MATHEMATICAL DISCOVERIES USING COMPUTATIONAL THINKING

Christine Bescherer and Andreas Fest

University of Education Ludwigsburg, bescherer@ph-ludwigsburg.de, fest@ph-ludwigsburg.de

For about 40 years now programming is used in mathematics classrooms as a constructivist concept of learning mathematics. Preservice teachers at the University of Education Ludwigsburg showed great difficulties in implementing this concept in a third grade mathematics classroom. Therefore we propose a framework for mathematical discoveries using computational thinking based on frameworks for inquiry-based learning and learning of computational thinking.

Keywords: Inquiry-based mathematics learning, computational thinking, mathematical discoveries

INTRODUCTION

The 'hype' of introducing computational thinking in primary and secondary classrooms has reached Germany for some time now. In a design-based-research project 'Digital Learning in Primary Schools Stuttgart / Ludwigsburg' ('Digitales Lernen in der Grundschule Stuttgart / Ludwigsburg' dileg-SL) at the Ludwigsburg University of Education funded by Deutsche Telekom Stiftung we tried to combine the learning of mathematical concepts with computational thinking. The project as a whole aimed at the development of digital learning scenarios at primary schools. Beneath the productive and critical exposure to digital media in different contexts and subjects like German and English language, mathematics, biology, music, physical education, another important objective was the development of primary schoolchildren's basic competencies in computer science and algorithmic thinking.

In the mathematics sub-project, 25 preservice teachers learnt in two special mathematics education seminars about 'computational thinking' and the use of programming to understand mathematical concepts. Then they had to use this knowledge to develop learning scenarios for third grade classrooms fostering the understanding of mathematical concepts or the development of mathematical mental models. The learning scenarios covered topics like patterns with regular polygons or orientation in a plane. To support the school children developing these mental models they had to work with the programming language Scratch (https://scratch.mit.edu/). These scenarios were tested in two 90 min sessions in a 3rd grade classroom and afterwards reflected on by the whole group.

We found that all preservice teachers had a very big problem in common: They all tended to forget about the mathematics in their learning scenarios. Even the constant reminders 'that we are supposed to teach mathematics' and a planning structure where they had to describe the mathematical objectives and background extensively didn't help. The newness of programming in a mathematics classroom and the 'fun things' which are possible with Scratch made it very difficult for the preservice teachers not to ignore the mathematics.

Therefore, we want to do research on the topic: How to get preservice teachers to succeed in creating learning scenarios for inquiry-based mathematics learning using coding. To answer this question, we will follow the design-based research

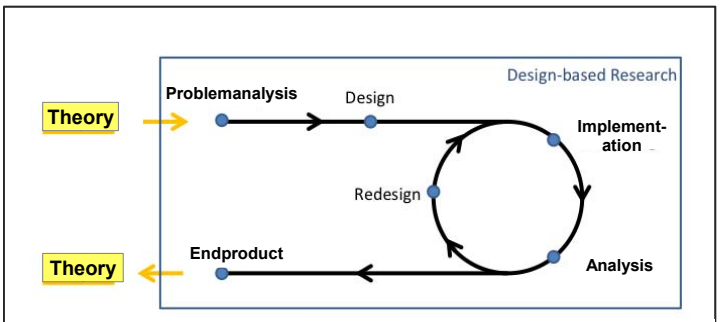

Figure 1: Design Based Research Approach

approach (cf. Easterday, Lewis, & Gerber, 2014). In the design-based research approach theory is an important input as well as an important output (s. Fig. 1). For our study we need a theoretical framework which guides students during the creation of the learning scenarios, works for the evaluation of the scenarios as well as can be used for measuring the competences the preservice teachers acquire during the seminar. The focus in this paper will be on the development of the theoretical framework, on which the study will be based.

THERETICAL BACKGROUND

First, we describe the learning concepts we based our project on and then we will introduce our framework.

Inquiry-based-learning in mathematics

Following the constructivist approach to mathematics learning, learners have to build their own network of knowledge and mathematical competences which include mathematical processes like mathematical problem solving or reasoning (NCTM, 2000). So teaching mathematics in a way that allows learners to develop their own mathematical competencies requires different pedagogical approaches than traditionally used in mathematics classrooms. 'Inquiry-based learning is a more student-centered way of learning and teaching, in which students learn to inquire and are introduced to mathematical and scientific ways of inquiry.' (Maaß & Artigue, 2011, p. 779)

Common to most definitions of inquiry-based-learning is that it is learner-centred and lets learners experience how real scientist in the field work. Pedaste et al. (2015) give a very good definition at the beginning of their paper:

> 'Inquiry-based learning is an educational strategy in which students follow methods and practices similar to those of professional scientists in order to construct knowledge. It can be defined as a process of discovering new causal relations, with the learner formulating hypotheses and testing them by conducting experiments and/or making. Often it is viewed as an approach to solving problems and involves the application of several problem solving skills. Inquiry-based learning emphasizes active participation and learner's responsibility for discovering knowledge that is new to the learner. In this process, students often carry out a self-directed, partly inductive and partly deductive learning process by doing experiments to investigate the relations for at least one set of dependent and independent variables.' (Pedaste et al., 2015, p. 48)

Inquiry-based learning is not the only constructivist, student-centred approaches in mathematics teaching and they all have a lot in common (Artigue & Blomhøj, 2013). Discovery learning i.e. 'occurs whenever the learner is not provided with the target information or conceptual understanding and must find it independently and with only the provided materials.' (Alfieri, Brooks, Aldrich, & Tenenbaum, 2011, p. 5). However, the 'pure' discovery without any guidance of the teacher has limited effects. It is suggested that …

> 'overall, the effects of unassisted-discovery tasks seem limited, whereas enhanced-discovery tasks requiring learners to be actively engaged and constructive seem optimal. Based on the current analyses, optimal approaches should include at least one of the following:
>
> 1) guided tasks that have scaffolding in place to assist learners,
>
> 2) tasks requiring learners to explain their own ideas and ensuring that these ideas are accurate by providing timely feedback, or
>
> 3) tasks that provide worked examples of how to succeed in the task.' (Alfieri et al, 2011, p. 35)

We want the mathematics learners to make mathematical discoveries using programming, but we choose the inquiry-based learning framework because it is closer to our purpose. Students in schools rather re-discover mathematical knowledge, which is already in the world, than make a 'free discovery'. Further the teacher decides on the pedagogical approach and he or she plans the aims, problems, supporting tools and materials for making the discoveries. Very often the teacher also decides on the questions which are to be investigated (Dobbler, Zwart, Tanis & van Oers, 2017). So, we decided to choose the inquiry-based-learning framework by Pedaste et al. (2015).

Pedaste et al. (2015) developed the framework from a systematic literature review of 32 papers on inquiry-based learning. Figure 2 gives an overview of the framework:

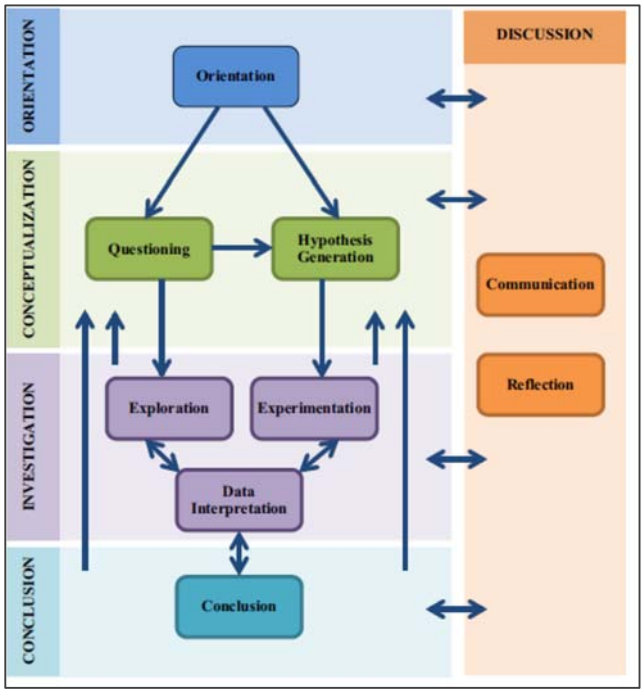

Fig. 2: Inquiry-based learning framework (general phases, sub-phases, and their relations), Pedaste et al., 2015, p. 56)

There are different pathways through this framework depending on the kind of questions and the approach (hypothesis-, question-, or data-driven).

Computational thinking

'Computational thinking' is a rather inflationary used concept, which is still defined in a wide variety. Wing (2006) describes 'conceptualization' and 'thinking on multiple levels of abstraction' as important aspects of computational thinking. Grover & Pea (2013) give in their review of the state of the field on computational thinking in K-12 a list with different elements of computational thinking i.e. 'abstractions and pattern generalizations (including models and simulations), systematic processing of information, symbol systems and representations, algorithmic notions of flow of control, conditional logic, … debugging and systematic error detection' (Grover & Pea, 2013, p. 39-40). Obviously, these elements cover very different levels of knowledge: some are factual knowledge (i.e. conditional logic), some are basic concepts of computer science thinking (i.e. algorithmic notions of flow of control) as well as typical procedures computer scientists use (like debugging and systematic error detection). Therefore, these elements are not a list to go through step by step to master computational thinking, rather the factual knowledge and basic concepts should be understood and deepened by working like a typical computer scientist.

Brennan & Resnick (2012) describe a framework of three dimensions '*computational thinking concepts* (the concepts designers engage with as they program' such as sequences, loops, events, iteration, parallelism, conditionals, operators, data), '*computational thinking practices* (the practices designers develop as they engage with the concepts' such as being incremental and iterative, testing and debugging, reusing and remixing, abstracting and modularizing), 'and *computational perspectives* (the perspectives designers form about the world around them and about themselves' such as expressing, connecting, questioning). (Brennan & Resnick, 2012, p.1).

Computational thinking concepts are similar to mathematical concepts (i.e. regular polygons, limits, functions, patterns,…) which have to be learned specifically and they also require specific methodological approaches. The computational thinking practices on the other hand are more like tools to do 'real' programming. This means that a programmer uses all these practices and very often different tools can be used to solve the same problem.

Specific pedagogical approaches to learn computational thinking concepts already start in primary school, i.e. Gadanidis, Hughes, Minniti, & White (2017) describe a concept for introducing computational thinking in primary mathematics classrooms where grade 1 pupils experience several aspects of computational thinking. Kotsopoulos, Floyd, Khan, Namukasa, Somanath, Weber, & Yiu, (2017) describe a *pedagogical framework for computational thinking* (CTPF) which

'includes four pedagogical experiences: (1) unplugged, (2) tinkering, (3) making, and (4) remixing. Unplugged experiences focus on activities implemented without the use of computers. Tinkering experiences primarily involve activities that take things apart and engaging in changes and/or modifications to existing objects. Making experiences involve activities where constructing new objects is the primary focus. Remixing refers to those experiences that involve the appropriation of objects or components of objects for use in other objects or for other purposes. Objects can be digital, tangible, or even conceptual.' (Kotsopoulos et al. 2017, p. 154)

The pedagogical framework for computational thinking (CTPF) as shown in fig. 3 was 'intended to provide a preliminary lens for structuring teaching and learning for students by considering how to teach computational thinking (Kotsopoulos et al. 2017, p. 168).'

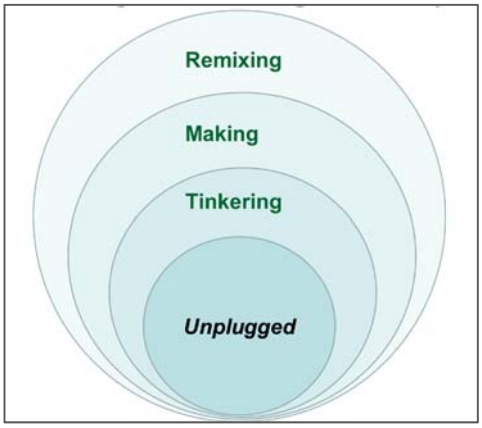

Figure 3: Pedagogical Framework for Computational Thinking (Kotsopoulos et al. 2017, p. 159)

Figure 3 also shows that these experiences follow a certain sequence. The *unplugged experience* is usually at the beginning of a teaching unit and no 'real' computer is used.[1] For this experience a little Play Mobil figure with an attached pen or even a Bee Bot (Highfield, Mulligan, & Hedberg, 2008) which can be 'programmed' using the arrows on his back to drive to certain spot (s. fig. 4) can be used.

Fig. 4: Tools for unplugged experiences

'*Tinkering experiences* primarily involve taking things apart and engaging in changes and/or modifications to existing objects.' (Kotsopoulos et al. 2017, p. 160). The tinkering experience is similar to the worked examples approach used in mathematics teaching (Renkl, 2002, Scherrmann, 2017) where example solutions of complex mathematical

[1] To use ‚unplugged' games, materials and situations for teaching computer science is a teaching concept used for quite some time, s. https://csunplugged.org/en/

problems are given, sometimes with mistakes, which the students have to identify, or as a cloze, which the students have to complete. This is also connected to the discovery learning (s. above). Both these experiences follow the rule that you should never start learning to program with an empty screen.

In the *making experience* student build new objects/programs. 'Making experiences, depending on the objects used, require deeper knowledge than tinkering where objects for the most part are already constructed or pre-existing. In making experiences, students are required to problem-solve, make plans, select tools, reflect, communicate, and make connections across concepts.' (Kotsopoulos et al. 2017, p. 162).

'*Remixing experiences* involve sharing (intentionally or through hacking) an object and modifying or adapting it in some way and/or embedding it within another object to use it for substantially different purposes. Remixing requires a significant level of proficiency to identify a useable object and then adapt and modify it to suit new purposes.' (Kotsopoulos et al. 2017, p. 165)

For the ScratchMath project (https://www.ucl.ac.uk/ioe/research/projects/scratchmaths) a 'framework for action' to bring together primary programming and mathematics was developed by Benton, Hoyles, Kalas & Noss (2017). It consists of the '5Es' – 'explore', 'explain' 'envisage', 'exchange' and 'bridgE' which scaffold the design of learning scenarios combining programming and mathematics learning. These '5E' are described as follows:

> *'Explore*: Opportunities should be provided for pupils to investigate ideas by trying things themselves and debugging errors.
>
> *Envisage*: Pupils should be encouraged to predict outcomes before running scripts and then reflect on the actual outcome.
>
> *Explain:* Opportunities should be provided for whole-class discussions led by teachers as well as with peers through the inclusion of reflective questioning.
>
> *Exchange*: Meaningful opportunities to share and build on others' ideas should be included.
>
> *bridgE:* The links with the primary mathematics curriculum as a powerful idea should be made explicit. (Benton et al., 2017, p. 122-123)

Using these pedagogical concepts for learning mathematics respectively learning computational thinking we propose the new framework for *mathematical discoveries using computational thinking*.

FRAMEWORK FOR MATHEMATICAL DISCOVERIES USING COMPUTATIONAL THINKING.

The framework as shown in fig. 5 will be explained using the example of discovering the properties of regular polygons. The start of a mathematical discovery is a mathematical object (i.e. a square as one example of a regular polygon). Using the unplugged experience (CTPF) the children explore what are the defining parts to draw a square. (i.e. Repeat 4 times: Go forward a distance of 50, turn right 90°). Then the question is given by the teacher: Will it be the same with all regular polygons? So now the children have to inquire about the mathematical properties of regular polygons. The goal is to find the minimal defining information for a plane figure. (In the case of the regular polygons this would be the number of edges and the length of one side.). To understand these properties of the regular polygons the children tinker with given programs or make their own (CTPF), but they always have to predict the expected outcome of their program (envisage, 5 E).

The exploring (5E) and experimentation to find the mathematical properties is accompanied by explanations (5E) and exchanges (5E) among the children and/or with the teacher. When the properties of the mathematical object are verified the bridge (5E) to the results of the mathematical discoveries – the mathematical concepts or mental models – will be built.

The remixing experience (CTPF) is not included in the framework because it requires high level abstraction. So remixing is not forbidden, but the question arises whether the cognitive effort of remixing programs is adequate to the mathematical discovery.

Using the inquiry-based learning framework we added the specific experiences and aspects of the two pedagogical frameworks for computational thinking. The main objective is always the mathematical one and the computational thinking parts are chosen and used to support the discovery of the mathematical concepts/ development of the mental models. To do it the other way around – using mathematical concepts to support the understanding of computational thinking – could be also a fascinating research question – for another time.

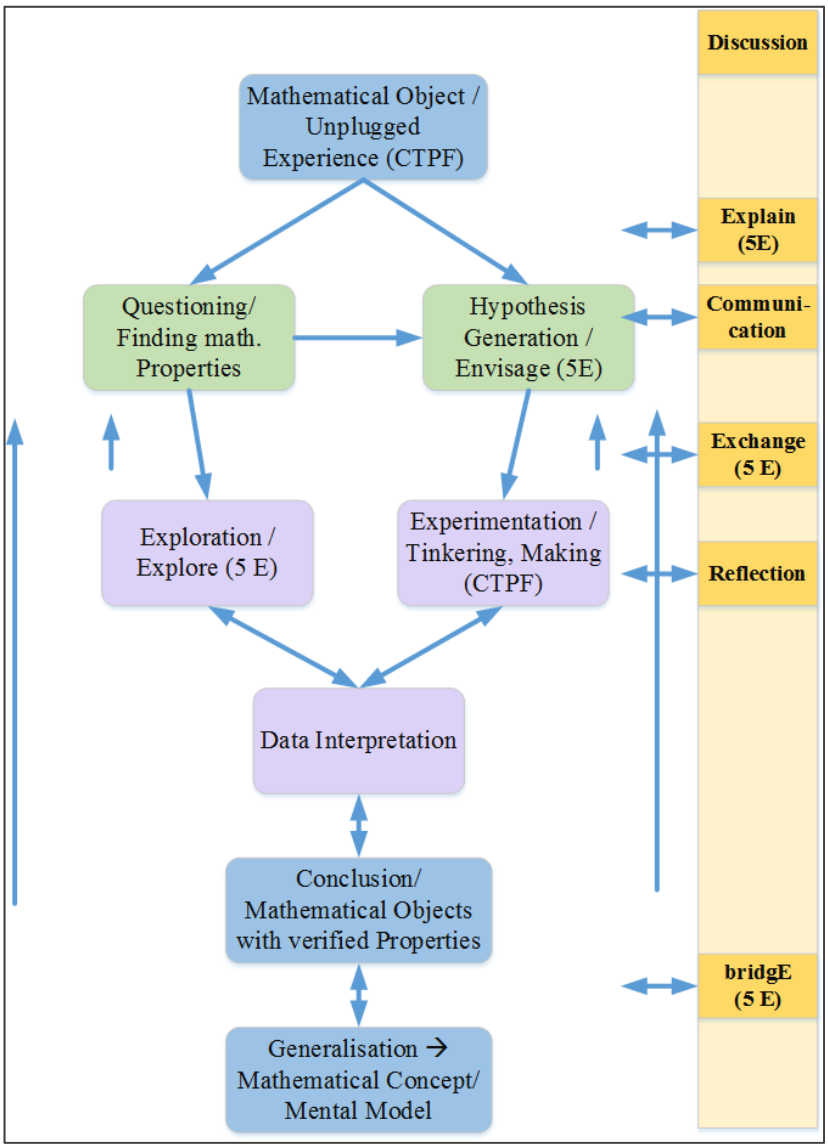

Fig. 5. Framework for Mathematical Discoveries Using Computational Thinking

EVALUATION USING A DESIGN BASED RESEARCH APPROACH

This is a framework which originated from the observation of preservice teachers' difficulties to develop learning scenarios for mathematical concepts using programming. Our overall objective is to develop seminars at university level, which allows teacher students to master the development of learning scenarios for inquiry-based mathematics learning using programming. For that we follow the design-based research (Anderson & Shattuck, 2012) approach in several cycles (s. fig. 1).

The Framework for Mathematical Discoveries Using Computational Thinking (MaDUCT) will on three levels: As guidelines for the development of the learning scenarios, the evaluation of the

learning scenarios as well as for the description/measuring of the competencies the teacher students are supposed to acquire in the university seminar.

The next steps of our study will be:

- Development of best practice examples: The authors will create several learning scenarios for inquiry-based mathematics learning using programming for primary and secondary classes using the MaDUCT-framework. These scenarios will be implemented and evaluated in schools by different preservice teachers in their master thesis.
- Development of an evaluation rubric on the quality of learning scenarios for inquiry based mathematics learning using programming.
- Selection, adaption and development of questionnaires und interview guidelines for the teacher students to document their understanding of the framework and its application in mathematics classrooms.
- Design of the teaching concepts for the university seminar on inquiry-based mathematics learning using technology. Conducting a pilot seminar at the Ludwigsburg University of Education and analysis of the students increase in competencies and of the quality of the learning scenarios developed by the teacher students.
- Redesign of the seminar according to the results of the evaluation.
- Implementation of the seminar at the Ludwigsburg University of Education and dissemination of the concept.
- Publishing the evidence-based theoretical framework on learning scenarios for inquiry based mathematics learning using programming.

CONCLUSION AND OUTLOOK

The process from identifying a 'problem' to developing an intervention which could solve the problem in a university context takes several semesters. But using a design based research approach also allows us to development evaluation instruments, best practice examples and new theories, which makes the effort worth it.

REFERENCES

Alfieri, L., Brooks, P. J., Aldrich, N. J., & Tenenbaum, H. R. (2011). Does discovery-based instruction enhance learning?. *Journal of educational psychology*, *103*(1), 1-18.

Anderson, T., & Shattuck, J. (2012). Design-Based Research: A Decade of Progress in Education Research? *Educational Researcher*, 41(1), 16–25. https://doi.org/10.3102/0013189X11428813

Artigue, M. & Blomhøj, M. (2013). ZDM Mathematics Education 45: 797. https://doi.org/10.1007/s11858-013-0506-6

Benton, L., Hoyles, C., Kalas, I. & Noss, R. (2017) *Digital Experiences in Mathematics Education* 3: 115-138. https://doi.org/10.1007/s40751-017-0028-x

Dobber, M., Zwart, R., Tanis, M., & van Oers, B. (2017). Literature review: The role of the teacher in inquiry-based education. *Educational Research Review*, *22*, 194-214. https://doi.org/10.1016/j.edurev.2017.09.002.

Easterday, M. W., Lewis, D. R., & Gerber, E. M. (2014). Design-based research process: Problems, phases, and applications. In. *Proc. of International Conference of Learning Sciences 2014.* Online at https://repository.isls.org/bitstream/1/1130/1/317-324.pdf .

Gadanidis, G., Hughes, J. M., Minniti, L. & White, B. J. (2017) Computational Thinking, Grade 1 Students and the Binomial Theorem. *Digital Experiences in Mathematics Education*. 3, 77–96. https://doi.org/10.1007/s40751-016-0019-3.

Grover, S. & Pea, R. (2013). Computational Thinking in K–12 A Review of the State of the Field. In Educational Researcher, 42. (1). 38-43.

Highfield, K., Mulligan, J., & Hedberg, J. (2008). Early mathematics learning through exploration with programmable toys. In *Proceedings of the Joint Meeting of PME* (Vol. 32, pp. 169-176).

Kotsopoulos, D., Floyd, L., Khan, S., Namukasa, I. K., Somanath, S., Weber, J., & Yiu, C. (2017). A Pedagogical Framework for Computational Thinking. *Digital Experiences in Mathematics* (3) 154-171. https://doi.org/10.1007/s40751-017-0031-2.

NCTM (2000) *National Council of Teachers of mathematics: Principles and Standards for School mathematics.* Reston, Virginia, USA.

Pedaste, M., Mäeots, M., Siiman, L. A., De Jong, T., Van Riesen, S. A., Kamp, E. T., ... & Tsourlidaki, E. (2015). Phases of inquiry-based learning: Definitions and the inquiry cycle. *Educational research review*, 14, 47-61. https://doi.org/10.1016/j.edurev.2015.02.003

Renkl, A. (2002). Worked-out examples: Instructional explanations support learning by self-explanations. *Learning and instruction* 12(5), 529-556 (2002)

Scherrmann, A. (2017). Learning with worked examples – how does it work in a real classroom setting? *Proceedings 10th Congress of European Research in Mathematics Education*.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM* 49(3), 33-35.

DOI: 10.17185/duepublico/70804
URN: urn:nbn:de:hbz:464-20191126-152456-0