Reflective Pedagogical Practice on and in Introduction to Programming and Software Engineering

Steffen Becker
Institute of Software Technology (ISTE)
University of Stuttgart
Stuttgart, Germany
steffen.becker@informatik.uni-stuttgart.de

Christine Bescherer
Institute for Mathematics and Computer Science
University of Education Ludwigsburg
Ludwigsburg, Germany
bescherer@ph-ludwigsburg.de

Andreas Fest
Institute for Mathematics and Computer Science
University of Education Ludwigsburg
Ludwigsburg, Germany
fest@ph-ludwigsburg.de

***Abstract*: In an educational project aimed at introducing reflective pedagogical practice to improve the situation of pre-service teachers in computer science at the University of Stuttgart, the need to restructure the introductory 'Programming and Software Engineering' course for all computer science students arose. So, the reflective pedagogical practice - by the lecturers - led to changes in the course. The first steps of the educational re-design - following a design-based research approach – are describe here.**

Keywords: pre-service teachers in computer science, reflective pedagogical practice, introduction to programming and software engineering

I. Introduction

Pre-service teachers[1] in computer science at the University of Stuttgart have to take the introductory "Programming and Software Engineering" course (PSE) in their 1st semester together with all the other computer science students. For most of these students, especially the ones from Baden-Württemberg, this is the first contact with concepts and strategies for programming.[2]

So, we wanted to use this opportunity, when these future teachers make their own learning experiences with programming, to introduce them to a reflective pedagogical practice [1], [2].

We believe that the pre-service teachers' problems with their first experiences using – formal – programming and software engineering concepts and strategies will be similar to the ones their students will face in the future when they teach at school. So, this point in time at the beginning of the university program is a good opportunity for the pre-service teachers to reflect on their own learning experiences. When they will have finished their university degree in Computer Science Education and will start their work in schools, they will be advanced computer scientists themselves. So, it will probably be much harder for them to remember their first steps into programming including the problems they had.

In a project funded by the Ministry for Science, Research, and Art, Baden-Württemberg the group combined of a software engineer, a computer science educator, a mathematics educator and an experienced PSE tutor, who is also an advanced pre-service teacher, works in a design-based research approach [3] to address this issue.

II. Theoretical Background

A. Reflective Pedagogical Practice

Reflective pedagogical practice was the most important of the concepts, we used to design our intervention. Reflective practice in teacher education means on one hand that pre-service teachers reflect on their in-school-experience in an internship [2]. "Reflective practice is considered necessary if teachers are to learn from their own teaching experiences and the experiences of others." [1 p. 482]. But it can also refer to reflection of one's own learning experiences [4] to identify e.g. what were the obstacles in understanding some topics and what supported the understanding process.

B. Similarities in pre-service mathematics teachers' programs

There is a similar problem for pre-service mathematics teachers called the "double discontinuity" [5] which describes the conceptual changes students face in the transition from high school mathematics to university mathematics, which is much more formal and abstract than school mathematics. After their university degree in mathematics education, when going back to school as teachers they face the second discontinuity: They have to break down their mathematical knowledge to the level in the classroom. In Germany several projects and initiatives aimed to improve these situations ([6], [7], [8]). Further, in most German universities there are no special lectures in mathematics for pre-service teachers but they attend the same Calculus or Analytical Geometry courses as (future) research mathematicians.

Some of the recommendations of the above-mentioned initiatives to deal with the problems arising out of the double discontinuity are to group the pre-service teachers in special groups for the tutorials[3] and to substitute one of the mathematics problems on the weekly worksheets by special learning or teaching related problems [7]. According to Ableitinger [7] these pre-service-teacher oriented task could either use typical school problems as a way to connect the

[1] ‚Pre-service teacher' means students in pre-graduate or graduate teacher programs compared to ‚in-service-teachers', who are already working in schools.

[2] In the state of Baden-Württemberg computer science as a compulsory subject in secondary schools only started recently and with only one hour a week in 7th grade.

[3] In Germany mathematics lectures are usually accompanied by small group tutorials, where students work on or present their solutions of the weekly worksheets.

In the subsequent task from a mathematics contest for students ("Känguru der Mathematik 2009") the participants were asked to solve which of the following figures is the greatest one.

(A) $\sqrt{2}-\sqrt{1}$ (B) $\sqrt{3}-\sqrt{2}$ (C) $\sqrt{4}-\sqrt{3}$ (D) $\sqrt{5}-\sqrt{4}$ (E) $\sqrt{6}-\sqrt{5}$

A student in grade 12 chose answer (E) and stated:

"$\sqrt{6}-\sqrt{5}$ is the greatest figure, because roots are monotone. So, the greater is x, the greater is f(x). Thus, their difference is the greatest, as well (by going more to the right)."

1. Analyze the student's answer. Where do you see problems with the argumentation?

2. Provide an own student-oriented answer to this topic.

3. Show in general: $\lim_{n\to\infty} \sqrt{n} - \sqrt{n-1} = 0$

Fig 1: Example of pre-service-specific math problem [9]

available mathematical knowledge to issues in university mathematics. Figure 1 shows an example of this approach. There the necessity to prove the theorem given in the last part is "grounded" in a typical math problem from high school. On the other hand, these pre-service specific tasks can use the university mathematics viewpoint e.g. to better understand the different levels of abstraction or the coherence of different mathematical topics. For example, while discussing the characteristics of – abstract – vector spaces, the pre-service teachers could identify examples of this mathematical structure used in school mathematics like the 3-dimensional coordinate space, the complex numbers or even function space.

III. Realizations

The aim of the project is to improve the situation of the pre-service computer science teachers in the PSE course. So, we planned these measures:

- all pre-service teachers are gathered in two extra tutorial groups[4]
- these tutorials are conducted by an advanced pre-service teacher
- in each worksheet there is one dedicated extra task asking for a reflective pedagogical practice

In the discussions while preparing these tasks, the project group realized that it is not enough to change one task per worksheet, but that the structure of the course and the content of the different lectures were also not very supportive to learning programming and software engineering regardless whether the students were pre-service teachers or not. So, we went through all the lectures to identify and name the learning aims, to sort the prerequisites and the necessary definitions for the students to understand the topics.

Examples of these changes include that we added explicit learning objectives at the beginning of each lecture focusing on the programming related concepts and definitions. These learning objectives play the role of a table of contents of the lecture which makes it easier for the students to get an idea of the extend of concepts taught in the lecture. Following that information, we now provide insights into the didactical[5] idea of the lecture in order to make it transparent to the students. For example, we use the Hamster simulator by D. Bohles (http://boles.de/hamster/simulator.html) to start with a miniature language and a miniature world to mitigate the effect of being overwhelmed by the sheer number of concepts used in programming. Also, it adds a gamification aspect to the exercises on the worksheets to increase the students' motivation to explore programming on their own.

We also apply the objects-first teaching [12] approach which focuses in the first lectures on using and creating objects. This shifts the usual algorithmic topics some weeks towards the lecture's end. Students who have prior experience in programming often have algorithmic experience. We explain to the students, why we do not start with algorithms: The reason is that concepts like branches or loops are not as important in OOP as students usually think - based on their prior knowledge and experience.

However, already in the previous year we learnt that the students do not see why teaching objects-first is of relevance. In the lecture feedback forms they said the course is not teaching what they believed is of most importance. Explaining in detail why objects are taught first and why this has some advantages aims at increasing the acceptance within the students. As a first effect, it has already increased the acceptance of this didactical method among the tutors which start to favor it now. In the meantime, also the student evaluation of the second instance of the lecture has taken place. In the feedback forms we can see a significant trend that the students understood much better why teaching objects first is an interesting and useful didactical approach. The number of students complaining about object firsts has decreased to outliers (less than 5 students in a group of almost 200). Their main reason for rejecting the approach is often based on insufficient tooling support (BlueJ as IDE in contrast to Eclipse or IntelliJ). We consider this not as fundamental issues with the objects first approach, but rather with its implementation.

Finally, the lectures have been enhanced with a competence-oriented list of skills and capabilities the students should have once they fully understood the contents of the lecture. In the last semester, we observed that a large number of students learned a concept's definition but were unsure about what level of application they should be able to do in

[4] Usually there are less than 20 pre-service teachers attending PSE and about 400 students of computer science plus 100-200 students taking PSE as imported subject.

[5] ‚Didactical' is used here in the European understanding and is similar to the concept of Shulman's 'pedagogical content knowledge' [10].

exams. For example, a large number of students was able to define what a type of a variable is conceptually, but were unable to derive the types of variables in a given small program fragment. The newly structured lecture content now is very explicit in this by naming the expected learned competences and the level of competence we expect for passing the course.

The following examples give an idea of the of the pre-service teacher specific tasks in the worksheets from wintersemester 2019/20:

As stated above - beside adding information about educational aspects to the lectures - we designed exclusive tasks for pre-service teachers. As this is still an ongoing process we give some highlights of the tasks we added to the worksheets of this lecture's instance. At the time of writing roughly 60% of all worksheets of the semester have been designed and used. One task focused on our "challenging assignments". Challenges have been added to all worksheets to offer some interesting tasks also for those students who already have prior programming experience as they complete the worksheet in half the time of the beginners. This is a common challenge in teaching programming as often teachers will face students who learned programming before, e.g., in a self-taught manner. In the pre-service teachers' worksheet, we make the future teachers aware of this problem by making them reflect on their own different levels of experience and discuss the pros and cons of this approach.

In the following worksheets, the pre-service teachers are supposed to reflect the gamification approach using the Hamster simulator and the objects-first approach. They will have to discuss how they experienced their own learning using these didactical elements of the lecture. They can base their discussions on our explicit information about the designed didactical concept of each lecture. One specific task was to reflect on the differences between the Hamster simulator mini world and Kara the ladybug's mini world [6]. There are differences in the complexity, the number of supported languages, and also in the objects first vs. algorithms first way of teaching. We expected the pre-service teachers to identify and analyze those differences.

Finally, in even further worksheets we had the pre-service teachers reflect on algorithmic problem solving vs. object-oriented problem solving. They should list the pros and cons of using the objects-first vs. the algorithms-first approach at school and make up their mind which approach they would like most based on their own experiences.

As part of the latter, we had a task for the pre-service teachers to discuss the pros- and cons of using a simplified IDE for objects first (BlueJ in our case). They had to contrast it to using another simple IDE, a more complex IDE or no IDE at all. This task was supposed to make the pre-service teachers explicitly reflect on the role IDEs play in learning to program. The idea was to contrast hiding a lot of technical details in the IDE vs. seeing and potentially understanding them when using no IDE or a complex IDE.

Another task for the pre-service teachers was to compare their experience in learning a computer science topic to the experiences they had in learning other, but related topics (math, physics, etc.). The idea was to identify and utilize the positive aspects in teaching computer science and to mitigate the negative aspects of it.

Contracts and design-by-contract are important aspects of the objects first teaching approach. Hence, we made the pre-service teachers to reflect on the use of contracts in a (potentially more advanced) course at school. The objective of the task was for them to come up with ideas how to teach a rather complex topic in such an advanced course in school. In detail, the pre-service teachers were asked how they would find an abstraction of the concept or how they would design simple examples which convey the motivation of contracts in programming.

As we used the hamster simulator's mini world to introduce gamification in our exercises, we asked the pre-service teachers to reflect on their experiences in doing game-based exercises. In addition, they were asked to survey other game-based teaching methods which already exist and can be used in schools (like writing games in Scratch, using teaching related smartphone apps, etc.)

As part of teaching OOP, the pre-service teachers had to learn the concept of inheritance. As this is another more advanced topic, we made them again reflect on their experiences in understanding inheritance. Based on this, we asked them again to come up with ideas whether and how to introduce inheritance in the context of a school's course. In addition, we asked them to design small tasks and exercises which potentially could be used in such a course.

In the last exercise designed so far, we added a task about how to introduce and illustrate the usefulness of debugging. As debugging is often a task which involves many and often overwhelming features of an IDE, the main objective was how to illustrate debugging to students. The hamster and ladybug mini worlds provide already partial answers in their integrated IDEs. These approaches make the idea and usefulness of debugging much more tangible.

IV. First results

As the course is still ongoing with the redesigns sketched above it is too early for an objective assessment. However, from a subjective point of view based on observations made in the lecture and tutorial groups the implemented improvements have a positive impact. Students are more motivated and seem to be less lost in the large amount of information a lecture at a university is able to provide. The pre-service teachers' tutorial groups find the additional exercises interesting. Before, they were treated as any other student in the course Programming and Software Development, the question how to teach the subject has not been addressed and remained implicit. Because of this they also did not develop an identity as a group and complained a lot that they were confronted with teaching related questions only from the third semester onwards – and then it was disconnected from their own experiences. Now they get a pre-service teacher related view on the topics right from the first lecture in their computer science topic.

V. Further steps in the design-based research process

The whole project is an example of design-based educational research [11]. After identifying the 'problem' – in our case the difficulties especially the pre-service teachers

[6] https://www.swisseduc.ch/informatik/karatojava/kara/

have with the course Introduction to Programming and Software Engineering – some theoretical input is collected. Then the design-based research cycle (s. fig 2) will be followed through.

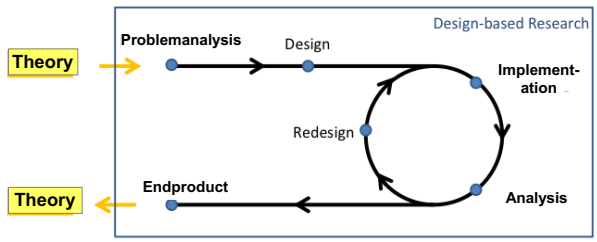

Diagramm according to Nando Stöcklin: http://educationaldesignresearch.de/wasistedr/

Fig. 2: Design-based Educational Research Cycle to develop new learning scenarios.

Some of the theoretical inputs are described above and the first implantation of the course is taking place during the winter semester 2019/20. Beside the detailed comparison of the evaluation results (s. above) we will ask students to give a formative feedback on the improvements using the Teaching Analysis Poll method [13], questionnaire. Additional interviews with a few pre-service teachers and the tutors will be conducted. According to the results of these feedbacks the redesign of the course Introduction to Programming and Software Engineering will take place and be run again. If the evaluation results of this run will be satisfying, the whole concept will be described in detail and disseminated further.

REFERENCES

[1] Lane, R., McMaster, H., Adnum, J., & Cavanagh, M. (2014). Quality reflective practice in teacher education: A journey towards shared understanding. Reflective Practice, 15(4), 481-494. DOI: 10.1080/14623943.2014.900022

[2] Power, A., Clarke, M., & Hine, A. (2002). Internship learning connects the dots: The theory and practice of reflection. Online at https://www.aare.edu.au/data/publications/2002/cla02481.pdf

[3] Easterday, M. W., Lewis, D. R., & Gerber, E. M. (2014). Design-Based Research Process: Problems, Phases, and Applications. In: Polman, J., Kyza, E., O'Neill, D., Tabak, I., Penuel, W. Jurow, A., O'Connor, K., Lee, T. & D'Amico, L. (eds.). International Conference of the Learning Sciences (ICLS) 2014, (1), 317-324. https://doi.dx.org/10.22318/icls2014.317

[4] Graham, A. & Phelps, R. (2003). Being a teacher: developing teacher identity and enhancing practice through metacognitive and reflective learning processes, Australian Journal of Teacher Education, vol. 27, no. 2, pp. 11-24. Online at https://ro.ecu.edu.au/ajte/vol27/iss2/2/

[5] Kilpatrick, J. (2019). A Double Discontinuity and a Triple Approach: Felix Klein's Perspective on Mathematics Teacher Education. In: H.-G. Weigand, W. McCallum, M. Menghini, M. Neubrand, G. Schubring (Eds.). The Legacy of Felix Klein. Springer, Cham, 215-226.

[6] Buchholtz, N., & Kaiser, G. (2013). Improving mathematics teacher education in Germany: Empirical results from a longitudinal evaluation of innovative programs. International Journal of Science and Mathematics Education, 11(4), 949-977.

[7] Ableitinger, C., Hefendehl-Hebeker, L. & Herrmann, A. (2013). Aufgaben zur Vernetzung von Schul- und Hochschulmathematik. In: H. Allmendinger, K. Lengnink, A. Vohns, G. Wickel (Eds.). Mathematik verständlich unterrichten. Perspektiven für den Unterricht und Lehrerbildung. Wiesbaden: Springer Spektrum, 217-233.

[8] Beutelspacher, A., Danckwerts, R. & Nickel, G. (2011). Mathematik Neu Denken. Empfehlungen zur Neuorientierung der universitären Lehrerbildung im Fach Mathematik für das gymnasiale Lehramt. Online at https://www.uni-siegen.de/fb6/didaktik/personen/rainer-danckwerts/resources/empfehlungen_mathematik_neu_denken.pdf, last retrieved 10/11/2019

[9] Isaev, V. & Eichler, A. (2017). Measuring beliefs concerning the double discontinuity in secondary teacher education. CERME 10, Feb 2017, Dublin, Ireland. hal-01949039.

[10] Berry, A., Friedrichsen, P., & Loughran, J. (Eds.). (2015). Re-examining pedagogical content knowledge in science education. Routledge.

[11] Van den Akker, J., Gravemeijer, K., McKenney, S., & Nieveen, N. (2006). Introducing education design research. In J. V. D. Akker, K. Gravemeijer, S. McKenney & N. Nieveen (Eds.), *Educational design research*. New York: Routledge. 3-7.

[12] B. Meyer (2013). Touch of Class, Springer Verlag

[13] Frank, A. & Kaduk, S. (2017). Lehrveranstaltungsevaluation als Ausgangspunkt für Reflexion und Veränderung. Teaching Analysis Poll (TAP) und Bielefelder Lernzielorientierte Evaluation (BiLOE). In Arbeitskreis Evaluation und Qualitätssicherung der Berliner und Brandenburger Hochschulen und Freie Universität Berlin (Hrsg.), QM-Systeme in Entwicklung: Change (or) Management? Tagungsband der 15. Jahrestagung des Arbeitskreises Evaluation und Qualitätssicherung der Berliner und Brandenburger Hochschulen am 2./3. März 2015, Freie Universität Berlin. 29-51.