Cooperative Learning Method Based On Game Design and Visual Object Oriented Environment to Teach Object Oriented Programming Course

Yulia[1],Rudy Adipranata[2]
Informatics Department
Petra Christian University
Surabaya, Indonesia
[1]yulia@petra.ac.id, [2]rudya@petra.ac.id

***Abstract*— Learning object oriented programming (OOP) is a hard task for many beginning students who have already familiar with procedural language paradigm. Over the last six semesters, only about 59.27% students who take OOP class are passed. To address persistent difficulties mainly imposed by the already known paradigm, we adopted the approach of cooperative learning based on game design with visual programming environment to teach OOP. Using the context of game design, we identify the concept of object to assist students with understanding object oriented design principles.**

We use GameMaker to teach about object concept by creating a game and MinimUML to teach about class design before implementing any code. As the implementation of cooperative learning, the class is divided as some groups.

This method is implemented in second semester of year 2008/2009 in five classes and about 124 students are taken this course. As the result, 80.64% students are passed. It is much better than the previous semesters.

Keywords- GameMaker, object oriented programming, UML

I. INTRODUCTION

Object Oriented Programming (OOP) is a course we offered to the students who have already studied, during their first programming course, procedural programming using C language. The focus of the course is on teaching the basic concepts of the OOP paradigm in C. The course is comprised of both lectures and laboratory activities and is taken by all Informatics Engineering students.

OOP is continuation from Algorithm and Programming (AP) course and becomes base for most of majors in Informatics Engineering Department. Learning OOP is very difficult for many students. Based on data, result of OOP classes from the last six semesters indicates that level of pass of this course is low that is 59.27% (the overall results for the last six semesters are summarized in Table 1). The first lessons in OOP are rich and complex, so many students get confused.

TABLE I. COURSE RESULTS FOR THE LAST SIX SEMESTERS

Semester	Pass		Fail	
	Num of students	Success Rate (%)	Num of students	Fail Rate (%)
Second - 2005/2006	36	65.45%	19	34.55%
First - 2006/2007	68	50.37%	67	49.63%
Second - 2006/2007	126	75.90%	40	24.10%
First - 2007/2008	42	55.26%	34	44.74%
Second - 2007/2008	68	52.71%	61	47.29%
First - 2008/2009	34	48.57%	36	51.43%
Average	59.27%		40.73%	

During this time, from the research to the students who had already takes OOP, we found some reasons that make students had persistent difficulties are: 1) the students had already familiar with procedural programming paradigm, so they difficult to change the paradigm from procedural to object oriented. 2) It is difficult to explain the problems from the student's perspective. 3) It is hard to trace how many times a student commits similar errors and so observe repeating similar problems solving patterns. 4)The lecturer may not know who is having difficulties until it is too late, may not be able to know why the students don't understand, may not be able to convince the students to seek help, and may not have enough time to know the student's need in a large class.

In this research, we had to invent new ways to overcome those problems. We considered issues in science education to overcome those problems, mainly cooperative learning with a visual OOP environment as a tools to describe the object concept.

II. COOPERATIVE LEARNING

Cooperative learning (CL) is the instructional use of structured group learning in the classroom requiring purposeful implementation. Cooperative learning requires students to use acquired knowledge in a group setting with a structured learning task [1].

Cooperative learning techniques have been applied with a wide variety of subject matter and a broad spectrum of populations. Good discussions of cooperative learning methods and research can be found in [2, 3, 4]. Each students of a team is responsible not only for learning what is taught but also for helping team mates learn, thus creating an atmosphere of achievement. Students work through the assignment until all group members successfully understand and complete it.

Research has shown that cooperative learning techniques can: 1) promote student learning and academic achievement. 2) Increase student retention. 3) Enhance student satisfaction with their learning experience. 4) help students develop skills in oral communication. 5) Develop students' social skills. 6) Promote student self-esteem. 7) Help to promote positive race relations

III. LEARNING OOP CONCEPT BASED ON GAME DESIGN

It is easier to teach OOP by using game design, because in a computer game, everything is an object: the monsters, wall segments, coins, bonuses, power-ups, and the guns and bullets. Thinking about creating games means thinking about objects and how they react to one another and to the player's input. So the game creator naturally thinks in an object-oriented way. Overmars used GameMaker to create a game [5], as Figure 1 shows. The designer creates objects. Some objects have a visual representation, such as an animated sprite. Others, like those that control game flow or maintain the score, might lack this feature. Multiple instances of the same object can appear in the game at the same moment. Instances have properties. Some are built-in, like the speed with which the instance moves and the sprite used to represent it. Others can be defined, manipulated, and checked using actions or code. The user must define each object's behavior. While some objects, like wall segments, will have no behavior, others, like the avatar representing the player, will most likely have complicated behavior.

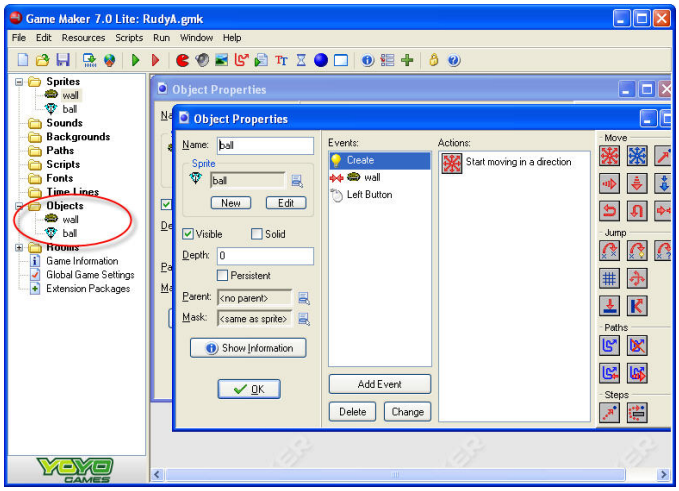

Figure 1. Object Concept using Game Design

IV. VISUAL OOP ENVIRONMENT USING MINIMUML

Writing of program code directly at the time of doing OOP is a difficulty especially when student recognizes the object concept for the first time. This thing is because of object concept adopted from real world having form visually and the relation of between objects can be depicted. While programming of either procedural and or object oriented to apply programming language which in the form of typing of certain words so that difficult to depict it visually. The following source code is an example of a piece of object oriented program code.

```
#ifndef __STUDENT_H
#define __STUDENT_H

class CStudents
  {
   private: //variable declarations
     char _StudentId[9];
     char -StudentName[30];
   public: //method declarations
     CStudents(void); //constructor
     ~CStudents(void); //destructor
     void setStudentId(char id[9]);
     void setStudentName(char name[30]);
     char* getStudentId();
     char* getStudentName();
  };
#endif
```

To overcome the thing, has been developed some application visual programming environment to assist programming in visual form and program code that can be yielded automatically by the application. For development of program in C++ language, Turner [6] develops minimUML, that is an visual programming environment by using UML (Unified Modeling Language) [7]. MinimUML has ability to yield program code in C++ language based on diagrams made by user. Figure 2 is visual display by using minimUML with the same program case like source code above.

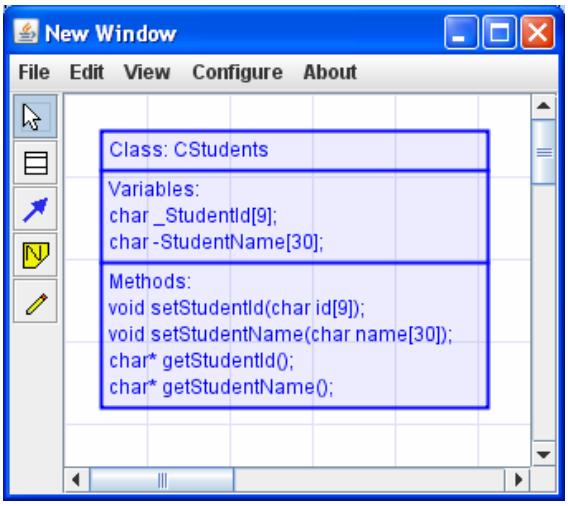

Figure 2. Class Design using MinimUML

V. CLASS STRUCTURE

The structure of the OOP course is based on two 100 minutes sessions per week. As the implementation of cooperative learning, the class is divided as some groups. One group consists of five until six students. Class time is used as follows: 1) lecture class and 2) laboratory activities. For lectures class : 20-25 minutes, group presentation to other groups,. 20-30 minutes, short lecture to present new material, and 40-45 minutes, planned activity. For laboratory activities: 10-15 minutes, preliminary test about previous material, 10-15 minutes, short lecture to present new material, and 60-70 minutes, planned activity.

In the first week, there is a preliminary test about fundamental programming. This test result applied for forming of group of where the members of the group are consisted of student by top-rating, low and medium.

In the early of lecturing, every group will receive a certain matter as according to lecture's time table. Each group will prepare the matter and is obliged to present it at a meeting as according to matter sequence. At the time of presentation, every member of the group must present at each other group. Thereby every member will prepare and there is no member depends on other member. Assessment done by each group receiving presentation.

After the presentation, question and answer, and discussion completed, lecturer does review. Thereby learning process teaches focus at student centered learning, not anymore lecturer as center study. Here lecturer will apply auxiliaries like GameMaker and minimUML.

At the laboratory activity, CL strategy cannot be done by each participant because they will stay at a computer to finalize problems. At this laboratory activity, the student will do some problems directly in front of computer.

Evaluation system consisted of five components that are First Test, Middle Semester Test, Second test, Final Semester Test and Laboratory Activity. First Test was taken from group activity of during doing presentation.

For Middle and Final Test, done self-sportingly by student, where student have to finalize some problems which they have gets before. Middle and Final Test are done directly in front of computer. For second test, done by group, where each group is have to finalize a game problems that is giving from the lecturer. The assessment of laboratory activity is gotten by doing task in each laboratory meeting, where the assessment is individual.

IMPLEMENTATION: TEACHING GAME DESIGN USING GAMEMAKER AND MINIMUML

As a case study for teaching object oriented concept, we used GameMaker as shown in Figure 3 below. Student will learn about the object concept, where an object is not only have data but also function. At the example below is shown how an ball object can be given data like color, measure, form and also some functions like moving, bounds, flows on and what happened when an object is formed (constructor concept) and when an object destroyed (destructor concept)

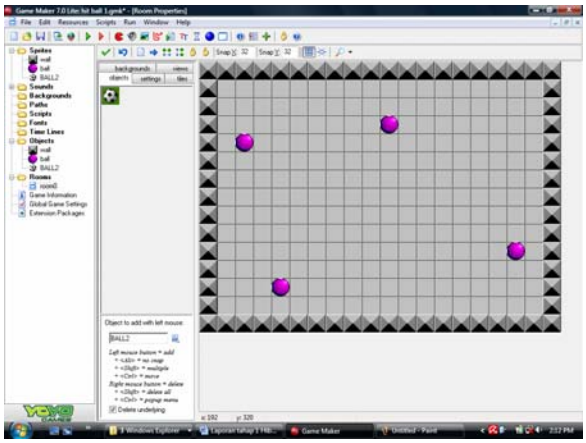

Figure 3. Using GameMaker to Learn OO Concept

As a case study for game design, we use WARCRAFT game [8] as shown in Figure 4 below. Figure 5 is the class diagram design using minimUML for the WARCRAFT game. At this example, we design a base class which called UNIT where this class will be inherited to become class WORKER and FIGHTER. Class FIGHTER will be inherited to become class HERO where in class HERO has object from class ABILITY. The class design using minimUML is easier to be understood by the students because it is submitted visually in the form of class diagrams. Then we can generate the source code in language C from this class diagram design.

Figure 4. War Craft [8]

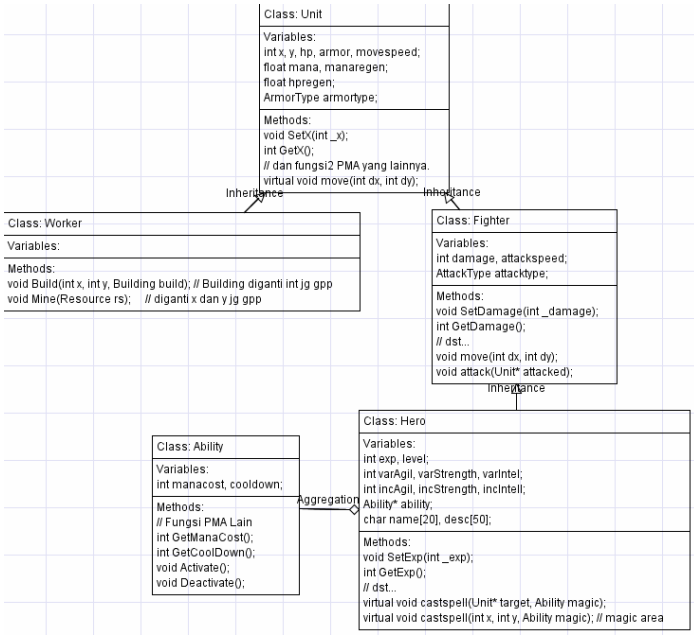

Figure 5. Warcraft Class Design using MinimUML

VI. EVALUATION

This system was implemented in 5 classes with 124 students. The average grade value is 2.49. The average grade value form each class can be seen in Figure 6 and the result of the entire classes can be seen in Table 2.

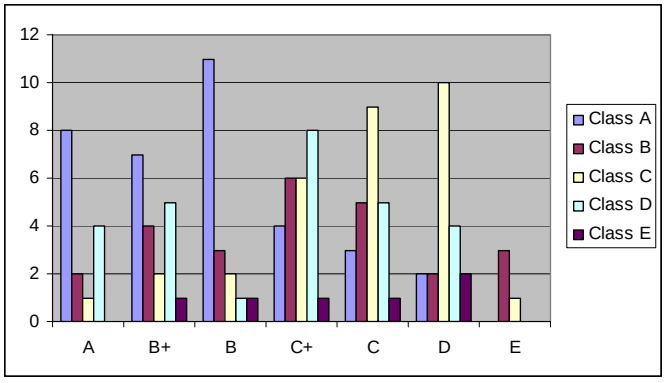

Figure 6. The Grade Value

TABLE II. COURSE RESULT AFTER IMPLEMENT THE NEW SYSTEM

Grade	Number of Students	Percentage
A	15	12,097%
B+	19	15,323%
B	18	14,516%
C+	25	20,161%
C	19	18,548%
D	24	16,129%
E	4	3,226%

From the table above, the total percentage of the passing students (students with grade A, B+, B, C+, and C) are 80.64%. This result is much better than the results before.

From the questionnaire result gived to the students, most of students feel happy to learn OOP with cooperative learning system, applies game and visual design because 1) in a team, they can helping each other, 2) The existence of game makes them more enjoy 3) Easier to understand object concept after seeing design visually.

But there are also some input from student about group presentation because sometimes they don't understand what explained by their friend because the presented student not ready and or difficult to explain. Also because this subject performed in a laboratory, where every student is in front of computer, hence there are difficulty to discuss.

VII. CONCLUSION

Learning object-oriented design and programming is challenging for novices, especially for student who have already familiar with procedural language paradigm. The shift from procedural to object-oriented paradigm is a difficult task. We faced the problems emanating from such a transition and we were guided to a fundamental change in the way we teach introductory object-oriented programming. Many students have learning difficulties which cannot be entirely solved by teachers. To help student learning, we have designed a new method in teaching concept based on game design and visual environment in a team work so that he or she can apply them in design and programming

problems. This method can improve the graduation of student participants from 59.27% become 80.64%.

REFERENCES:

[1] B. Cameron, "Active and Cooperative Learning Strategies for the Economics Classroom," in Teaching Undergraduate Economics: A Handbook for Instructors, William B. Walstad, Phillip Saunders, editors, Boston, London and Toronto: Irwin/McGraw-Hill, 1998

[2] D. W. Johnson. and F. P. Johnson, Joining Together: Group Theory and Group Skills, Prentice-Hall, 1975

[3] S. Sharan, Handbook of Cooperative Learning Methods, Greenwood Press, 1994

[4] R.E. Slavin, Cooperative Learning: Theory, Research, and Practice, 2nd Edition, Prentice Hall, 1995

[5] M. Overmars, "Learning Object-Oriented Design by Creating Games," IEEE Potentials Magazine, Volume: 23, Issue 5, pp 11-13, Dec. 2004-Jan. 2005

[6] S.A, Turner, M.A. Pérez-Quiñones, and S. H. Edwards. "minimUML: a Minimalist Approach to UML Diagramming for Early Computer Science Education," Journal on Educational Resources in Computing (JERIC) Volume 5, Issue 4, December 2005

[7] Turner, Scott A., Manuel A. Pérez-Quiñones, and Stephen H. Edwards. "minimUML: A Minimalist Approach to UML Diagramming for Early Computer Science Education," Journal on Educational Resources in Computing (JERIC) Volume 5, Issue 4, December 2005

[8] http://www.worldofwarcraft.com/