

DESIGNING A PROGRAMMING COURSE TO FOSTER CREATIVITY USING UML MODELING TEMPLATE

Norio Ishii

Aichi Kiwami College of Nursing, Jogan-dori 5-4-1, Ichinomiya, Aichi, Japan

Yuki Nagao

College of Engineering, Chubu University, Matsumoto-cho 1200, Kasugai, Aichi, Japan

Yuri Suzuki, Hironobu Fujiyoshi

College of Engineering, Chubu University, Matsumoto-cho 1200, Kasugai, Aichi, Japan
Research Institute for Information Science, Chubu University, Matsumoto-cho 1200, Kasugai, Aichi, Japan

Takashi Fujii

College of Engineering, Chubu University, Matsumoto-cho 1200, Kasugai, Aichi, Japan

Keywords: Creativity, Programming education, Instructional design, UML, LEGO Mindstorms.

Abstract: Recently, the education program which aimed at raising of the creativity is practiced engineering education. Though the effect of that education is being investigated by the questionnaire, that isn't sufficient as an objective evaluation. So, we propose the UML (Unified Modeling Language) modeling template that aimed at the use to the evaluation of the learning result in the creativity education. Then, the education that a proposed template was used for was practiced. Furthermore, we researched a change in "creative problem solving ability" by evaluating a made model objectively. Some effect was confirmed from that result of investigation.

1 INTRODUCTION

In recent years, creativity education in universities has extensively incorporate project-based classroom practices aimed at fostering a creative attitude in learners through experience in production activities. The goal of these courses is to have the learners acquire creative attitudes and engineering knowledge by developing ideas through group discussions, and by giving concrete form to those ideas.

In our prior study, we proposed a framework for achieving more effective design of creative education in engineering (Ishii, Suzuki, Fujiyoshi, Fujii, & Kozawa, 2006; 2007). The course developed learners' creativity by having them make a robot by using LEGO Mindstorms. The result of the

course was an improvement in learners' general creativity, as seen from improvement in their idea generation activity.

In this research, we focus on development of learners' creativity of computer programming as a more expertise in engineering. Specifically, we introduce a modeling template based on UML (Unified Modeling Language) into a programming course. By using a modeling template (hereafter called "UML modelling template"), learners are able to more efficiently develop programming creativity for skills such as "the ability to express their ideas in programs" and "the ability to improve a program by adjusting it to meet conditions." In addition, teachers are able to objectively evaluate programming creativity by referring to the models that learners create.

In this report we first give an outline of the modeling template and explain the evaluation method of the model that was created using the template. Next, we report on the programming development course in which we introduced the modeling template.

2 UML MODELING TEMPLATE

2.1 UML Modeling Template in Programming Education

UML is a standard method of notation for analysis and design of object-oriented software. In developing a system that uses a UML model, program developers ask, as a stage in needs analysis, “What is required by the system?” In other words, they carry out modeling by focusing on user needs. However, it often happens that user needs are unclear, and beginning programmers find it difficult to understand user needs from the development project they have been given. Thus these learners find it difficult to describe in writing a model for user needs. So, we propose a UML modeling template to support modeling of user needs.

2.2 UML Modeling Template Format

The modeling template we propose in this research is based on ideas used in needs modeling in the development of information systems. Specifically, we provide learners with three formatted templates—a functions specifications sheet, a detailed specifications sheet, and a related specifications sheet.

2.2.1 Functions Specifications Sheet

The purpose of creating a functions specifications sheet is to arrange and order needs in regard to the system. A functional specifications sheet arranges and orders needs according to what is important for the system; in other words, it is useful in extracting necessary functions the system should have based on the software development tasks given to programmers. These activities correspond to the activities that extract a ‘use case’ from a use case diagram in UML modeling.

Functions needed to attack a course
<ol style="list-style-type: none"> 1. line trace <ul style="list-style-type: none"> • judgment of brightness • straight ahead movement on black lines of the track • curved movement on white areas of the track • smooth curves 2. avoiding obstructions <ul style="list-style-type: none"> • using ultrasonic waves for recognition • behavior when avoiding something 3. traversal of an incline <ul style="list-style-type: none"> • stable line trace • adjustment of motor power

Figure 1: An example of functions specifications sheet.

For example, in this research we report on a course we taught in which learners make a robot. The learners write a functional specifications sheet to extract the ‘functions that are necessary to make a robot run a racecourse.’ Using the functional specifications sheet proposed in this research, learners itemize the names of the functions they extracted and give simple explanations of for each of these functions (Figure 1).

2.2.2 Detailed Specifications Sheet

The purpose in creating a detailed specifications sheet is to describe in detail the particular parts of a system. A detailed specifications sheet describes the detailed flow of functions that have been extracted. The flow of the various functions becomes clear through a detailed description of the functions that are needed by the system and listed in the functions specifications sheet. Using a detailed specifications sheet template, learners describe and itemize the following for each function: function name, outline of the function, beginning conditions, flow of the function, and final conditions. This template also required learners to sketch an image of the robot’s movements, so as to easily bring out the details of each function (Figure 2).

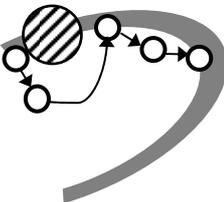
Function name	avoidance of obstructions
Outline of function	make a line trace quickly while avoiding obstructions
Beginning conditions	when approaching obstructions
Flow of functions	<ol style="list-style-type: none"> 1. substitution of 0 for a 2. turn to the right 3. go forward 4. go forward while turning to the left 5. recognize line 6. substitute $ic + 1$ for ic 7. quickly make a line trace 8. substitute 0 for ic
Final conditions	when ic is greater than 1,000
Image sketch	

Figure 2: An example of detailed specifications sheet.

2.2.3 Related Specifications Sheet

The purpose of creating a related specifications sheet is to make clear the total image of the system. A related specifications sheet describes how the functions made clear in the functions specifications sheet are related to each other. These activities correspond to the creation of a collaboration diagram in UML modeling. Using the related specifications sheet template, learners describe in a diagram the changes in functions of the whole system. Then, based on the beginning and final conditions described in the detailed specifications sheet, the conditions for changes can be decided (Figure 3).

2.3 Evaluation of the UML Modeling Template

In this research we evaluated learners' performance according to six items that can be evaluated objectively. These items were: 'number of functions,' 'originality of functions,' 'existence of unrelated links,' 'improvement of new methods and functions,' 'detail of image sketches,' and 'resemblance to the sample model.'

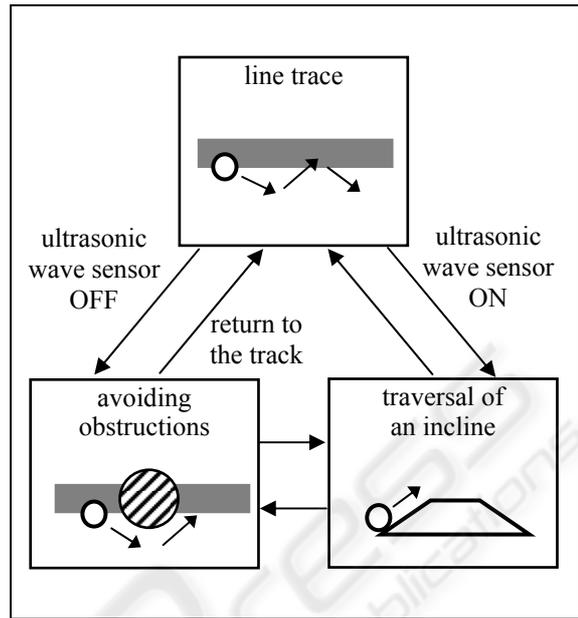


Figure 3: An example of related specifications sheet.

Using the making of a robot carried out in this research as an example, we explain below the evaluation method for each item. Based on the evaluation criteria below, we gave a score from 1 to 5 for each item (for a total maximum score of 30 points).

2.3.1 Number of Functions

We evaluated the number of specifications in the function specifications sheet. Using the average number of specifications made by the teams evaluated, we gave high scores (4 or 5 points) to teams whose number of specifications was above the average and low scores (1 or 2 points) to teams whose number of specifications was below the average.

2.3.2 Originality of Functions

With this criterion we evaluated a team according to whether the functions the team thought of had originality and were not thought of by other teams. We evaluated the model's originality, based on the content of the functions specifications sheet and the detailed specifications sheet. We gave a score of 5 to teams whose specifications sheets included original functions and a score of 1 to teams whose specifications sheets did not have any original functions.

2.3.3 Existence of Unrelated Links

With this criterion we evaluated the appropriateness of the descriptions in the related specifications sheet. We looked at the relation lines drawn on the related specifications sheet, and we gave a score of 5 when there were no relation lines drawn for unchanged conditions (that is, there were no unnecessary lines drawn). When relation lines were drawn for unchanged conditions, we reduced the score from 5 according to the number of unnecessary relation lines drawn.

2.3.4 Improvement of Methods and Functions

With this criterion we evaluated whether functions and course attack strategies not introduced in the model in the first half of the course were introduced in the model in the second half of the course. In the course we judged whether new functions were introduced from the function names that were described in the function specifications sheet. We also judged whether new racecourse attack strategies were introduced from the flow of functions and image sketch described in the function specifications sheet. We gave a score of 5 for the introduction of new functions and course attack strategies and a score of 1 for no introduction of new functions and racecourse attack strategies.

2.3.5 Detail of Image Sketches

With this criterion we evaluated the extent learners were able to express their own ideas in the model. In the course we based our evaluation on the average number of comments per function for the functions listed in the image sketch in the detailed specifications sheet. We gave a score of 3 to teams whose number of comments was the same as the average number of comments per team. Teams whose number of comments was above this average received a high score (4 or 5), and teams whose number of comments was below this average received a low score (1 or 2).

2.3.6 Resemblance to the Sample Model

In the course written examples of a function specification sheet and a detailed specifications sheet were passed out to learners. We evaluated teams according to the extent to which their function specification sheet and detailed specifications sheet resembled the written samples passed out to learners (Table 1).

Table 1: Evaluation criterion of resemblance to the Sample Model.

Point	Evaluation criterion
5	No resemblance between the team's functional specification sheet and detailed specifications sheet and those in the sample model.
4	In the avoidance of obstructions in the detailed specifications sheet, the method of avoiding obstructions resembled that in the sample model.
3	In the functions specifications sheet, the outline of specifications resembled the outline in the sample model.
2	In regard to avoidance of obstructions in the detailed specifications sheets, both the method of avoiding obstructions and the comments made resembled those in the sample model.
1	The team's functional specifications sheet and detailed specifications sheet were both similar to the sample model.

3 DESIGN OF A COURSE USING UML MODELING TEMPLATE

We designed a programming course in creative education; this course held in the Autumn Term in the 2007 academic year at the Chubu University College of Engineering.

3.1 Learning Phases

The educational program consisted of three main phases.

3.1.1 Phase 1: Introduction

As an environment for creative activities by learners, put in place a programming environment comprised of a laptop PC for each learner. In this phase, the learners acquired basic knowledge of LEGO Mindstorms and UML modeling.

At that time, to help learners understand UML modeling, we gave an explanation in slides that gave

function names and outlined concrete examples for function specifications sheets. In regard to function specifications sheets and detailed specifications sheets, we also passed out hand-written concrete examples to learners.

3.1.2 Phase 2: Experiencing Creative Activities

Learners formed pairs, working together to produce a robot; they then participated in a “time trial” competition. The competition is a race comprising one lap of a course. The learners were required to produce a robot that not only moves, but also avoids obstacles and has a function that traces a line where it has moved. Robot system modeling was carried out two times, in the first half of the course and in the second half of the course.

During these activities, the learners regularly recorded their UML model to the modeling template sheets. These activities are recorded using the creative activity support system described in 3.2.

3.1.3 Phase3: Reflection

The field of learning sciences, which studies human learning processes in educational situations points out the importance of “meta-cognition” in which the learner’s own activities in an educational or learning situation are seen from a “meta” perspective (Brown, 1987). In this research, we incorporated this meta-cognitive activity of “reflection” into the course.

After the creative activities, learners undergo “reflection” to deepen their understanding and awareness of their own creative activities. Learners summarize their groups’ creative processes in a chart using a piece of paper measuring about 2m x 1m (hereafter called “reflection sheet”).

The reflection sheet is divided in three sections: Plan, Do, and See. The PDS cycle is a synonym for the PDCA cycle, which is a management method for operations to continually maintain and improve their quality. The learners position the UML model, PAD (Problem Analysis Diagram), and photo of the robot in the sheet. As a supplementary explanation for these materials, at each stage of the creative activities, the learners write on the sheets (1) what they are planning and (2) what results they achieved. The materials used for placement on the reflection sheet are the items recorded in the creative activity support system.

3.2 Creative Activity Support System

In this study, we therefore developed a creative activity support system that would enable recording, management, and viewing of the groups’ creative activities, to provide support for the learners’ “Reflection” activities. This web-based system, which is comprised of a PHP linked with a database (MySQL), enables the learners to upload any items related to the group’s creative activities using a browser.

The information recorded by the learners is updated in real time, so the learners can check the ranking status of their own or other learners’ teams at any time. Using this system, the learners regularly record the status of their groups’ activities. They then create the Reflection Sheet while viewing their own creation processes, which they have recorded in the system, and downloading the appropriate data as required.

4 EVALUATION OF THE COURSE THAT INTRODUCED MODELING TEMPLATES

We compared models in the first half of the course and those in the second half of the course and analyzed changes in learners’ programming creativity. We analyzed 25 teams that had no changes in members among the learners.

4.1 Changes in Overview of the UML Models

First, we compared the total number of points for the models in the first and second halves of the course. Figure 4 shows the points for each team’s model in the first half of the course and the second half of the course.

As a result of this analysis, we confirmed that the models in the second half of the course on the whole achieved higher points than those in the first half of the course ($t(24)=2.843$, $p<.01$). In other words the modeling skills of learners in the course improved. This is one result that shows an improvement in learners’ programming creativity.

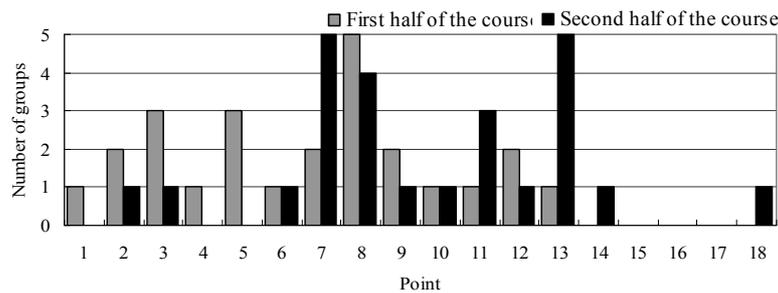


Figure 4: Changes in total points of the UML models.

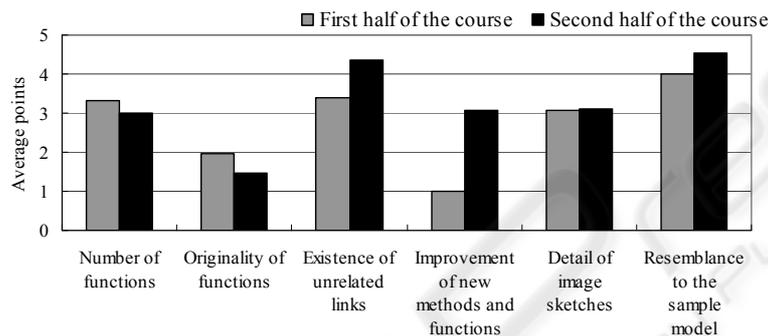


Figure 5: Changes in each evaluation criterion of the UML models.

4.2 Changes in Each Evaluation Criterion of the UML Models

Next we compared the scores for each evaluation item in the first half and the second half of the course. Figure 5 shows the averages for each evaluation item in the first half of the course and the second half. The result of our analysis was that in the second half of the course scores increased for the items 'Existence of unrelated links ($t(24)=2.001$, $p<.10$)' and 'Resemblance to the sample model ($t(24)=3.375$, $p<.01$).'

This result shows that through the course the learners understood the descriptive form for modeling and became able to write down their ideas appropriately as a model. These results show the effectiveness of introducing a modeling template.

On the other hand, the scores for 'number of functions' decreased in the second half of the course. In addition, scores for 'Introduction of original functions' were low in both the first half and the second half of the course. In the future it is necessary in the course to try to come up with some ways to improve the scores on these items.

5 SUMMARY

In this research we designed a programming course that introduced a modeling template based on UML. The results were that learners' modeling skills improved in the course. This shows the effectiveness of the modeling template we introduced in this research, with one of the results being an improvement in learners' programming creativity.

REFERENCES

- Brown, A.L., 1987. Metacognition, executive control, selfregulation, and other more mysterious mechanisms, In F.E. Weinert & R.H. Kluwe (Eds.), *Metacognition, Motivation, and Understanding*. Lawrence Erlbaum Associates, Hillsdale, NJ.
- Ishii, N., Suzuki, Y., Fujiyoshi, H., Fujii, T., Kozawa, M., 2006. Designing effective learning environments for creativity. *WSEAS Transactions on Advances in Engineering Education*, 3, 955-962.
- Ishii, N., Suzuki, Y., Fujiyoshi, H., Fujii, T., Kozawa, M., 2007. A framework for designing and improving learning environments fostering creativity. *Psicologia Escolar e Educacional*, 11, 59-69.