Directing All Learners to Course Goal with Enforcement of Discipline Utilizing Persona Motivation

Dong Phuong DINH†, Nonmember, Fumiko HARADA††, and Hiromitsu SHIMAKAWA†††, Members

SUMMARY The paper proposes the PMD method to design an introductory programming practice course plan that is inclusive for all learners and stable throughout a course. To achieve the course plan, the method utilizes personas, each of which represents learners having similar motivation to study programming. The learning of the personas is directed to the course goal with an enforcement resulting from the discipline, which is an integration of effective learning strategies with affective components of the personas. Under the enforcement, services to facilitate and promote the learning of each persona can be decided, based on motivation components of each persona, motivational effects of the services, and the cycle of self-efficacy. The application of the method on about 500 freshmen in C programming practice course has shown this is a successful approach for designing courses.

key words: contextual inquiry, persona, motivation, learning strategy, discipline, enforcing services, facilitating services, summary learning behaviors, progressive learning behaviors

1. Introduction

Programming practice courses are often compulsory subjects of engineering training in many universities. Since the number of learners taking the course often reaches to hundreds, the course involves many teachers and teaching assistants (TAs).

Motivation and learning strategies are essential elements for learners to be successful in the learning [13]. Both of them have many components [14]. Large number of learners have wide diversity in the components. Some learners study programming for their curiosity. They control their time for programming. Other learners uninterested in programming engage themselves in it under support of friends, because it is a compulsory subject. The course should be inclusive, which means it must provide benefits not for specific learner groups, but for as many learner groups as possible. Moreover, the learners change their motivation and learning strategies as they improve their abilities. The Instructional Design methods [6], [15] consider it important to follow the changes to give services suitable for individual learners. However, any adjustment during the course is critical in case many parties are involved. Since the adjustment may affect some parties, it needs assent from them, which prevents timely accommodation. In addition to that, many educators such as teachers and TAs take charge of hundreds of learners. The success of the adjustment depends on the ability of each educator, which lacks the fairness in education. Therefore, the course plan should initially involve services helpful for learners in various situations they may encounter throughout the course. In other words, the course plan should be stable. However, it would be infeasible to include tactics for huge diversities coming from hundreds of learners into a course plan.

To address the diversities, studies based on the ARCS model[9] esteem learner motivation. Motivating factors in e-learning tools for programming are discussed in [10]. McGill [11] evaluates programming education with robots from formal and qualified feedbacks from learners. Motivation transition of learners in programming are analysed in [19]. An example of programming course design methods based on the ARCS model is illustrated in [8]. Many kinds of tactics [2], [5], [12] are proposed to maximize the performance of all learners. Nevertheless, none of them addresses a design method to build an inclusive and stable course plan in an educational environment where hundreds of learners are supervised by many educators.

The paper proposes the PMD (Persona, Motivation, and Discipline) method to design inclusive and stable course plans for environments many parties in large quantity take part in. The method assumes characteristics of whole learners usually remain unchanged for a few years. It takes actual contexts of past learners and their operations on the learning supporting site. A persona combined with its scenario is a model of learners similar in motivation and learning strategies. The model enables course designers to predict their learning behaviors. Before the course starts, course designers determine educational services based on the prediction. Course designers extract a discipline from self-regulation strategies of high achievement learners to direct various contexts of all learners toward the course goal. In the PMD method, course designers make a balance of facilitating services with the discipline, referring motivation components stated in a scenario of each persona. A persona provides every educator with a common image of a group of learners similar in motivation and learning strategies. Founding on the common image, a course plan is designed with educational services for motivation of every persona and the discipline. The PMD method minimizes variances in supervision among many educators.

Applying the PMD method to over 500 students supervised by 10 teachers and 100 TAs, we have improved the
learning time of the students and succeeded in keeping their scores even for harder assignments, in 2 successive years.

2. Introductory Programming Practice Course

2.1 Motivation and Learning Strategy

Motivation and learning strategies of a learner play vital roles to her success [17]. Pintrich et al. [14] have proposed “A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ)” which is a guideline to understand motivation and learning strategies of a learner. It explains motivation and learning strategies with multiple scales. For each scale, it provides question samples for easy reference. The scales can be used to identify motivation and learning strategies of a learner studying programming.

For motivation, MSLQ lists up intrinsic, extrinsic, task value, expectancy, and affective components. Learners who have intrinsic components participate in target tasks because their goals are achievement of the tasks. They work because of reasons such as their challenge, curiosity, and mastery. For intrinsic components, MSLQ lists example questions like “In a class like this, I prefer course material that really challenges me so I can learn new things.” MSLQ also indicates the correlation of the question with the final grade is 0.22. Learners depending on extrinsic components participate in tasks because a learning task is the means to their goals. Components such as rewards, scores, and competition make them engage in the tasks. Task value components are determined by learner perceptions of the tasks in terms of interest, importance, and utility. Expectancy components refer to learner beliefs that their efforts will result in good outcomes. Affective components are related to anxiety for tests, grading for credits, and so on. For example, since they worry about tests, they make efforts.

A learning strategy is a personal approach to understand information and solve problems. Learning strategies consist of resource management strategies and self-regulation strategies. The resource management relates to the learner ability to understand the usefulness and effectiveness of given resources for the learning. As for resource management, MSLQ enumerates study environment strategies regarding to setting of study places, peer learning strategies corresponding to collaboration with peers, and help seeking strategies to get supports from others including peers and instructors. Self-regulation strategies work in the process whereby learners systematically direct their thoughts, feelings, and actions toward the attainment of their goals [16]. Yukselturk [20] states self-regulation learning strategies are strong predictors for success in on-line courses. Self-regulation strategies in MSLQ involve cognitive and metacognitive strategies, strategies on time, and strategies on effort regulation. Cognitive strategies refer to methods to learn, such as how to remember new concepts, to organize them, and to understand them, while metacognitive strategies correspond to methods to tune and adjust cognitive strategies. Strategies on time plan study time. Effort regulation strategies control their efforts in the face of distractions and uninteresting tasks.

2.2 Summary and Progressive Learning Behaviors

To improve the usability of a working environment, the contextual inquiry method [1] insists to delve into working processes of users with interviews to them. To establish a good course plan, we need to know every detail of behaviors of learners in specific conditions. Suppose the learner mentions help from her friend to solve the assignment, when an interviewer asks how she solves an assignment. In the contextual inquiry, the interviewer should delve into topics the learner mentions. In this case, for example, the interviewer asks why the help occurs, how the help takes place, and how the learner feels after the help. The further the interviewer delves into the context, the clearer image of her learning the interviewee obtains. In the contextual inquiry, the interviewer describes a scenario summarizing what the interviewee has experienced. The scenario expresses contexts [4], which consist of sequences of actions and feelings of the learner in specific conditions. The paper refers to behaviors in the scenario as summary learning behaviors. Summary learning behaviors contain useful information to identify motivation and learning strategies of the learner.

To obtain a clear image of learning behaviors in specific conditions, a questionnaire would not be a good method, because questions in it are pre-determined with assumptions of investigators. Learners who give answers to the questionnaire may not have real experiences for conditions assumed in questions. It prevents collection of truthful information from learners. The contextual inquiry let us know what behaviors learners take in actual conditions.

Web servers facilitate learning anywhere and any time. They also enable to take operation logs of learners to understand their learning progress. The logs express interactions of learners with the servers through Web pages such as viewing assignments, downloading sample codes, and submitting source codes. Since the logs are taken throughout the course, learning activities are recorded in a progressive way. The logs express progressive learning behaviors.

Together with summary learning behaviors from the contextual inquiry, progressive learning behaviors bring information expressing motivation and learning strategies of learners. Summary learning behaviors are obtained from oral words of learners. Meanwhile, progressive learning behaviors are series of genuine actions learners actually take. Suppose course designers try to support their motivation and learning strategies using only summary learning behaviors. Information in interviews is not always expressed in a quantitative manner. The course designers should confirm summary learning behaviors against progressive learning behaviors. Compared with summary learning behaviors, progressive learning behaviors are far less expressive for what learners want, what they expect, what they mind, how they behave in specific learning contexts. It is necessary to combine the both kinds of learning behaviors to design a successful
2.3 Cycle of Self-Efficacy

Every learner has her own motivation components. Some learners work because of extrinsic components, and others for expectancy ones. Because of the diversity, requirements for a course plan vary with learners. Among them, intrinsic components are indispensable for successful learning. The educators should lead low intrinsic learners to a status where they have enough intrinsic motivation. Because of low intrinsic motivation, they are not willing to work on programming. However, if they work on programming assignments, they may be able to solve some of them. The experience gives the learners pleasures and confidence, which increases expectancy in them. Because of confidence, they get willing to work, which means the intrinsic motivation is improved in them. It is a good cycle of self-efficacy. Even if those learners start to work, they will stop learning, when they face difficulties in programming. Educators should support them, providing services facilitating their learning. For example, face-to-face supervisions of TAs should be prepared for learners who cannot solve easy basic assignments.

For learners whose initial intrinsic motivation is high, they would have high will to master the knowledge. The educators should give services to promote their motivation further. For example, hard graphics assignments with visual execution results would appeal them to explore programming more. Announcement of the score rank would promote them to try to solve many hard assignments. If there is no service to maintain and promote their intrinsic motivation, the intrinsic motivation gets decreased [9].

3. Personas for Inclusive and Stable Course Plans

3.1 Contexts

Some learners are eager for programming, while others have no interest in it. Every learner has her own state in learning. States strongly depend on her motivation and learning strategies. In a learning process, a learner receives a stimulus. Her state changes into a new one as a reaction to a stimulus in an environment. She takes an action depending on her state, her environment, and the stimulus.

A scenario expresses state transitions of a learner along with her actions. We can understand meanings, functions, purposes of actions with states, stimuli, and environments [7] described in a scenario. A context is a chain of states along with stimuli and environments, which is visualised with Fig. 1. Similarly, a learning behavior corresponds to a series of actions along with a state transition.

A context depends on motivation and learning strategies of the learner. MSLQ works as a good guideline to interpret motivation and learning strategies. We can traverse causal relationships of actions in a context. When course designers see the scenario, they assume motivation and learning strategies which explain state changes and actions in a context. They verify the assumed ones in others. Since the context expresses how her state changes, course designers can identify motivation and learning strategies which results in the state changes, referring to the guideline. MSLQ states scales of motivation and learning strategies. It also enumerates sample questions. For each of them, MSLQ shows the strength measured by the scale for its answer. Based on the similarity with the sample questions, motivation and learning strategies implied in the context can be assessed with rough ranges such as strong, weak, and none.

Suppose the following scenario is obtained through the contextual inquiry. A learner faces with a hard assignment in a practice class. She tries to ask help from a TA as her action. She also feels pleasant with encouragements from TAs. However, at that time, there is no TA available in the environment. She runs into a disappointed state. Referring to MSLQ, we can interpret the learner as an owner of a weak extrinsic component and a strong help seeking strategy, because she expects encouragements and easily gets disappointed when she fails to be helped.

Since a learner differs from others in motivation and learning strategies, she shows different actions even if she experiences same events as others do. Precise contexts and correct interpretation of them are necessary to provide suitable educational services for learners. The contextual inquiry enables us to get truthful contexts of learners, because it delves into what actually happens in learning processes.

3.2 Personas

A large number of students in programming training are divided into several classes. Since each of the classes is supervised by a separate teacher and TAs, variance of teaching often appears among classes. It should be avoided. Meanwhile, each class has great diversity because there are various learners. Course designers should minimize the variance as well as cover the diversity.

As Cooper states, scenarios of learners similar in their characteristics can be compiled into a scenario for a persona [3]. A persona is a virtual learner with a scenario stating its behaviors in specific contexts. It represents learning behaviors of a group of learners similar in motivation and learning strategies. Using a persona with its scenario, course
designers make teachers in individual classes to grasp a single image of members in a learner group, which contributes to elimination of the variance among classes. Instead of taking care for many individual learners, course designers can concentrate on managing a small set of personas. An inclusive course can be designed, taking care of personas dominant in the course. A persona enables to predict reactions they take on specific stimuli in a specific environment. Course designers can choose appropriate educational services for each learner group. Owing to the prediction with personas, they can determine all educational services covering learner diversity, before the course starts. Teachers can conduct a stable course plan throughout the course.

Instead of following development of individual learner, personas make course designers cover whole population of learners. Suppose a class consists of several personas including \( P_w \) and \( P_f \). \( P_w \) learns programming because it is a compulsory subject. If programming problems seem easy, he engages in the learning, otherwise he tends to set the problems aside. Namely, \( P_w \) has a strong affective component and a week self-regulation strategy. On the other hand, since \( P_f \) has extrinsic components, he works well when he gets a praise from TA. At one time of the course, with supervisions of TAs, a learner who used to belong to \( P_w \) may have proceeded to \( P_f \). A learner of \( P_f \) may fall into \( P_w \) because of no care from TAs. As each learner is constantly changing his characteristics, he may belong to different personas when the time proceeds. However, if the whole set of personas are involved in the design of the course plan, the course would be stable against the change of learners.

From empirical analysis in teaching, the paper assumes that the set of personas covering almost whole learners in an educational institute such as a university remain unchanged for a few years. The contextual inquiry is conducted on learners who have experienced the programming course.

3.3 Enforcement with Discipline

To give services suitable for the learning, the course designers should grasp all of the learning contexts a persona stays in. However, it is infeasible because there are too many learning contexts. The number of learning contexts can be reduced, directing the learning of all personas toward the course goal. As explained in 2.1, self-regulated learners direct themselves to the course goal [20]. The elimination of learning contexts out of the course goal makes it possible to design an inclusive course plan. For example, low motivated learners want to extend their codes submission as late as possible. If deadlines are 2 week while new assignments come out every week, they would engage in assignments 1 week behind. They would lose chances to acquire new skills every week. Self-regulated learners would submit their codes within 1 week, to catch up the course progress. A discipline is necessary to enforce all learners to behave as self-regulated ones do. Controlling environments and stimuli, all learners should be directed to the course goal.

The PMD method regards a discipline as self-regulation strategies of high achievement learners. Referring to a scenario of every persona, course designers find affective components of every persona, such as tests and deadlines. They make the course settings reflect the affective components, to introduce enforcing services. Suppose the discipline is solving all assignments early. To enforce it on a persona who minds scores, the educators should announce that solving assignments after deadlines would receive no score. The enforcement would make the persona try to solve all assignments within the deadlines. To impose the discipline to another persona anxious of tests, the educators should often test their understanding during the course. Since they mind tests, they would try to solve all assignments before the tests.

Learning strategies to achieve high results vary with training conditions. The course designers should determine a discipline from actual learning behavior in a target training condition. Assuming personas in an educational institute do not change for a few years, the PMD method extracts a discipline from self-regulation strategies of high achievement learners who have achieved good results in previous courses.

3.4 Facilitation with Motivation

Only enforcement does not work well to make learners engage in studying programming. As pointed in the ARCS model [9], learners should be motivated. The course designers should get their attention to improve task value components, and show the relevance to enhance intrinsic motivation components. They create a good cycle of the self-efficacy in each persona for the confidence, which satisfies the persona. Since a scenario of each persona indicates motivation components, they can choose appropriate services for every persona.

Suppose a persona is weak in intrinsic motivation, but strong in anxiety of bad scores. With the enforcement of solving all assignments within one-week hard deadline, the persona would start to work because of the anxiety. However, since he has little will to learn, he would not be willing to solve any problem. The course plan should prepare services where supervisions to overcome problems are timely given to them. Since the persona has little confidence, face-to-face supervisions of TAs would be effective. Once he can overcome the problems, the confidence in solving problems would be improved in him. After his ability gets improved, the course should not only give face-to-face supervisions services to support him to solve problems, but also give him promoting services to encourage him in learning more, such as feedbacks showing his learning improvements.

3.5 PMD Method

To address training of programming for large number of learners supervised by many teachers and TAs, the PMD method aims at achieving an inclusive and stable course plan. The course plan must be set up before the course starts. Moreover, major services should not be changed throughout
the course.

A persona along with its scenario models a group of target learners. It enables course designers to predict behaviors of learners corresponding to it. A discipline is an external factor to make learner engage in the training. Course designers enforce a discipline on all learners to direct their learning toward the course goal. Motivation represents an internal factor for learners to engage in training of their own accord. Course designers choose services for each persona from motivation stated in its scenario.

Expecting high achievement learners regulate themselves toward the course goal, the PMD method identifies a discipline from their progressive learning behaviors. Since course designers can predict learning behaviors of every persona from its scenario, they can choose course settings providing an environment and stimuli to force all learners to behave as if they have the discipline. The course settings direct all learners toward the course goal. The direction eliminates their contexts out of the course goal, as presented in Fig. 2. For example, a discipline making learners work on many assignments release teachers from encouraging learners in working longer time. Since the number of the contexts are limited, course designers can make an inclusive course plan covering learner diversity. The inclusiveness realizes a stable course plan free from adjustment during a course.

4. Steps to Realize PMD Method

The PMD method consists of five steps as shown in Fig. 3: identifying personas from the contextual inquiry, finding a discipline from learning behaviors of high achievement learners, choosing services combined with affective components to enforce the discipline, choosing services to facilitate the learning of every persona, and integrating the services to achieve a feasible course plan.

4.1 Identifying Personas

The contextual inquiry method is used to prepare qualified data expressing learning contexts. To get qualified contexts, the interviews are conducted on more than 50 learners [1]. The learners having finished a programming course conduct interviews with each other to describe their learning behaviors. Since learners have to make a lot of efforts in the programming course, many complaints would fill up the interviews just after the course. Interviews after a too long break would lose truthful contexts. In the PMD method, interviews are conducted after 1 semester break from the course. Before the interviews, lectures are conducted to teach the learners what are contexts. The lectures also explain the purpose of interviews is to know contexts where learners gain or lose their vigor to study programming. The lecture emphasizes the procedure of the contextual inquiry obliges interviewers to delve into the details of the learning behaviors mentioned in the answers. For example, suppose an interviewee mentions he solves all assignments within the week. The interviewer asks why he takes the learning strategy. He explains assignments in every week are oriented for a specific programming skill. By solving all of them, he can acquire the programming skill, which is necessary to solve assignments in the following weeks. He also want to avoid solving assignments of 2 weeks at the same time. The interviewer delves in a reason of his enthusiasm for acquiring all kinds of programming skill. His answer reveals he wants to be a game programmer.

Using interviews from many learners, personas are determined according to the following procedure.

1. Actual contexts obtained in each interview is summarized as a scenario. A scenario shows learning behaviors of a learner in specific contexts.
2. Each scenario is analyzed manually to identify motivation and learning strategies based on the scales explained in MSLQ [14]. Scenarios are classified into groups based on the similarity of motivation and learning strategies.
3. A representative learner is determined in each group as who best represents characteristics of motivation and learning strategies in the group. The scenario of the
learner is referred to as a representative scenario.

4. In each group, significant learning contexts in scenarios other than the representative one are appended to the representative scenario, to be compiled into a hybrid scenario.

5. Based on the hybrid scenario, a persona is determined as a virtual learner who characterizes motivation and learning strategies of the learner group. Course designers manually determine the strength of the association of the persona with each kind of motivation, referring question examples and their scale values in MSLQ. The strength is assessed in rough ranges such as strong, weak, and none.

The procedure reveals how many personas exist, what motivation and learning strategies each persona has.

4.2 Finding Discipline

A persona characterizes motivations and learning strategies in a group of actual learners. We do not always find a persona indicating efficient learning strategies to attain the course goal, because the classification of personas has no relationships with the efficiency. Meanwhile, learners of good achievements in past courses are expected to own efficient learning strategies to achieve the course goal. The progressive learning behaviors express how the best learners proceed their learning. To find the discipline, efficient learning strategies are figured out from their progressive learning behaviors. Assuming a normal distribution, the second step of the PMD method focuses on the best 17% learners, who are better than the average by more than one standard deviation based on the score of assignments and tests.

The analysis of the progressive learning behaviors of high achievement learners enables course designers to make assumptions on what are their learning strategies. Among many learning strategies, self-regulation strategies are esteemed in the PMD method, because they are important in on-line courses [20]. The course designers pick up several self-regulation strategies assumed to be used by high achievement learners. They examines whether each of the strategies appears in the progressive learning behaviors, to confirm high achievement learners have used it. They adopt a set of confirmed self-regulation strategies as a discipline.

For example, suppose a programming course consists of weekly classes. The objective of earlier weeks is prerequisites for later weeks. Let the setting of the course as follows. Three difficulty levels of assignments are given to the learners every week: easy, intermediate, and hard ones. The course requirement for credit is to solve intermediate assignments within a two-week deadline. Easy assignments help some learners fill in understanding gaps. Hard ones are prepared to satisfy other learners with challenging minds. High achievement learners would regulate themselves so that they solve all assignments within one week, in spite of the two-week deadline. The regulation avoids duplicated burdens of assignments in the week and new ones coming in the succeeding week. They also seem to know that solving only intermediate assignments does not enhance their programming ability. Judging from their learning behaviors, course designers would make a discipline so that learners should finish all assignments within one week.

4.3 Choosing Services Containing Affective Components

In the third step, course designers seek for affective components to enforce the discipline, using the progressive learning behaviors. They utilize anxiety of learners to make learners behave according to their intention. The anxiety may come from affective components such as a deadline, a test, a check, and so on. Once the affective components for the discipline are identified, course designers find out services containing the affective components.

Suppose a discipline is to make learners work for longer time. Many affective components are available, each of which makes a specific persona study. For example, to personas afraid of the checks, course designers can adopt interactive checks, where supervisors question learners on their codes in a face-to-face mode to confirm their understanding. To those afraid of tests, course designers should conduct frequent tests during the course. Let us consider another discipline aiming that the learners should solve all assignments early to acquire enough programming skills for successive weeks, as mentioned in 4.2. The course designers need to identify what factors work well to make learners of a specific persona finish all assignments before the succeeding week comes. If hard deadlines are found to be an affective component to the persona, assignments with hard deadlines should be included in the course plan. It is effective to prevent the persona from postponing their submissions.

4.4 Choosing Facilitating Services

This step aims at enumerating service candidates to facilitate the learning of personas. Course designers choose services to facilitate each persona to keep the discipline, from the correspondence of motivation components of every persona to motivation effects of each service.

The first task in this step examines motivation effects of each service. For a service used in previous courses, course designers confirm how it contributes to motivating learners, using the progressive learning behaviors. Let us consider an interactive check, where a TA checks a source code in a face-to-face manner with a learner. The TA together with the learner confirms what points the code has achieved, and what points the code has not. The TA supervises the learner to achieve a good code. Consequently, the learner would improve knowledge and skill after careful supervisions. In other words, interactive checks effective for learners owning extrinsic components improve their intrinsic and expectancy components. To confirm the effects of the interactive check, we can examine how the learner behaves after she receives the interactive check: whether she takes another interactive check, whether she tries to solve more assignments, whether
she improves her score, and so on.

Sometimes, learners request a new service in interviews. Since course designers have no progressive learning behaviors on it, they have no other way to estimate its motivation effects. Suppose graphics assignments as new motivation components. They are expected to have association with task value effect, because they appeal the learners with the visual execution results.

In the second task, the course designers choose facilitating services for each persona. The association strength of each persona with motivation components is interpreted from scenarios as explained in 4.1. The course designers choose services which are incident with the persona in terms of motivation aspects, to determine facilitating service candidates for the course plan. At this time, they give priority to ones whose motivation effects are confirmed in the first task. An example is a help-seeking persona who has extrinsic and affective components. Suppose a scenario shows a persona prefers interactive checks and pair programming. Suppose interactive checks are confirmed to be associated with extrinsic components, while pair programming is only estimated to be, in the first task. Since interactive checks are associated with the help-seeking persona in terms of extrinsic components, they are included in the service candidates for the persona. Pair programming also joins the service candidates, but it has a lower priority than interactive checks.

4.5 Integrating Services into Feasible Course Plan

An actual course plan may not include all service candidates because of constraints such as the cost or the quantity of supervisors. Course designers should build a feasible course plan, considering the constraints. Some services may cost too much to provide for all learners, even though they are effective. Course designers should provide the services, giving high priorities on personas they want to improve most.

At the same time, course designers should provide other services to promote learners who fail to receive the costly services, considering contexts where they need the services. Contexts acquired from the contextual inquiry play vital roles in the consideration. For example, interactive checks for all learners are infeasible because of the limitation of human resource such as teachers and TAs. Even if interactive checks are confirmed to be good for many personas, course designers have to reduce the number of learners using the service. It is one way to eliminate advanced learners from a target group of the service. Suppose the contextual inquiry shows that the learners want to get not knowledge but praises from TAs in interactive checks. Reflecting the context, course designers would substitute interactive checks with a ranking service.

5. An Example of PMD Method

We present experiences of application of the PMD method upon the condition of Ritsumeikan university, Japan. The course plan in 2010 is improved into ones in 2011 and 2012.

5.1 Training Condition

Every academic year, about 500 freshmen enrol the mandatory introductory programming practice course. The total number of students in 2010, 2011, and 2012 are 474, 546, and 557, respectively. The lecture course on C language specification is synchronized with the programming practice course. In the lecture course, students are divided into 2 classes whose teachers have common materials and tight communication for identical teaching. Meanwhile, they are divided into 10 classes in the practice course. Each class is supervised by 1 teacher and 10 TAs, who spent almost whole class time for individual supervision and source code checks, because they are indispensable in programming practice. Table 1 shows the course schedule, which does not include the 1st, the 2nd, the 10th and 15th week. They are used for Linux practices, emacs practices, a midterm test, and an end-term test, respectively. The schedule is identical in the three years.

The course plans in 2010 is the followings. Every week, one regular practice class and two complementary ones, 90 minutes for each, are arranged for all of the students to practice under supervisions of TAs and teachers. Assignments of every week are arranged into three difficulty levels: easy, medium, and difficult. The total score given for all assignments of a week is 130, in which the easy ones, the medium ones, and the difficult ones take 30, 60, and 40, respectively. The students can submit source codes for the easy and difficult assignments any time in the semester, while two-week soft deadlines was obliged to the medium ones. If a source code for a medium assignment is submitted later than 2 weeks, the submission is accepted but the score of the code is deducted 10% every one week delay. The assignments and some sample codes are given to the students via a Web site though which the students can submit the source codes at any time from any place. The regular classes take attendance into account while the complementary ones do not. To help students overcome initial programming difficulties, the teachers and TAs give interactive checks on all source codes submitted before the midterm test. To the codes submitted after the mid-term test, they do not give any interactive check, but off-line checks.

5.2 Personas from Contextual Inquiry

We extract personas in the first step. We have conducted the

| Week | Content          | Week | Content                   |
|------|------------------|------|---------------------------|
| 3    | variable & expression | 9    | function with array       |
| 4    | conditional statement | 11   | string                    |
| 5    | loop statement    | 12   | string with pointer       |
| 6    | nested loop       | 13   | structure                 |
| 7    | function call     | 14   | recursive call            |
| 8    | array             |      |                           |
contextual inquiry to get summary learning behaviors. The following shows a part of an interview example of student R (interviewer) and student E (interviewee).

R: What makes you inclined to programming?
E: I regard programming as a challenge. When I have solved a tough assignment, I obtain a strong sense of achievement.
R: When you face a tough one, you might sometimes find no way to solve it. Have you ever run into such a situation?
E: Yes, many times. In such a case, I will repeat to check sample codes in the textbook. Sometime I try easier assignments in the same section.
R: What will you do, if you cannot get anything from them?
E: I will search Web pages explaining similar matters using the Internet. I do not prefer to be supervised by TAs, because I feel lost in the challenge.
R: But, they give you hints, even an encouragement sometimes.
E: I am not pleased, even if they encourage me. but, when they give high grades to my codes, I get satisfactions.

The scenario below is described from the interview. “Mary regards the programming as a challenge. She likes programming because she gets a strong sense of achievement when she has solved hard assignments. When she cannot solve them, she repeats to check sample codes or tries easier assignment in the same section of the textbook. If she cannot get anything from them, she tries to find explanation of similar matters with the Internet. Because of her pride, she seldom asks TAs for supervision. Their encouragements do not please her, either. In the mean time, high grading for her codes from TAs satisfies her.”

The scenario reveals the context of facing tough assignments. In the context, student E has shown her attitude to address assignments. She also explains ways to overcome them, which is utilizing information from the textbook and the Internet. The attitude implies her intrinsic component, while searching for information from textbooks and the Internet indicates her learning strategies. The context also reveals two resources, sample codes and explanations, which are hints to prepare services for the course plan.

To build the course plan of 2011, we have got scenarios from 73 students. We have found 3 groups of students similar in motivation and learning strategies. The groups remain unchanged in the analysis for 2012. One group consists of 27 students willing to learn. They understand the more they engage in the study, the more they can get knowledge and skills. Some of them want their high ability to be praised by supervisors or friends. This group, represented by persona P1, is specified to have strong intrinsic, expectancy, and task value components, along with weak extrinsic ones. Moreover, P1 knows how they should study to improve their abilities. They know to utilize various resources such as explanations by TAs and sample codes in the Internet. It reveals that P1 has learning strategies to achieve a good result.

The second group consists of 19 students who understand the compulsory programming subject brings many benefits to them. Their learning purpose is to get a credit. Their attitudes to the learning are not stable. As far as they can go on with assignments easily, they are eager to study. But, for assignments which seem hard to solve, they tend to put them aside, because they do not believe their ability. They often stick to specific resources such as TA helps. They do not know utilizing many kinds of resources. We refer to this group as persona P2. We regard P2 has weak intrinsic, and strong extrinsic, weak task value, and strong affective components. P2 has no learning strategy.

The third group, represented with persona P3, consists of 27 students who think that they are obliged to learn programming. They do not like to learn but have to, because of the compulsory subject. They learn only at school with help of supervisors. Different from P1 and P2, P3 does not show any will to learn. P3 has no intrinsic component. Since P3 minds obligations so much, P3 has strong affective component and weak extrinsic one. P3 has no learning strategy.

Figure 4 shows associations of the three personas with motivation components. The solid lines show strong associations, while dashed lines correspond to weak ones. The strength of the association is manually determined when course designers determine personas from scenarios of many learners, referring the combination of sample questions and their correlation with the final grade in MSLQ. The figure also indicates service references of the personas in their summary learning behaviors with dotted lines. Assignments are requirements for students to make source codes. Interactive check is briefly described in Sect. 4.4. Complementary classes are prepared for teachers and TAs to supervise students at a loss to solve assignments. Sample codes are model codes with detail explanations. Graphics assignments [18] enable students to view execution results visually. Ranking shows scores of top ranked students.

5.3 Early Submissions and Long Learning Time

The top 17% highest score students are selected according to the average score of weekly assignments. To find efficient self-regulation strategies specified in the second step of the PMD method, we examine the difference of learning behaviors of the top students from those of the remaining. Every week has its own matters to learn. The matters in
the week are used in the succeeding weeks. High achievement students would know the importance of catching up the course progress. They would try to understand matters in the week for succeeding weeks. Among several indexes, we have examined the submission timing and the learning time, to know the self-regulation ability. As we have expected, they solve many assignments within the week, to master the programming knowledge and skills of the week.

Table 2 shows the average days from the announcement of assignments until the code submissions. As the table presents, the top ones submit their source codes much earlier than the remaining. They submit all source codes for basic assignments within 14 days, those for intermediate assignments within 6 days, and those for hard assignments within 31 days. Specially, they solve almost all intermediate assignments within 1 week, which means the submission takes place before the assignment issue in the succeeding week.

Figure 5 shows the average learning time of top students (dashed line) and that of the remaining (solid line). The dashed line keeps higher than the solid one throughout the course. Every week, the top students spend more time to learn than the others do.

Both of early submissions and long learning would be important features among learning strategies of the top students. Early submissions without spending much time for coding would result in codes with bugs. On the other hand, if students spend long time on the learning without submitting codes, their learning would not be fruitful.

5.4 Hard Deadline Assignments and Weekly Tests

The analysis in 5.2 has showed P1 has self-regulation strategies, while P2 and P3 do not. We should enforce the strategies on P2 and P3 as the third step in Fig. 3.

To enforce early submission and long learning time as a discipline, we have to find affective components of P2 and P3. As Fig. 5 shows, the learning time of all students increases strongly between the 13th and the 14th week in the 2010 course. The source code submissions are accepted until the end of the 14th week in 2010. It is a hard deadline. The 15th week is reserved for the end-term test. It proves that the students mind hard deadlines and the end-term test a lot. Table 2 also reveals the affective effects of deadlines. For no-deadline assignments, it takes long time for students to submit them. In contrast, for soft-deadline ones, the submissions happen within the deadline.

We conclude hard deadlines and tests cause the students to learn. We can expect services with these affective components work as a discipline to direct students toward the course goal. Assignments with one-week deadlines would make students solve all assignments before new assignments are issued in the succeeding week. A weekly test would make students study long to master the knowledge and skills every week. With the services, students with no learning strategy would behave as if they have the learning strategies of the top students.

5.5 Facilitating Services

With the discipline, students of P3 would start to learn because they are afraid of the tests and the deadlines. However, they would stop learning soon because they have no intrinsic motivation. They do not want to overcome the difficulties. The course plan should have services so that TAs give supervisions in a timely fashion, not only to help them overcome the difficulties, but also to make them understand the matters. Students belonging to P2 have both intrinsic motivations and affective components. The affective components make them start to learn, but they are easily discouraged by difficult problems because of weak intrinsic motivation. The course plan should provide services with which they can acquire supervisions to solve the problems. For those who belong to P1, since they have strong intrinsic motivation and learning strategies, they can learn by themselves. The course designers should provide services to encourage and promote them.

As the fourth step of the PMD method, we examine services for each persona in Fig. 4. P3 prefers easy and intermediate assignments, interactive checks, complementary classes, and sample codes. As Table 3 shows, all of the former three services have extrinsic components, which match motivation components of P3. Easy assignments initiate them into programming. Since P3 minds scores, they engage in intermediate assignments. Interactive checks and complementary classes provide them with on-time supervisions from TAs. All of the services are good candidates for facilitating P3. Providing sample codes is a new service. Sample codes would be helpful for P3 because they contain detail comments. For P2, interactive checks, complementary classes are also good because P2 can acquire supervisions from the two services. Sample codes are also useful.
for P2 to refer to. In addition to interactive checks and complementary classes, P2 indicates the preference of intermediate assignments and graphics assignments. They are chosen as candidates because the motivational effects of these services match those of P2. Similarly, hard assignments, and ranking are chosen for P1.

5.6 New Course Plan

We have listed service candidates as following: three difficulty levels of one-week hard deadline assignments, on-line weekly tests, interactive checks, complementary classes, graphics assignments, sample codes, and ranking.

According to the fifth step, a feasible course plan must be integrated, picking up services from the candidates. With constraints of 1 TA over 5 students in regular and complementary classes, we cannot give interactive checks for all source codes of many students. We must limit the checks for certain students to make the new course plan feasible. Students belonging to P1 and P2 would join to complementary classes on their will to get supervisions from TAs, because they have intrinsic motivation. Students corresponding to P3 may not ask questions to TAs when they encounter problems, because they have no intrinsic motivation. We should oblige the checks on these students. Through the checks, oral explanations from TAs would help them not only solve problems, but also understand the basic points. The demonstrations to make a successful program during the checks would increase their confidence to continue to learn. Therefore, interactive checks are obliged only for students who fail the test of the week.

5.7 Results and Discussions

We have conducted the programming exercise courses with course plans of 2011 (CP 2011) and 2012 (CP 2012) designed with the PMD method. The course plan differs from the one of 2010 in terms of

- 1 week hard deadline to submit weekly assignments,
- weekly tests,
- interactive checks for failures of every weekly test, and
- announcement of top 10 score ranking.

Different from CP 2011, TAs are encouraged to check source codes as soon as they are submitted in CP 2012.

The progressive learning behaviors in 2011 and 2012 are compared with those in 2010 to see the effectiveness of the PMD method. First, we examine how the enforcement and the facilitation work on all of the students. Assignments are issued every week. We should check how deadlines and weekly tests improve source code submissions. Figure 6 shows how many source codes are submitted on each day after the announcement of the assignments. In CP 2010, the submission, presented by black bars, happens very little within 14 days. The source codes submitted within 14 days is only 37.45% of the total assignments. In contrast, in CP 2011 and 2012, which imposes the students to submit source codes within one week, the submission happens almost every day, as presented with striped bars and gray bars. The total submission rate in 2011 and 2012 are 84.05% and 87.03%, respectively. Earlier check of source codes in 2012 encourages students in earlier submission.

Figure 7 presents the average learning time of students of the three course plans. The students in 2011 and 2012 spend much more time than the ones in 2010. The average learning time of students in 2010, 2011, and 2012 are 205.5 minutes, 257.0 minutes and 288.1 minutes, respectively. The average learning time of the top 15% students in 2010 is 254.5 minutes. It is amazing the average learning time in 2011 is more than that of the top students in 2010. It is also the case in 2012. The dashed line corresponding to CP 2010 increases rapidly at the 13th week, which is almost the end of the semester. In the meantime, the solid line of CP 2011 and the dotted line 2012 do not increase. In 2011 and 2012, the farther the course proceeds,
the shorter the average the learning time is, though the submission rate is higher. That means, in 2011 and 2012, it takes less time for students to solve assignments, as the course proceeds.

Finally, we compare student scores in 2010, 2011, and 2012. We classify students into 4 groups by their total scores of all weeks to examine how the score of each group changes. Figure 8, Fig. 9, and Fig. 10 present the average score of each groups in 2010, 2011, and 2012, respectively. In 2010, all of the lines fall down rapidly from after the mid-term test where students learn difficult matters such as array and structure. In contrast, the lines of the four groups in 2011 and 2012 keep horizontal. All of the students in CP 2011 keep working throughout the course. Because they improve programming skills, they solve assignments using less time as Fig. 7 shows. It is also noteworthy that the score of poor students of 2011 and 2012 rise in the middle of the semester, while that in 2010 goes down rapidly right after the beginning of the semester. The result proves that the course plans revised with the PMD method in 2011 and 2012 can prevent the learners from reducing their scores. The discipline has directed all students to the course goal. The revised course plan gives P3 affective components with weekly tests and hard deadlines, along with supervisions of interactive checks. Since they can get explanations and demonstrations through interactive checks, they attain to solve assignments. The revised course plan gives them beliefs to programming, or expectancy components, which motivate them to learn more. For P2, the revised course plan provides learning strategy using one-week hard deadlines. Since they finish assignment within 1 week, they can get knowledge necessary to solve new assignments in the next week before the new assignments are issued. Since they can solve the new ones, they get confident and interested in programming. To P2, the revised course plan brings expectancy components and task value components. For P1, the revised course plan does not only promote their intrinsic motivation with hard assignments, but also motivates them with extrinsic effects of ranking service. Overall, the revised course plan can enhance motivation for all of the personas, which makes them keep working throughout the course.
6. Conclusions

We propose the PMD method for designing an inclusive and stable introductory programming practice course that motivates all learners throughout the course. It utilizes personas representing learners with similar motivation and learning strategies. The personas are determined from summary learning behaviors taken with the contextual inquiry. The method enforces a discipline on all personas to direct their learning to the course goal. The course designers choose services to facilitate each persona under the enforcement. The method has enlarged the learning time and improved their scores in the successive 2 courses of introductory programming practice for over 500 freshmen in each course in Ritsumeikan university.

References

[1] H. Beyer and K. Holtzblatt, Contextual Design: Defining Customer-Centered Systems, Morgan Kaufmann, 1998.
[2] M. Biggers, T. Yilmaz, and M. Sweat, “Using collaborative, modified peer led team learning to improve student success and retention in intro CS,” ACM SIGCSE Bulletin - SIGCSE ’09, vol.41, no.1, pp.9–13, 2009.
[3] A. Cooper, The Inmates are Running the Asylum: Why High Tech Products Drive Us Crazy and How to Restore the Sanity, Pearson Education, 2004.
[4] S. Cooper and S. Cunningham, “Teaching computer science in context,” ACM Inroads, vol.1, no.1, pp.5–8, 2010.
[5] L.F. Denton, D. McKinney, and M.V. Doran, “A melding of educational strategies to enhance the introductory programming course,” 35th ASEE/IEEE Frontiers in Education Conference, F4G-7, 2005.
[6] W. Dick, The Systematic Design of Instruction, Prentice Hall, 2008.
[7] E.J. Fox, “Contextualistic perspectives,” in Handbook of Research for Educational Communication on Technology (3rd.Ed.), ed. J.M. Spector, M.D. Merrill, J. van Merrienboer, and M.P. Driscoll, Erlbaum Assoc., 2007.
[8] H. Jin, “A hybrid instructional design model for the combination of motivation theory and constructivism,” Proc. 4th International Conference on Computer Science & Education, pp.1652–1656, 2009.
[9] J.M. Keller, Motivational Design for Learning and Performance: The ARCS Model Approach, Springer, 2010.
[10] K.M.Y. Law, V.C.C. Lee, and Y.T. Yu, “Learning motivation in E-learning facilitated computer programming courses,” J. Computer and Education, vol.55, pp.218–228, 2010.
[11] M.M. McGill, “Learning to program with personal robots: Influences on student motivation,” ACM Trans. Computing Education, vol.12, no.1, Article no.4, 2012.
[12] S. Mitra, T.M. Rao, and T.A. Bullinger, “Teaching software engineering using a traceability-based methodology,” J. Computing Sciences in Colleges, vol.20, no.5, pp.249–259, 2005.
[13] P.R. Pintrich and E.V. DeGroot, “Motivational and self-regulated learning components of classroom academic performance,” J. Educational Psychology, vol.82, no.1, pp.33–40, 1990.
[14] P.R. Pintrich, D.A.F. Smith, T. Garcia, and W.J. McKeachie, “A manual for the use of the Motivated Strategies for Learning Questionnaire (MSLQ),” National Center for Research to Improve Post secondary Teaching and Learning, 1991.
[15] R.A. Reiser, Trends and Issues in Instructional Design and Technology, Pearson, 2012.
[16] D.H. Schunk and B.J. Zimmerman, Self-regulated learning: From teaching to self-reflexive practice, Guilford Press, 1998.
[17] R.E. Slavin, Educational psychology: Theory and Practice, 10th, Edition, p.317, Pearson, 2006.
[18] K. Tanigawa, F. Harada, and H. Shimakawa, “Detecting learning patterns during exercise from function call logs,” International Journal of Advanced Computer Science, vol.1, no.1, pp.30–35, 2011.
[19] H. Tsukamoto, H. Nagumo, Y. Takemura, and K. Matsumoto, “Analysing the tradition of learners’ motivation to learn programming,” Proc. 38th Frontiers in Education Conference, S4B, 2008.
[20] E. Yukseturk and S. Bulut, “Predictors for student success in an online course,” Educational Technology & Society, vol.10, no.2, pp.71–83, 2007.

Dinh Dong Phuong received Bachelor of Mathematics and Information science in 2001 in Danang Education University. Received Master of Information Engineering in Ritsumeikan University in 2008, Japan. From 2009, a Ph.D candidate at Ritsumeikan, interested in Education Engineering.

Fumiko Harada received B.E., M.E, and Ph.D degrees from Osaka University in 2003, 2004, and 2007, respectively. She joined Ritsumeikan University as an assistant professor in 2007, and is currently a lecturer. She engages in the research on real-time systems and data engineering. She is a member of IEEE.

Hiromitsu Shimakawa received Ph.D degree from Kyoto Univ. in 1999. He joined Ritsumeikan Univ. in 2002. Currently, he is a professor in Ritsumeikan Univ. His research interests include data engineering, usability, and integration of psychology with IT. He is a member of IEEE and ACM.