Software development PBL focusing on communication using Scrum

Satoru Kizaki
Graduate School of Information Systems,
The University of Electro-Communications
Chofu-shi, Tokyo Japan
Email: kizaki@ohsuga.is.uec.ac.jp

Yasuyuki Tahara
Graduate School of Information Systems,
The University of Electro-Communications
Chofu-shi, Tokyo Japan
Email: tahara@is.uec.ac.jp

Akihiko Ohsuga
Graduate School of Information Systems,
The University of Electro-Communications
Chofu-shi, Tokyo Japan
Email: ohsuga@uec.ac.jp

***Abstract*— In this paper, we improved the PBL (Project-Based Learning) environment aiming at the collaborative-software development education by industry-university cooperation. In the fiscal year 2011, the agile technique adopted as the software development process, and the small-scale project was tried. However, there were problems like the shortage of communication between members, the customers and difference of a member's technical capabilities and the burden was placed to the individual. In the fiscal year 2012/2013, in order to solve these problems, the development process "Scrum" was adopted. We analyzed the result of projects. In the result, Scrum was being able to perform problem solving in which the member's diversity was harnessed. It could carry out improved relations as the team by the positive participation in customer's projects. And, it could build the communication base by "making the place team". Therefore, we understood using Scrum for success of projects.**

Keywords - Scrum; agile software development; Project-based Learning(PBL) .

I. INTRODUCTION

In recent years, Project-Based Learning (PBL) is attracting attentions and many courses are carried out. In a PBL course, students complete a project in a similar way as in actual business. The course improves skills and knowledge of the students that are useful in the actual society. Also in information system educational institutions, the number of examples of courses carried out as PBL is also increasing [1, 2, 3].

We have conducted university-industry collaboration project "Collaborative Management Laboratory", intended for undergraduate students of Keio University in the fiscal year 2011 [4]. The attempt was to implement a training course where both undergraduate students and engineers in industry collaboratively learn how to develop information systems through project-based learning. It started in the fiscal year 2005.

Each project involves three to five students and a project manager from an IT company. There were companies or personal clients that actually needed the developed systems in this project. We aimed at the development of software for general consumers. The project was aimed at the information systems engineers' training through the PBL execution from the educational perspective; however, it is required that the results are put out efficiently within a short term (about four months) as the development period.

In the fiscal year 2010, non-agile development processes such as waterfall types were used. In order to correspond to the change of the actual community, an agile development method was adopted after the fiscal year 2011 [5].

We are currently carrying out education in an agile software development method by this project. In Japan, software development methods are changing from waterfall types to agile types. Industrial communities are preparing frameworks for agile software development in Japan. However, educational communities are not ready for them. There are also cases where each educational institution individually carries out agile software development PBL. However, the software quality of the product created by the students is low. Therefore, they cannot obtain sufficient customer satisfaction.

To the cause, there were problems like the shortage of communication between members, the customers and difference of a member's technical capabilities and the burden was placed to the individual. In the fiscal year 2012/2013, in order to solve these problems, the development process "Scrum" was adopted. We analyzed the result of projects. In the result, Scrum was being able to perform problem solving in which the member's diversity was harnessed. It could carry out improved relations as the team by the positive participation in customer's projects. And, it could build the communication base by "making the place team". Therefore, we understood using Scrum for success of projects.

This paper reports the knowledge, which carried out the agile type development process "Scrum" through the project in the fiscal year 2012/2013.

The remainder of this paper is organized as below: Section 2 describes the agile development process of SCRUM and related ideas. Section 3 explains the practices in agile software development. In addition, we describe problems when applying the practices to software development PBLs. Section 4 describes the related work. Section 5 presents the trial and the results of our agile PBL. The agile software development PBL "Collaborative Management Laboratory" is described. Section 6 describes trial of Agile PBL (The fiscal year 2011). Section 7 describes trial of Scrum PBL (The fiscal year 2012/2013). Finally, we present our conclusions and future works.

II. AGILE SOFTWARE DEVELOPMENT

According to an investigation of the U.S. Forrester Research company in 2010 [6], adoption of the agile development in the U.S. is 35%, and has far exceeded 13% of the waterfall's type adoption rate. Waterfall model had been a major development style until 1990s. However, this model does not fit small-scale projects in recent years.

Moreover, since development costs, operation and maintenance costs with changes of design change are high, the agile development method failed in order to implement functions required by customers and deal with changes flexibly. For example, eXtreme Programming (XP) [7], Scrum [8], etc. devised by agile development methods.

Scrum is the agile development method, which has spread in most of Europe and America etc. It is based on the research on product development in Japan conducted by Takeuchi and Nonaka [9], and was systematized as a software development process by Sutherland [10]. The Scrum has focused on side of project management so that a team can work together. The member's role consists of a "Product Owner" who decides requirements and a priority as a person in charge, a "Scrum Master" which gives support so that a project may progress smoothly, and a "Team" which is the developer's group. After that, how to proceed depends on the functional unit to which a Product Owner called the "Product backlog". A function is described in story form. In addition they define priority and completion to each function. Product backlog has to include all the information needed in development of all the released products.

Secondly, a scrum master and a team probe (the work items (tasks) carried out in order to clarify the function of a product backlog) create a "Sprint backlog". A sprint backlog details of the function for which the product owner makes his decision. The contents are decided in a sprint plan meeting. The length of each sprints are constant and makes the product which the can be shipped by one sprint. Moreover, daily work begins from the daily-scrum. With daily-scrum, the work report and problem sharing are a done by all members of a team.

We work deciding on fixed time called a time-box. When work is done, we perform retrospective thinking with KPT methods [11]. The KPT method is acronym of Keep, Problem, and Try. "Keep" is good things, "Problem" is problem points and "Try" is improvement plans. In agile development, this method was born in Japan in 2005, in order to solve various problems of teams.

III. PROBLEMS OF CARRYING OUT PBL IN PRACTICE

With agile development methods, we suppose that the effect would be seen by applying them with practice is a style of the development effect accepted to be based on experience.

Although there is no defined practice, we expected the following effects in Scrum introduced by the previous session. However, when we carried out PBL, the problem described in the next section occurred.

A. Test-Driven Development

Test-driven development (TDD) is a software development process that relies on the repetition of a very short development cycle [12]. At first, the developer writes an automated test case that defines the desired improvement or new function, produces the minimum amount of code to pass that test, and finally refracts the new code to acceptable standards. Moreover, there are problems that there is no telling what should just be tested in the case of the student unfamiliar to programming.

B. Pair Programming

Pair programming is an agile software development technique in which two programmers work together at one workstation [13]. One, the driver, writes code while the other, the observer (navigator), reviews each line of code as it is typed in. The two programmers switch roles frequently.

However, in the case of the distributed development environment, it cannot develop together. Research by remote pair programming is done in the existing research [14]. There was a team, which took in pair-programming also, in the project of the 2011 fiscal year. One student had a report, as "I am inexperienced in programming however when things are done for pair-programming with a member, the problems is founded immediately. I became a new coder". However, when both experiences are insufficient, a navigator cannot do exact advice, either and cannot expect an education effect. Moreover, time will be restricted to work within a coursework.

C. Continuous Integration

Continuous Integration (CI) is the practice; in software engineering; of merging all developer workspaces with a shared mainline several times a day. It was first named and proposed as part of eXtreme Programming (XP) [15]. CI is a software development practice where members of a team integrate their work frequently; usually each person integrates at least daily leading to multiple integrations per day.

Many teams find that this approach leads to significantly reduced integration problems and allows a team to develop cohesive software more rapidly. The tool, which realizes CI, is famous for Jenkins [16]. However, there are also problems of costing early introductory costs.

D. Code Refactoring

Code refactoring is a "disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior" undertaken in order to improve some of the nonfunctional attributes of the software. Advantages include improved code readability and reduced complexity to improve the maintainability of the source code, as well as a more expressive internal architecture or object model to improve extensibility [17].

Code refactoring fixed the inner structure of source codes, without changing operation of programs. Code refactoring corrects codes by a periodical code review based on indication of the others. However, Refactoring is difficult to carry out a code review periodically by agile one for the below-mentioned reason.

IV. COLLABORATIVE MANAGEMENT LABORATORY

A. Related Work

In the U.S., which is an agile development's advanced nation, agile development is taken in software development education.

At Fort Lewis College, there is a strong emphasis on the liberal arts and the role of service learning in education [18]. In the computer science curriculum, the software engineering course is a natural place to include service learning because of its project-based focus and emphasis on the complete software development lifecycle. Students get to work on real projects rather than throwaway toy projects devised by the professor. This leads to greater student motivation and involvement with the projects. They gain experience working with clients. This gives them valuable real-world experience that is impossible to replicate in the classroom.

At Carnegie Mellon University Silicon Valley, students start their master's program with the Foundations of Software Engineering course. This course is team-based, project-based, and mentored. Ji et al observed and presented the data from five years of 50 teams developing the same project each year and the effects of transitioning from Waterfall to Extreme Programming [19]. The characteristics between these two methods were evaluated and compared. Waterfall teams spent more time creating high ceremony documents where as Extreme Programming teams spent more time writing code and documenting their design in their code.

B. Agile Software Development Education

The agile software development PBL "Collaborative Management Laboratory" is described. This coursework has adopted the agile development method "Scrum". The project model (Fig. 1) of this course requires, the customer (product owner) to be involved in the project. Moreover, the project was composed of three to five students and the scrum masters from IT companies. In addition an agile specialist (Agile coach [20]) participates besides a teacher, and it supports unfamiliar teams to methods. The teachers performed technical assistance other than agile one. The agile coaches offered introductory support of the Scrum frameworks.

By carrying out this model to a coursework, the students can learn in the environment near to actual software development.

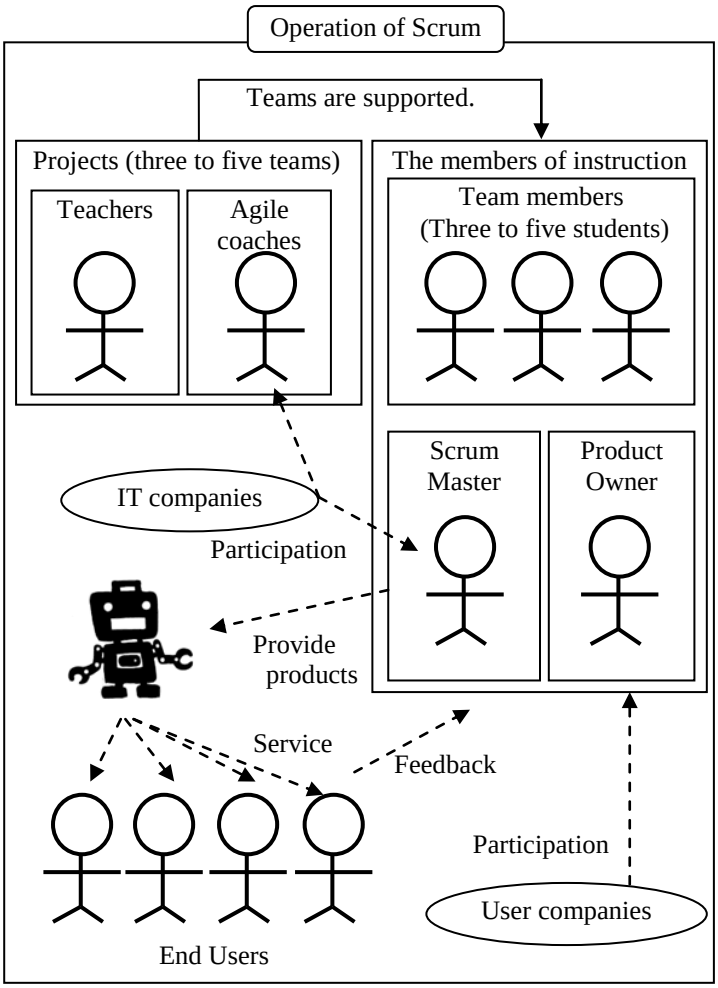

Fig. 1. Model of a project in the proposed PBL system.

In the project in the fiscal year 2011, the professionals of agile development joined the members of instruction. They performed a lecture on agile development, and they taught how to use BTS (Bug Tracking System) at the beginning of class. Furthermore, the IT engineer, working members of society, joined the members of each project as Project Leader (PL). They performed technical support for each student.

In the fiscal year 2012, 20 students took part in project. Prior to starting the project, they understood about agile software development. The knowledge of Scrum was taken in from this year.

In the fiscal year 2013, 12 students took part in project. We carried out based on our experience in the fiscal year 2012. The students understood about agile software development.

C. Criterion for Evaluating of Project on Agile PBL

In agile software development, to produce software with business value high for customers desired.

Therefore, the criterion for evaluating on agile PBL becomes the point whether the product with high business value could be made. A project is failure, if customer satisfaction cannot be obtained even if they are able to make products. However, since the students experience in actual business does not have ideas about products with this high quality, there are cases where projects fail.

According to the Kano Model [21], which is a model for understanding a customer's liking objectives, there are two kinds of quality, "must-be quality" and "attractive quality (quality exceeding expectation)".

The developers have to deliver products without defects. It is meant as must-be quality. Customers think it natural that the product satisfies the quality. In addition, customers have dissatisfaction, if must-be quality is not offered. We are required to provide customers with added value (offer attractive quality).

So, if the quality of product is high, customer's comprehensive degree of satisfaction can also be raised. The valuation method of the customer satisfaction using the Kano Model is described by the proposal technique.

V. CASE STUDY AGILE PBL

A. The project in the fiscal year 2011

In the project in the fiscal year 2011, the professionals of agile development joined the members of instruction. They performed a lecture on agile development, and they taught how to use BTS (Bug Tracking System) at the beginning of class. Furthermore, the IT engineer, working members of society, joined the members of each project as Project Leader (PL). They performed technical support for each student.

Moreover, an agile software development process and affinity adopted ticket-driven development [22] as the raising-visualization and working efficiency of work purpose. Ticket-driven development is a development style which manages by dividing work into tasks and assigning tickets from BTS (Bug Tracking System).

To initiate, a ticket is first published. By using a ticket disagreement of communications can be reduced. By adopting ticket-driven development, information is sharable by the persons concerned, and the role for every worker in charge is also clear, and we can equalize work load, and we can understand who is in what kind of situation.

B. Problems

The workload allotment begins with the PL's registering tasks into BTS (Bug Tracking System). Although the BTS is a slightly difficult system to learn, once it understands the process was considerably easier.

At the start of this project, the agile coach instructed the members on how to use the BTS. Therefore the ticket management could well receive. The PM and students were able to understand their tasks using tickets. However, this tool was hardly used by the students. Because the students did not use the tool usually, it was difficult to introduce the tool into the projects.

C. Skill Questionnaire

In the project in the fiscal year 2011, the creation of a program, completion, and delivery of goods, which realized "The next-generation mobile network service", was carried out [23]. We are creating a program that uses Robot Service Network Protocol (RSNP) that is robot service-oriented protocol specification.

Then the skill questionnaire and the quiz were carried out on the team, which started the project (Table I).

As a result, it turned out that the student A's skill was high and he has experiences of application development. The student B found that programming skill was low. Although the student C did not have experiences of tools, it turned out that he has high potential of programming.

TABLE I. THE SKILL QUESTIONNAIRE BEFORE BEGINNING PROJECT

Questions	**Students**		
	A	*B*	***C***
Programming was learned by the lesson.	YES	YES	YES
Fundamental grammars are known.	YES	No	YES
Eclipse can be used.	YES	No	No
Easy applications can be made.	YES	No	No
Android applications can be made.	YES	No	No
Java programming quiz	71.4%	28.5%	85.7%

D. Training and Code Reviews

The results of skill questionnaires revealed the variety of the students' abilities. Therefore, the student trained programming of Android application. The student B learned from the fundamental grammar of Java.

The student C learned about Android application development. The writer also understood that has effect of learning about training and code reviews from post-project reports. He reported *"I was able to learn the indispensable basic knowledge for Android development. Moreover, as a result of the code review, I studied how to write a code legible for the others"*. This report demonstrates the effect of code reviews.

Since students do not have experience of getting others to read the code written by them, they can expect an educational effect. Moreover, the writer understood that have some efficacy in training.

On the other hand, the student B who spent much time on training reported *"My technology is low. Therefore, the team has been made to influence. And it has developed into hindrance. Furthermore, huge time is needed although I worked mastering technology. After all, I hindered the team and the project finished me"*.

Read in this point, the student C had the nature of programming and was able to work effectively by short training. However, even if the student B carried out to training, he needed huge time. In addition he found that it was difficult to perform effective education in the period and time when it opted for the coursework.

Moreover, the student's B motivation has also fallen because he assigns not actual development but training. When experienced in a certain amount of programming from this result, training is effective, and for the student of a beginner level who learned the syntax and algorithm of programming by the lesson, it turned out that an effect could seldom be expected by training about programming education. However, the student B reported, *"I worried alone and time passed over me. Therefore, a writer had to ask early those who understand and to do work"*.

In this project, each member needed to fully taken the communication in a team.

VI. CASE STUDY SCRUM PBL

A. The project in the fiscal year 2012

In the fiscal year 2012, 20 students took part in project. Prior to starting the project, they understood about agile software development. Therefore, the lecture (the 2nd time of a lesson, the 3rd time) about environment for agile software development or collaboration tools (GitHub et al.) was performed.

Before using methodology and tools, the teachers, member-of-society engineers, and students who participate in projects actually need to understand the directions for these theories and tools. The lecture was agile coach's role. They participated in the weekly lesson. Agile coaches were spread having knowledge. Therefore, these became possible to utilize methodology and tools effectively.

The projects were announced by each product owner (PO) at the 3rd lesson. There was each plan from the issue resolution of companies to the personal request of smart phone application development. The students took part in the interesting project. The projects were five projects. As for each team, 4 to 5 students gathered. Furthermore, the teacher introduced the member-of-society engineer who becomes the scrum master (SM). Each team chose the scrum master. The project A, the project C, and the project E created smart phone-oriented application. Moreover, the project B developed Facebook application. The project D developed software corresponding to iPod.

Moreover, arrangement of the desk was not made facing each other before the fiscal year 2012. They concentrated only on one's jobs. And, each student could not take communication.

Arrangement of the desk of a classroom was used as the "island type" (Fig. 2) in the fiscal year 2012. An "island type" packs a desk into a group and refers to the form arranged in every lump (island).

By using this arrangement, there were effects, which communicate actively because students adjoin each other.

Figure 2. Arrangement of the desk of "island type".

B. Participation of the agile coaches

The agile coaches are specialists who are offering agile introductory support.

According to reports from Yahoo, the agile coaches can make a significant contribution. In this study scrum teams without coaching support increased their productivity by 35%, while those with agile coach support recorded 300% or greater improvement.

C. Use of the Distributed Version Control System

GitHub [24] is a tool (remote repository) for supporting the distributed version control system Git [25]. The project in the fiscal year 2012, version management of the resource is performed using it. Git can be worked in a local environment, even if you cannot access to Internet.

However, there are many students not using them. Therefore, the agile coaches performed training of them in the 2nd time of the classwork. As a result, all the students who have participated in the classwork were able to use them.

As of February 2013, those are used in all the projects. The students have to manage it, in order to do collaborative work between members. It is used as shown in Fig. 3.

1) The workflow of this technique is shown below:

Step 1: The student's leader creates remote repository of project to GitHub.

Step 2: The students do the clone (copy) of the project products from the remote repository.

Step 3: The students update or create files (e.g. source code etc.). Then, the local repository is committed in the updated files.

2) Notes are shown below:

a) When other students have updated the remote repository, they have to pull the updated files.

b) The agile coaches and a scrum master support a student when she is in trouble in the operation of the tools.

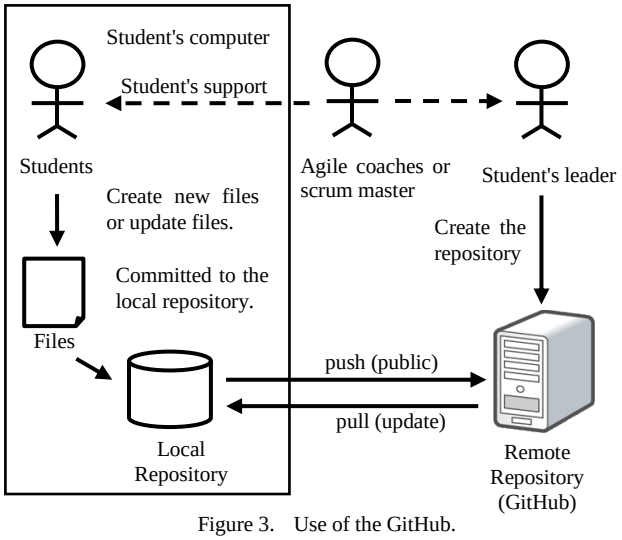

Figure 3. Use of the GitHub.

D. The iteration cycle of projects

Each project carried out the iteration cycle (sprint period) of Scrum per two weeks. This lesson is the subject of 2 top continuations at one week. After the lesson of 3rd time, they worked at jobs of each projects. Project members were sitting next to each other at a desk. Then we could set up a comfortable environment for communication.

We determined what the first week would do in the sprint period by holding the sprint-planned meeting (We carried out once in two weeks after that). In the weekly lesson, daily scrum (scrum meeting) was performed first. Students shared the following things within the team in the first 15 minutes.

a) Work done before last time
b) Work to be done by next time
c) Trouble

Next, with the time box, they examined the ideas using brainstorming; and they furthered development of the team. The students carried out a meeting (Retrospective) in 15 minutes of the last of the lesson using the KPT method etc. And they examined the improvement proposal about project advance using the technique.

The scrum masters participated in the meeting in the lesson. They performed facilitate of the subject for discussion. The scrum masters did not participate in actual work, but they assisted the team so that students could develop smoothly. The students demonstrated the new products.

Then the product owner (client) evaluated the products at the end of a sprint period. After evaluation of product owner, the retrospective was conducted by the project led by scrum masters.

E. Projects introduction in the fiscal year 2012

The project A made the smart phone application that called "SAMISHINBO" (Fig. 4). This application finds simply the friend who wants to make the same the thing (for example, I go to karaoke), which we want to do with a smart phone. Development was divided into the development group and the designer group. They shared information for application development.

The project B developed Facebook application. This product owner wanted to collect "Like" (mainly for overseas users). He has requested to Facebook-application used the Facebook page of the comics synthesis community site "MANGAREBORN" was developed at his company. ("MANGAREBORN" is building a worldwide community of manga fans in support of the manga artists.)

In the case of this project, I received sample application and a petition from the time of development, and the development image stuck. Each skill was checked first the early stages of development and the insufficient portion were the plan to compensate with studying.

The project C made the smart phone application. This project created Android 4.1 application (Mounting of the Twitter client aiming at the anime actual condition, and the Web platforms).

However, they were not able to complete to the last. Moreover, since the earnings model was imperfect, the product for which the user is asking was not able to be made. Students continue to show the intention which continues development, and although it was difficult, they want to expect the future result in the lesson period.

The project D developed software corresponding to iPod. This project created iPod application using the Unicage software development method.

“Unicage software development method” is a unique development approach composed of simple UNIX command scripts and text files. This product used for the spectator of J. League looking at various data in the stadium.

The project E created application for search and peruse the report updated on the website from a smart phone.

Specifically, search / inspection function of the cooking recipe report which uses the goods of a client was created.

Figure 4. The iPhone screen of “SAMISHINBO”.

F. The useful communication tool

Comparison of communication summarizes on the table 2. The useful communication tool is Facebook. This tool is an online social networking directory that connects people with friends and others who work, study and live around them. People use Facebook to keep up with friends, upload an unlimited number of photos, share links and videos, and learn more about the people they meet.

There are multiple advantages of Facebook. The first, Facebook have mass appeal of implementation tools. The second become accessible to more people. The 3rd carries out group management, and is that only the fixed member can see. For this reason, the thing on which it advises actively within the project is expectable.

TABLE II. USE OF COMMUNICATION TOOLS

Communication tools	**Teacher**	**Student**	**SM/PO**
Mail	Using	Using	Using
LINE	Not using	Using	Seldom using
Skype	Using	Seldom using	Using
Facebook	**Using**	**Using**	**Using**

G. Review Part Prioritization by the Kano Model

The priorities of review parts are determined by the Kano Model. The reason of the adoption of this model is that it can decide the priorities of requirements (features) without owner's subjectivity and speculative omission. The Kano Model asks fivefold choice question about the cases if functionality in question is not realized (dysfunction form of question) and cases if the functionality in question is realized (functional form of question). After that, this technique identifies prioritization from an evaluation two-dimensional table (Table 3).

1) *The reply of question is as follows:*
   a) *I like it.(Like)*
   b) *I expect it.(Expect)*
   c) *I'm neutral.(Nutral)*
   d) *I can tolerate it. (Like with)*
   e) *I dislike it. (Dislike)*

For example, when a functional question is "I expect it." and a dysfunctional question is "I dislike it", "Mandatory" selects as a priority from the evaluation two-dimensional table. "Mandatory" means an indispensable feature.

Moreover, "Linear" is indispensable. And more this type is realized, more the product can surpass the customer's expectation. "Exciter" is a point of differentiation. "Questionable" indicates the feature is questionable. If there is a "Reverse" feature, it would lead to troubles. "Indifferent" expresses the user does not care the feature.

TABLE III. THE TWO-DIMENSIONAL TABLE IN THE KANO MODEL

functional questions \ dysfunctional questions	1	2	3	4	5
1	Q	E	E	E	L
2	R	I	I	I	M
3	R	I	I	I	M
4	R	I	I	I	M
5	R	R	R	R	Q

M: Mandatory
L: Linear
E: Exciter
Q: Questionable
R: Reverse
I: Indifferent

H. Priority of Code Reviews

The code review is carried out from the function (feature) whose determined priority is high. The priority of the code review is indicated to the sprint backlog (Table 4). For example, when the function in which evaluation of a priority is "Mandatory" is not filled, customer satisfaction falls. Therefore, it needs to be implemented.

Case Liner or Exciter: When many these features are able to be developed, customer satisfaction fills.

Case Questionable, Reverse or Indifferent: They need not be implemented. Therefore, code reviews of them are not necessary.

TABLE IV. SPRINT BACKLOG

ID	**Sprint backlog**		
	Feature	Priority	Person in charge
#1	Development environments are built.	-	All of the members
#2	Web servers are built.	-	Student A
#3	The new function A is made.	Mandatory	Student B
#4	A function is added to the function B.	Liner	Student C
#5	A function is added to the function C.	Exciter	Student D

I. Customer Satisfaction

The following formula defines the customer satisfaction in the agile software development PBL. Usually customer satisfaction has been evaluated through questionnaire. Our proposal evaluates the customer satisfaction as the degree of the functionality of software quality characteristics. The customer satisfaction is calculated by the following division. The denominator is the total of features that are identified from the required items of the product backlog created by the products of each feature and each progress ratio. Finally, the rate (%) of the realized features is calculated (Fig. 5).

$$CS = \frac{\sum_{i=1}^{n}(\alpha_i \times P_i)}{\sum_{i=1}^{n}\alpha_i} \times 100$$

Customer satisfaction: CS
Object feature : α_i
Number of object features : n
Progress ratio : P_i

Figure 5. Customer satisfaction.

J. *The communication base by "making the place team"*

In this project, the scrum master has participated in each project. They do the facilitator role which makes the place team. "Making the place team" is giving a member's advice or removing problem for the teams.

VII. CONCLUSIONS AND FUTURE WORKS

In this paper, we improved the PBL (Project-Based Learning) environment aiming at the collaborative-software development education by industry-university cooperation. In the fiscal year 2011, the agile technique adopted as the software development process, and the small-scale project was tried. However, there were problems like the shortage of communication between members, the customers and difference of a member's technical capabilities and the burden was placed to the individual. In the fiscal year 2012/2013, in order to solve these problems, the development process "Scrum" was adopted.

We analyzed the result of projects. In the result, Scrum was being able to perform problem solving in which the member's diversity was harnessed. It could carry out improved relations as the team by the positive participation in customer's projects. And, it could build the communication base by "making the place team". Therefore, we understood using Scrum for success of projects.

This result of research is going to be evaluated on the basis of operation data after the completion of projects in February, 2014. We are going to analyze operation data (e.g. the developer's argument, the meta-data of source codes, etc.) from now on using the repository mining technology [26] in which research is advancing in recent years.

ACKNOWLEDGMENT

This research was done in the industry-university cooperation project "Collaborative Management Laboratory" in Keio University. I deliver powerful statement of appreciation for Associate Professor.Yoshihide Chubachi of an Advanced Institute of Industrial Technology (AIIT), Part-time Lecturer.Ken Okada of Keio University, agile coaches, and the undergraduate students of Keio University who participated in the projects.

[1] Davenport, D, "Experience Using a Project-Based Approach in an Introductory Programming Course", IEEE Trans. Education, Vol.43, No.4, 2000, pp.443–448.

[2] N.Nitta, Y.Takemura & I.Kume, "A practice of collaborative project-based learning for mutual edification between programming skill and artistic craftsmanship". 39th Frontiers in Education Conference (FIE), 2009, pp.1–5.

[3] H.W.A.S. Gondim, A.P.L. Ambrósio & F.M. Costa, "TaskBoard - Using XP to Implement Problem-Based Learning in an Introductory Programming Course". 12th International Conference on Agile Software Development (XP2011), pp.162–175.

[4] Y.Matsuzawa, H.Ohiwa, "Learning Information Systems Engineering and Its Management from Experience of a Tiny Project through University-Industry Collaboration", International Conference on Advanced Learning Technologies 2007 (ICALT), pp.538–542.

[5] S.Kizaki, Y.Chubachi, "Improvement of collaborative work on software development PBL in a distributed environment", Invitation to 3rd International PBL Symposium 2012.

[6] J.Hammond, "Five Ways To Streamline Release Management". Forrester, 2011.

[7] K.Beck, B.Boehm, "Agility through discipline A debate", IEEE Computer, 2003, pp.44–46.

[8] L.Rising, Norman S.Janoff, "The Scrum Software Development Process for Small Teams", IEEE Software, 2000, pp.26–32.

[9] H. Takeuchi, I. Nonaka, "The new new product development game. Harvard business review", Vol. 64, No. 1, 1986, pp.137–146 .

[10] Sutherland, Jeff, "The Power of Scrum. CreateSpace Independent Publishing Platform", 2011.

[11] E.Derby, D.Larsen and K.Schwaber, "Agile Retrospectives: Making Good Teams Great", 2006.

[12] Kent Beck, "Test-Driven Development: By Example. Addison-Wesley Professional", 2002, pp.240.

[13] Williams, Laurie, "Integrating Pair Programming into a Software Development Process", 14th Conference on Software Engineering Education and Training, 2001, pp.27–36.

[14] A.Shaw, "Extending the Pair Programming Pedagogy to Support Remote Collaborations in CS Education", 6th International Conference on Information Technology: New Generations, 2009, pp.1095–1099.

[15] M.Fowler: Countinuous Integration [articles], http://www.martinfowler.com/articles/continuousIntegration.html.

[16] Jenkins, http://jenkins-ci.org/.

[17] M.Fowler, Refactoring Home Page [web site], http://refactoring.com/.

[18] Hanks.B, "Becoming Agile using Service Learning in the Software Engineering Course", Agile Conference (AGILE) 2007, pp.13–17.

[19] Feng.Ji, "Comparing extreme programming and Waterfall project results". Software Engineering Education and Training (CSEE&T), 2011 24th IEEE-CS Conference on, 2011, pp.482–486.

[20] Silva, K., Doss.C: The Growth of an Agile Coach Community at a Fortune 200 Company, Agile Conference (AGILE), 2007, pp.225–228.

[21] Kano Noriaki, Nobuhiku Seraku, Fumio Takahashi, Shinichi Tsuji. "Attractive quality and must-be quality" (in Japanese). Journal of the Japanese Society for Quality Control 14 (2), April 1984, pp.39–48. ISSN 0386-8230.

[22] Sakai, Makoto, "TiDD: Ticket Driven Agile Development Method (in Japanese)", 5th Workshop of Forum on Reliable Computer Software, http://sakaba.cocolog-nifty.com/PDF/TiDD_FORCE09.pdf, 2009.

[23] S.Ushio, Y.Ito, K.Okada, T.Kitahara, H.Tsuji, S.Moriguchi, M.Narita & Y.Kato, "The Digital Travel Diary System Using the Network Service Platform". 2011 Workshops of International Conference on Advanced Information Networking and Applications (WAINA), 2011, pp.890–895.

[24] GitHub, Cooper, Peter (10 April 2008). "GitHub Officially Launches: Git Hosting A-Go-Go!". Ruby Inside.

[25] Hamano, Junio, "[ANNOUNCE] Git v1.8.1.2". git mailing list. Retrieved 2013-01-28.

[26] W.Shang, Z.Ming Jiang, B.Adams, Ahmed E. Hassan, "MapReduce as a General Framework to Support Research in Mining Software Repositories" (MSR), Proc. of MSR2009, 2009, pp.21–30.