and SVN by trying to connect to the servers with HTTP. When the trouble that there is no response from the server occurs, it sends an email of a warning message for us. Fortunately, any troubles which these preparations commit effectively did not occur in our experience.

C. Teaching Support Environment

In PBL style exercise, it is important that the teacher behaves as the facilitator whose role is the adviser rather than the instructor. Appropriate advice from facilitators to students is considered vital to enhance the value of PBL. Of course the effectiveness is the same with software development PBL. What needs to be done to give students specific advice is correct understanding of team's status not only about development products but also about process by facilitators. Through by two data, one is history of activity stored in the task management tool and the other is changing log of the source repository, it is possible to get general status of process and products. Put another way, it may be possible for facilitator to give the advice on real-time using these data. However, it is difficult to grasp problems on development from raw log data of resources varying with time, such as a task management tool or a source code repository, for anyone except for an expert.

Against this background, a novel Gantt chart tool getting development status easily was developed in the 3rd year. Unfortunately, scale marks of existing Gantt chart tools are usually calibrated by a day or an hour. It is not enough grain size for facilitators to understand development status on software development PBL. A significant advantage of our tool against the existing is not miss a few changing of temporal variation because of variability of time grain size.

Using this tool, our recognition process of student's status became much easier. Although some features are required by facilitators through the practical use of the facilitation on the PBL. With the increasing of demands, additional features were implemented for our tool in the 5th year. For example, alarm function by highlighting problem point on the Gantt chart and instinctive visualization of team's state by graphs. Figure 4 shows output of the tool. Upper left side table means task state of each member, right side illustrates rate

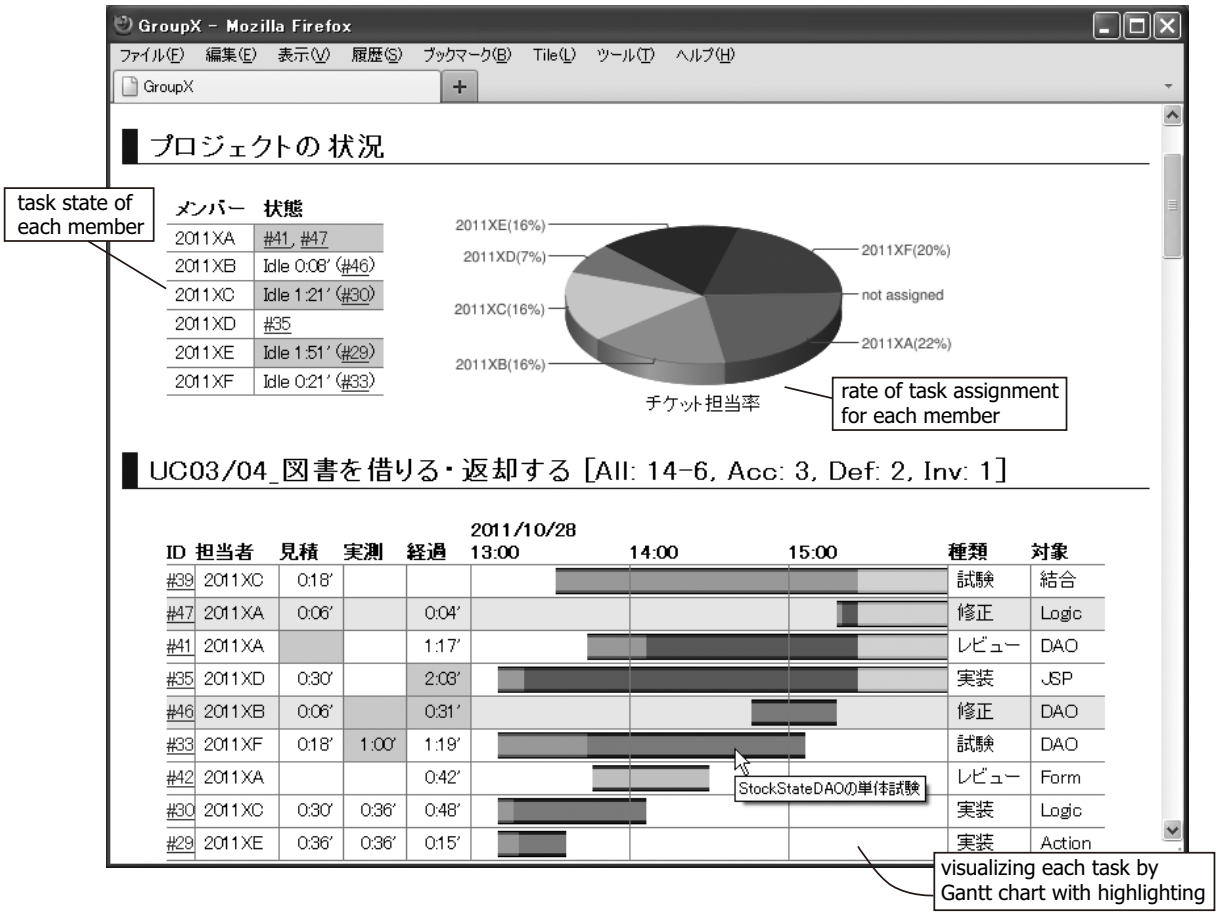

Figure 4. Output of the visualization tool

of task assignment for each member and bellow one is a Gantt chart with highlighting.

Our tool was greatly helpful to grasp development status but the problem of data accuracy was exposed. Ergographic data taken from task management tool are deeply depend student's input. Perhaps, students may make declarations an overclaim or underclaim to avoid low marks. In other words, the accuracy of these data is questionable. So we need to think of effects of the non accurate data.

IV. Future Work

In this section, we explain some future works about each sub environment.

For making improvements of the development environment, we plan to use edubase Cloud as a DaaS (Desktop as a Service) provider. DaaS is a service that the user can access a virtual desktop terminal via the Internet. There are some merits for both students and teachers by using DaaS. These merits are

- the hardware performance required for students' laptop PC is decreased, because they can develop the system as long as the PC connects with VPN via the Internet,
- the common development environment can be prepared independently of students' laptop PC, and
- there are no need for teachers to distribute the huge (more than few GBytes) image of virtual machine.

However, using DaaS has a high dependence on the network environment. If there is no connection to the Internet, the development is impossible. When all students gather in a room and exercise, they will use one network environment and the network load will be high. We have to clear the network problems.

The current problem of the support environment is that cooperation between planning of project management and inputting data to Trac is impossible. If students plan for project management by using some tools, since these are independent from Trac by us, they might have to maintain the relation between Trac and these tools manually. There are some methodologies to make cooperation between a plan of project management, such as WBS (Work Breakdown Structure), and a task management system, but introducing of specific environment may restrict students' ideas for

carrying the project. The future work is to make cooperation between the plans by some WBS-like tools and the tasks managed by Trac without restricting students' ideas.

For the teaching support environment, what we would like to point out is how to feedback the information getting from our tool. From educational ideal point of view, student's spontaneous understanding is desired but it is difficult in practice. Hence, if the output result of the tool was provided to students, their motivation might become higher. However, this action defeat the purpose of PBL "think and act for oneself", facilitator's decision of those presentation style affects students' understanding. So facilitator should decide the presentation style carefully.

V. Conclusion

Experiments of the software development PBL in IT Spiral focused on environments improvement have been introduced. For the development environment, we introduced a virtualization technology to give a non-stressful development environment to students. For the development support environment, ticket driven development style was adopted by using a system of task management tool. For teaching support environment, we developed a project visualization tool to recognize process of student's status easily.

Throughout 5 years trial, we had been applied some improvements as shown in section III to the software development PBL. Our improvements reduce students' complaints on usability of development environment and hardware performance. Students could concentrate on the essential part of the exercise, that is to improve their way of project management. On the other hand, we encountered new challenges. These points should be investigated further.

Acknowledgment

We thank participating universities and cooperative companies for the held of IT Spiral and practical software development exercise, and we are grateful to the successive students of this programs.

References

[1] Keidanren (Japan Business Federation), "Towards strengthening development of human resources in advanced information and communication technology by industry-academia-government collaboration," 2005, (in Japanese). [Online]. Available: http://www.keidanren.or.jp/japanese/policy/2005/039/honbun.pdf

[2] B. J. S. Barron, D. L. Schwartz, N. J. Vye, A. Moore, A. Petrosino, L. Zech, and J. D. Bransford, "Doing with understanding: Lessons from research on problem- and project-based learning," *The Journal of the Learning Sciences*, vol. 7, no. 3/4, pp. 271–311, 1998.

[3] Y. Matsuzawa and H. Ohiwa, "A result of trial education for software engineers through university-industry collaboration and project-based learning," *Journal of Information Processing Society of Japan*, vol. 48, no. 8, pp. 2767–2780, 2007, (in Japanese).

[4] T. Hananoi, G. Arita, S. Sawada, K. Ushijima, K. Yoshimoto, and K. Makizono, "A bi-directional practical education through industry-university cooperation," *Journal of Information Processing Society of Japan*, vol. 48, no. 2, pp. 832–845, 2007, (in Japanese).

[5] A. Sawada, T. Kobayashi, N. Kaneko, N. Nakamichi, H. Ohkubo, and S. Yamamoto, "A project based learning using airship control software development," *Journal of Information Processing Society of Japan*, vol. 50, no. 11, pp. 2677–2689, 2009, (in Japanese).

[6] H. Igaki, K. Kakimoto, S. Saiki, N. Fukuyasu, S. Kawaguchi, Y. Hayase, N. Sakiyama, and K. Inoue, "Process monitoring environment for reality-based software development practice based on comparison among student groups," *Educational technology research*, vol. 34, no. 1·2, pp. 95–105, 2011.

[7] M. Barker and K. Inoue, "IT SPIRAL: a case study in scalable software engineering education," in *Proc. 22nd IEEE-CS Conference on Software Engineering Education and Training*, 2009, pp. 53–60.