University of Groningen

Defining and Observing Modeling and Simulation in Informatics

Grgurina, Natasa; Barendsen, Erik; Zwaneveld, Bert; van Veen, Klaas; Suhre, Cor

Published in:

Informatics in schools : improvement of informatics knowledge and perception

DOI:

[10.1007/978-3-319-46747-4_11](https://doi.org/10.1007/978-3-319-46747-4_11)

IMPORTANT NOTE: You are advised to consult the publisher's version (publisher's PDF) if you wish to cite from it. Please check the document version below.

Document Version

Publisher's PDF, also known as Version of record

Publication date:

2016

[Link to publication in University of Groningen/UMCG research database](#)

Citation for published version (APA):

Grgurina, N., Barendsen, E., Zwaneveld, B., van Veen, K., & Suhre, C. (2016). Defining and Observing Modeling and Simulation in Informatics. In A. Brodnik, & F. Tort (Eds.), *Informatics in schools : improvement of informatics knowledge and perception : Proceedings* (pp. 130-141). (Lecture Notes in Computer Science; Vol. 9973). Springer. https://doi.org/10.1007/978-3-319-46747-4_11

Copyright

Other than for strictly personal use, it is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), unless the work is under an open content license (like Creative Commons).

The publication may also be distributed here under the terms of Article 25fa of the Dutch Copyright Act, indicated by the "Taverne" license. More information can be found on the University of Groningen website: <https://www.rug.nl/library/open-access/self-archiving-pure/taverne-amendment>.

Take-down policy

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.

Downloaded from the University of Groningen/UMCG research database (Pure): <http://www.rug.nl/research/portal>. For technical reasons the number of authors shown on this cover page is limited to 10 maximum.

Defining and Observing Modeling and Simulation in Informatics

Nataša Grgurina^{1(✉)}, Erik Barendsen², Bert Zwaneveld³,
Klaas van Veen¹, and Cor Suhre¹

¹ Teaching and Teacher Education, University of Groningen,
Groningen, The Netherlands

{n.grgurina,klaas.van.veen,c.j.m.suhre}@rug.nl

² Radboud University and Open University, Nijmegen, The Netherlands
e.barendsen@cs.ru.nl

³ Open University, Heerlen, The Netherlands
g.zwaneveld@uu.nl

Abstract. Computational Thinking (CT) is gaining a lot of attention in education. In this study we focus on the CT aspect modeling and simulation. We conducted a case study analyzing the projects of 12th grade high school informatics students in which they made models and ran simulations of phenomena from other disciplines. We constructed an analytic framework based on literature about modeling and analyzed students' project documentation, recordings of student groups at work and during presentations, survey results and interviews with individual students. We examined how to discern the elements of our framework in the students' work. Moreover, we determined which data sources are suitable for observing students' learning. Finally, we investigated what difficulties students encounter while working on modeling and simulation projects. Our findings result in an operational definition of modeling and simulation, and provide input for future development of both assessment instruments and instructional strategies.

Keywords: Computational thinking · Modeling and simulation · Informatics · Secondary education

1 Introduction

Following the increasing availability of computers in schools, several initiatives have been employed to aid students' learning in various disciplines through the use of computer models [2, 23, 24]. Nowack and Caspersen [6] argue why they “*believe understanding and creating models are fundamental skills for all pupils as it can be characterized as the skill that enable us to analyze and understand phenomena as well as design and construct artifacts.*” Wilensky argues, “*Computational modeling has the potential to give students means of expressing and testing explanations of phenomena both in the natural and social worlds*” [27]. Granger claims: “*Modeling is the new literacy*” [9]. This belief is also expressed in the fact that as of 2019, modeling and simulation (together called *Computational Science*), will be included in the new Dutch

high school informatics curriculum, described by the following high level learning objectives: “*Modeling: The candidate is able to model aspects of a different scientific discipline in computational terms*” and “*Simulation: The candidate is able to construct models and simulations, and use these for the research of phenomena in that other science field.*” Modeling itself will be a part of the compulsory core curriculum, described as “*Modeling: The candidate is able to use context to analyze a relevant problem, limit this to a manageable problem, translate this into a model, generate and interpret model results, and test and assess the model. The candidate is able to use consistent reasoning*” [1].

Modeling and simulation can be viewed as aspects of *Computational Thinking* (CT) [30] as they involve decomposition of open-ended problems and the construction and evaluation of models that simulate the nature of these problems in order to be able to provide solutions to those problems. The present study is part of a larger research project on CT in Dutch secondary education, investigating pedagogical aspects of CT and teachers’ pedagogical content knowledge (PCK, [22]) about these aspects. Following Magnusson et al. [17], we distinguish four aspects of content-specific pedagogy: (1) goals and objectives, (2) students’ understanding and difficulties, (3) instructional strategies, and (4) assessment. In the first phase of the project, we refined the CSTA definition of CT [10], explored teachers’ PCK [11, 12]; and made an initial exploration of the computational modeling process [13].

Aim of the Study. In this study we focus on CT skills related to modeling and simulation and we explore highly cognitively complex set of students’ activities related to modeling, in particular as an aspect of CT rather than as an aspect of e.g. mathematics [16]. We address the following research questions:

1. How can the intended learning outcomes of *Computational science* (modeling and simulation) be described in operational terms?
2. What data sources are suitable to monitor students’ learning outcomes when engaging in modeling activities?
3. What specific challenges do the students experience when engaging in modeling activities?

The first question addresses Magnusson’s aspect (1). The second contributes to aspect (4) – we plan to use our findings as input for a later study into a CT assessment instrument. The third question addresses aspect (2); our findings will help design teaching materials for modeling and simulation and thus indirectly contribute to (3).

Related Work. Previous work on characterizing modeling is done mainly in the areas of mathematics and natural sciences; see the following section. Research on making students’ learning process and outcomes visible has focused mostly on CT aspects such as algorithmic thinking or programming. The employed assessment instruments range from tests with closed questions [8], tests with open questions [18, 26], surveys [26], recordings or observations of students at work [18], examination of programming projects [4, 18, 26] to interviews with students [4, 14] and teachers [18]. In particular, Brennan and Resnick [4] “*are interested in the ways that design-based learning activities [...] support the development of computational thinking in young people*” and explore three approaches to assessment of the development of CT of the children

engaged in such activities. They discuss strengths and limitations of each of these approaches extensively and subsequently advocate a comprehensive approach to assessment that utilizes several data sources.

Context of the Study. Our exploratory case study was carried out during a project-based lesson series within a regular informatics course in the 12th grade of high school where students studied modeling and simulations. They used NetLogo to program models of phenomena from other disciplines and to explore them through running simulations. During a six-weeks period they studied Modeling and Simulations with NetLogo. The first three weeks were dedicated to studying the textbook material. During the rest of the period, the fourteen students comprising this class worked in seven groups on a practical assignment where they investigated a phenomenon of their choosing by making a model in NetLogo and exploring it through running simulations. When necessary, students were assisted in formulating their hypotheses or research questions. The entire process was strictly planned and contained milestones when the students turned in the required project documentation. At the end of the period, each group presented its model to the rest of the class and the students were encouraged to discuss their models, results, design choices, programming issues and other relevant questions. After the presentations, they turned in the final part of the project documentation where they described the feedback they got and their reaction to it. A few days later, twelve students (six groups) who finished their projects, turned in their final reports and NetLogo programs.

2 Modeling and Simulation

There is extensive literature on modeling in science and especially in mathematics. We take the latter as starting point and discuss *simulation modeling* as a special case of modeling.

2.1 Modeling

Van Overveld et al. [25] distinguishes two purposes of modeling: scientific research and technological design, and lists a number of goals that can be obtained through modeling: explanation, prediction, compression, abstraction, unification, analysis, verification, communication, exploration, decision, optimization, specification, steering and control and finally: training. The mathematical modeling process can be viewed as a problem solving activity (cf. [20]). We adopt the operationalization by Van Overveld et al. [25]:

1. *Definition stage*: the problem is stated and **researched** in the context domain (this is also considered a core aspect of CT [30]). The **purpose** of the model is formulated and a study is planned.
2. *Conceptualization stage*: Data are collected and a conceptual model is constructed and validated. In the process of **abstraction** it is decided what details to highlight and what details to ignore.

3. *Formalization stage*: the conceptual model is transformed into a formal model.
4. *Execution stage*: the model is being used for its purpose: this means solving the mathematical problem.
5. *Conclusion stage*: the results of the execution stage are **analyzed** and translated back into the problem domain, involving the presentation and interpretation of the results.

In addition, Perrenet and Zwaneveld [19] explicitly distinguish between the non-mathematical world containing the definition stage, conceptualization stage and conclusion stage on the one hand, and the mathematical world containing the formalization and execution stages on the other hand. Following each of these stages, reflection needs to take place: to check if any revisions are necessary by repeating that stage, to validate and verify the model, to assess the plausibility of the result and answer the initial purpose, to communicate the results and to learn from what one has done. After the completion of the modeling process, a **reflection** takes place and the whole process is possibly repeated. Hence, modeling can be seen as a *cyclic* process [19, 25].

2.2 Simulation Modeling

Simulation modeling can be seen as a special case of modeling in which the model consists of a computer program and therefore is *executable*. In comparison to the mathematical modeling process, the simulation modeling process shows a computational – rather than mathematical – interpretation of the conceptualization, formalization and execution stages:

1. *Conceptualization stage*: Data are collected and a conceptual model is constructed and validated. In the process of abstraction it is decided what details to highlight and what details to ignore. Problem is **formulated** in a way that enables us to use a computer and other tools to help solve them [5].
2. *Formalization stage*: a computer program is constructed, i.e. **requirements** and **specifications** are stated and the system is **implemented** and tested [7]. This includes making pilot runs, **verifying** the program and checking **validity** of the simulation model. If necessary the program is adjusted [15]. Thus, the formalization stage is a cyclic process in itself.
3. *Execution stage*: the model is being used for its purpose: designing and running **experiments** [15].

Simulation modeling encompasses three methods: (1) *System dynamics*, associated with high level of abstraction where the individual objects are aggregated. The models can be described in terms of differential equations that are often non-trivial to solve. (2) *Discrete event modeling*, where the system modeled is considered to be a process, “i.e. a sequence of operations being performed across entities”. The level of abstraction is lower. (3) *Agent based modeling* (ABM), which is made possible with recent growth of availability of CPU power and memory, does not assume any particular abstraction level. Agents have their properties and behavior and one can start building a model by identifying agents and describing their behavior even without

knowing how a system behaves as a whole. ABM makes it possible to model systems that are difficult to capture with older modeling approaches [3]. In our view, the last two characteristics of the ABM make it a suitable modeling method for our students who often lack deep understanding of the phenomena they model and make models specifically to deepen their understanding. To conclude, we consider conceptual representation which could be realized through the employment of ABM methods and software, in which *“you give computational rules to individual agents and then observe, explore analyze the resultant aggregate patterns”* [27] suitable for use in secondary informatics class *“because the individual-level behavior of agents is relatively simple, [and] ABMs feature relatively simple computer programs that control the behaviors of their computational agents”* [28].

In simulation modeling, repeating the conceptualization stage or going back and forth between the conceptualization stage and formalization stage are considered to be an integral part of the modeling process [15]. In the specific case of ABM, the boundaries between all modeling stages are blurred and it is considered a good modeling practice to develop a model in minute increments, cycling continuously through all modeling stages [29].

3 Method

The data were collected by the first author during the project based lesson series. In view of existing studies involving algorithmic thinking and programming (see the introduction), we decided to use a combination of several data sources as a promising approach for our exploration of the students' activities and learning difficulties in their projects.

During their work in the class and the final presentations, screen and voice *recordings* were made of students' groups. (No recordings were made of students working elsewhere, such as at home). Except for a few corrupted recordings, they were all transcribed verbatim. The *project documentation* of each group was collected. After receiving their grades, twelve students filled in an online *survey* individually where they were asked about how they approached the work on this project, difficulties they encountered, what they have learned, what they liked or disliked, and what suggestions they had for the improvement of the assignment. Students were also invited to be *interviewed*. Five semi-structured interviews were conducted with individual students. The students were requested to describe their projects and they were asked if they could design a new NetLogo model on the fly (i.e., draw a sketch of the interface on paper and describe the model in terms of agents and interactions). Finally, they were asked what they learned during their work on the projects. The interviews were recorded and transcribed verbatim.

Using atlas.ti CAQDAS software we performed a qualitative analysis of the recordings, project documentation, survey results, and interviews, with coding categories based on the elements of our operational definition (i.e., boldface items in Sect. 2): purpose, research, abstraction, formulation, requirements, specification, implementation, verification, validation, experiment, analysis, and finally, reflection. After coding, we ascertained the visibility of the modeling elements in each of the data

sources and examined the students' activities more in-depth, looking specifically for indications of students' difficulties connected to each of the elements.

4 Results

There were seven project teams. Five teams consisted of two students; one of three students, and one student opted to work by himself. Six of seven projects were successful; Team 5 did not finish theirs and did not turn in all the required project documentation. We first present an overview of visible occurrences of the elements of our modeling operationalization, organized by data source and student (team): see Table 1. Some elements were combined – see the descriptions below.

We now summarize the findings of our more in-depth analysis, organized by the elements of our operational description. We state our findings in general terms and illustrate them with characteristic text segments taken from the data.

Purpose. In the project documentation all teams clearly stated the purpose of their models. However, in the recordings we saw students tinkering with NetLogo and looking at existing models before deciding what phenomenon they wanted to model and explore. In answering the survey question whether it was difficult to decide what phenomenon to model and explore, four students answered affirmative and told us they had difficulties figuring out what could or could not be modeled. For S4a, who explored the behavior of partygoers together with S4b, the most important lesson learned during his work on this project was that it was important to have a clear idea of the purpose of the model before engaging in the modeling process – a thought shared by three other students in the survey.

Research. In the recordings we saw three students from two groups searching the Internet to learn about the phenomena they modeled. Team 3 reported in the documentation of their project about the possibilities to control the spread of the Ebola virus: *“Virus: does not spread through the air but through contact with an Ebola patient (sex, blood), slaughtering and eating of a sick animal, non-sterile needles. [...] incubation about 21 days, 9 out of 10 people die”*, without reporting the source. In the survey, S3b mentioned consulting her sister who studied medicine. Team 6, exploring the effect of ambient warmth and the presence of food on life of bacteria, did not report any research in their project documentation. Others did not visibly engage in research but developed their models based on what they already knew about the phenomena they modeled (e.g. Team 1 who explored chemical reactions – in the survey S1a wrote they learned that in chemistry lessons) or their presumptions (e.g. Team 7, who explored whether mousetraps were more effective than cats in catching mice, or Team 5, who explored the influence of weather on ice cream sales).

Abstraction. All students engaged in abstracting: choosing a level of abstraction, based on the decision they made with respect to relevancy of particular features and deciding what to include into their models and what to leave out.

In the recording we observed several students struggling to determine such a level of abstraction. For example, Team 1 - who initially neglected teacher's instruction to

Table 1. Frequencies of simulation modeling elements per data source per team or student. For example, Team 3 consists of students S3a and S3b.

		Purpose	Research	Abstraction	Formulating	Requirements and specification	Implementation	Verification and Validation	Experiment	Analysis	Reflection
Project documentation	Team 1	✓	✓	✓	✓	✓		✓			✓
	Team 2	✓		✓	✓	✓		✓			✓
	Team 3	✓	✓	✓	✓	✓	✓	✓		✓	✓
	Team 4	✓		✓	✓	✓		✓	✓		✓
	Team 5	✓		✓							
	Team 6	✓		✓	✓	✓		✓	✓		✓
	Team 7	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Surveys	S1a		✓		✓		✓		✓		✓
	S1b								✓		✓
	S1c	✓	✓			✓	✓		✓		
	S2	✓	✓	✓	✓		✓	✓	✓		✓
	S3a	✓	✓	✓	✓		✓			✓	✓
	S3b	✓	✓	✓			✓	✓	✓		✓
	S4a	✓						✓			✓
	S4b	✓	✓	✓	✓		✓	✓	✓		✓
	S5a	✓	✓				✓		✓		✓
	S6a		✓								
	S7a		✓	✓		✓	✓		✓		✓
	S7b		✓	✓					✓		
Interviews	S1a	✓		✓	✓	✓	✓	✓	✓		✓
	S2	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	S3a	✓	✓	✓		✓		✓		✓	✓
	S3b	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	S7a	✓		✓	✓	✓	✓	✓	✓	✓	✓
Recordings	Team 1	✓	✓		✓	✓	✓		✓		
	Team 2	✓		✓	✓		✓				
	Team 3	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
	Team 4	✓	✓	✓		✓	✓	✓	✓		✓
	Team 5	✓		✓	✓	✓	✓	✓	✓	✓	✓
	Team 6	✓	✓	✓		✓		✓	✓		
	Team 7	✓	✓	✓	✓		✓				
	Presentations	✓		✓		✓	✓	✓	✓	✓	✓

study the textbook first - had difficulties understanding the idea behind ABM and got ‘stuck’ in the notion of an aggregate state, e.g. thinking about pH as a contributing factor in a chemical reaction rather than the result of it. During the interview, S7a told us that he wanted his mice to reproduce but did not include this feature because he did

not know how to implement males and females. It did not occur to him that gender of the mice was not relevant in his model. Finally, as required, all students who finished the project turned in wish lists with features or aspects that were not implemented yet but should be considered for the next version of the model, thus demonstrating they were able to decide what to include or leave out.

Formulation. The assignment required a description of the behavior of the model in a natural language, and all the students who finished their projects did that. However, several students needed help to formulate their problems appropriately: e.g. only after choosing the right level of abstraction did Team 1 manage to formulate their problem appropriately and in the recordings we heard S1a say, *“Two of these things have to collide with each other and then something needs to happen”*.

Requirements and Specification. In the recordings it turned out to be hard to observe a distinction between requirements and specifications – see the results on Testing for a comprehensive example.

A description of requirements and specifications was a part of project documentation and all the students who finished their projects provided it. Team 7 wrote, *“The mousetraps need to be placed at random locations since we don’t know what the perfect locations would be. If a mouse contacts a mousetrap, then the mouse needs to die/disappear.”* In their project documentation, Team 1 stated requirements: *“In our program, two particles react to form two other particles. The probability that two particles react can be specified, as well as the reaction speed of the particles.”* Then they wrote specifications extensively: *“If two initial turtles (red and yellow) meet, then the current catalyst value (the left slider) determines whether they react.”* Team 3 wrote, *“In our model there is only one [breed of] turtles and it stands for people. These turtles can have various properties, such as being ill or healthy. They can be influenced by external factors such as medicine and their life span.”*

Implementation. While all students managed to implement something, some of them experienced difficulties. In the recordings we heard S1a say, *“I know what I want to do, but I don’t know how to code it. I don’t think it’s all that difficult, but...”* During the interview, S7a told us he refrained from including mouse reproduction because he did not know how to design this feature and program it. When constructing their programs, only one team worked top-down: the others rather engaged in bottom-up incremental development constantly adding new features to their models.

Verification and Validation. The recordings revealed a complex picture in which the distinction between validation and verification was not always clear. Team 4’s approach is representative of students’ strategy: they constructed their model (program) by cycling among stating requirements and specifications, implementing and testing, in minute steps: *“We have to do that with time, man, that they can only drink one beer in ten seconds or so, otherwise they drink too much!”* When testing, it was not clear whether they were validating their model or verifying their code: often they would run their program, see remarkable behavior and subsequently change the code. S4b: *“All dead.”* S4a: *“It begins to deteriorate now [in the simulation, the beer is gone and people leave*

the party quickly]” S4b: “But how could they all get the same amount of beer?” S4a: “That’s because of that piece of code.” S4b: “Really? Can’t that be changed? How did they do it with the sheep? [Referring to an example from NetLogo’s models library]” Subsequently they would change their code and continue their work in a similar fashion. Team 6 worked similarly. It was not clear whether S6b was validating or experimenting: “It works now but it is not balanced, so to speak.” S6a: “Yeah.” S6b concluded: “Yeah. That remains to be done” and went on to change the code. Later on they tried again. S6b: “And if we make this one a bit lower, say seven or so, then they die, that is really abrupt, like, either they live or so, or all dead.”

In the project documentation, all of the students reported that their models behaved as expected (validation). Several students described validating their models and adjusting when necessary. To this end, Team 1 wrote: “to prevent particles from reacting with each other immediately following a reaction, we built in a reaction pause. [...] That way you prevent particles from being stuck in a constant forth-and-back reaction.”

Experiment. Team 7 was the only team who documented systematically performed experiments with their model: they reported the initial parameter values (e.g. ten cats and nine mice) and included the resulting data plots in their project documentation. In the recordings we saw other students engage in experimenting to various degrees, but most failed to mention this in the project documentation.

Analysis. Not all the students provided an analysis of the results of the experiments, but in the project documentation, they all reported answers to the purpose of their models. Team 7’s analysis revealed, “The mousetraps were not always effective. Some mousetraps go off but the mouse manages to escape.” Finally, they concluded that mousetraps were more effective than cats in catching mice. In the recordings we saw Team 3 analyze their data, without reporting it in the project documentation, and their conclusion was, “We expected that the new medicine would decrease the spreading of Ebola. It turned out that the medicine worked rather quickly, but that the rate of infectiousness was of influence as well.”

Reflection. As required, all the students reflected on their models in the project documentation. Team 7 wrote: “Not everything in our model corresponds with the reality. But it is nice to experiment with it. You can make your model as large and complex as you wish.” In the survey the students were asked what they learned. S3b wrote: “It [modeling] is a good means to predict/research hypotheses. A good aid for research. I take chemical reactions as an example. You can make it and thus see (visualize) what happens,” a thought reflected by S1a too. S4a learned that it was important to have a clear idea of the purpose of the model before engaging in the modeling process and that models and simulations never completely correspond to the reality. Contrary to S4a’s reply, during the interview S1a expressed his astonishment about how easy it was to make a model that “actually reasonably corresponded” to what was modeled. He even went on to show it to his chemistry teacher.

5 Conclusion and Discussion

As to the **first research question** – *How can the intended learning outcomes of Computational science (modeling and simulation) be described in operational terms* – we have obtained an operational description based on literature on modeling and simulation. The elements of the description turned out to be suitable to classify simulation modeling activities of the students in our study. However, some of these had to be grouped together since the separate elements could not be distinguished. The resulting operationalization contains the elements *purpose, research, abstraction, formulation, requirements/specification, implementation, verification/validation, experiment, analysis, reflection*. This ‘blurring’ of activities is also described by Wilensky and Rand [29].

In answering our **second research question** – *What data sources are suitable to monitor students’ learning outcomes when engaging in modeling activities* – we found that every source enabled us to observe some aspects of the modeling process. The interviews provided the opportunity to observe all the aspect of the modeling process, closely followed by recordings of students at work. In the project documentation, the description of the model and the reflection are well represented and experimenting and analysis not so: contrary to the presentations, where exactly the opposite happens. The surveys, in their present form, did not provide much insight into the modeling stages the students engage in.

We are planning to use our results to develop an assessment instrument. In order for such an instrument to be feasible for classroom usage, a combination of project documentation and class presentation are promising data sources that enabled us to capture all aspects of students’ work. Our findings suggest that the instructions for documentation and presentation could be sharpened to improve visibility of (systematic) experimentation and data analysis within the model.

Finally, in answering our **third research question** – *What specific challenges do the students experience when engaging in modeling activities* – we identified some difficulties. Many students could not decide what to model exactly, and found it hard to decide on the level of abstraction and formulate the problem suitably for modeling through ABM. While all students managed to program something, not all of them were able to program all they wanted because either they could not decide on the relevance of a feature, or they did not know enough NetLogo to code it. During testing it appeared to be difficult to attribute unexpected behavior to a fundamental modeling mistake, a programming error, or unexpected (i.e. *emergent*) behavior that was characteristic for the phenomenon under scrutiny. Students tend to rely on an incremental trial-and-error strategy while implementing their simulation model. Only a few conducted systematic and well documented experiments. Most of these experiments, together with the analysis of the results, were intermingled with the construction of the models.

This incremental development is consistent with description of the modeling practice, for example by Wilensky and Rand [29]. An ad hoc incremental development (trial-and-error strategy) is typical for novices [21].

General Remarks. Although this was a small study with a limited number of participants, we learned a lot about students’ understanding of modeling and simulation.

Also, our findings indirectly informed us about the quality of the instruction, which leaves room for improvement.

Several students told us that through work on this project, they learned about the phenomena they modeled, which is in line with earlier findings [2, 24]. We often heard them laugh during their work and we observed that many students enjoyed working on this project. We saw that these informatics students were able to utilize their informatics/CT knowledge and skills to advance their learning in other disciplines.

In conclusion, we believe that the results of this research will contribute to the development of (1) suitable learning activities both within the informatics courses as elsewhere and (2) knowledge about teaching, monitoring and assessment of the CT aspect modeling and simulation. They will contribute to the development of the informatics curriculum in secondary education in the Netherlands, informatics teacher training and informatics education in general.

Acknowledgments. This work is supported by The Netherlands Organisation for Scientific Research grant nr. 023.002.138.

References

1. Barendsen, E., Tolboom, J.: Advisory Report (Intended) Curriculum for Informatics for Upper Secondary Education. SLO, Enschede (2016)
2. Blikstein, P., Wilensky, U.: An atom is known by the company it keeps: content, representation and pedagogy within the epistemic revolution of the complexity sciences (2009)
3. Borshchev, A.: The Big Book of Simulation Modeling: Multimethod Modeling with AnyLogic 6. AnyLogic North America, Chicago (2013)
4. Brennan, K., Resnick, M.: New frameworks for studying and assessing the development of computational thinking (2012)
5. CSTA Computational Thinking Task Force. Operational Definition of Computational Thinking for K-12 Education. <http://csta.acm.org/Curriculum/sub/CurrFiles/CompThinkingFlyer.pdf>. Accessed 16 Oct 2013
6. Caspersen, M.E., Nowack, P.: Model-Based Thinking & Practice
7. Comer, D.E., Gries, D., Mulder, M.C., Allen Tucker, A., Turner, J., Young, P.R., Denning, P.J.: Computing as a discipline. *Commun. ACM* **32**(1), 9–23 (1989)
8. Gouws, L., Bradshaw, K., Wentworth, P.: First year student performance in a test for computational thinking. ACM, East London, South Africa (2013)
9. Granger, C.: Coding is not the new literacy. <http://www.chris-granger.com/2015/01/26/coding-is-not-the-new-literacy/>. Accessed 09 Oct 2015
10. Grgurina, N.: Computational thinking in Dutch secondary education (2013)
11. Grgurina, N., Barendsen, E., Zwaneveld, B., van Veen, K., Stoker, I.: Computational thinking skills in Dutch secondary education: exploring pedagogical content knowledge. *ACM* (2014)
12. Grgurina, N., Barendsen, E., Zwaneveld, B., van Veen, K., Stoker, I.: Computational thinking skills in Dutch secondary education: exploring teacher's perspective. *ACM* (2014)
13. Grgurina, N., Barendsen, E., van Veen, K., Suhre, C., Zwaneveld, B.: Exploring students' computational thinking skills in modeling and simulation projects: a pilot study. *ACM* (2015)

14. Grover, S.: Robotics and engineering for middle and high school students to develop computational thinking (2011)
15. Law, A.M.: Simulation Modeling and Analysis, 5th edn. McGraw-Hill, New York (2015)
16. Maaß, K.: What are modelling competencies? *ZDM Math. Educ.* **38**(2), 113–142 (2006)
17. Magnusson, S., Krajcik, J., Borko, H.: Nature, sources, and development of pedagogical content knowledge for science teaching. In: Gess-Newsome, J., Lederman, N.G. (eds.) *Examining Pedagogical Content Knowledge*, pp. 95–132. Springer, Netherlands (1999)
18. Meerbaum-Salant, O., Armoni, M., Ben-Ari, M.: Learning computer science concepts with scratch. *Comput. Sci. Educ.* **23**(3), 239–264 (2013)
19. Perrenet, J., Zwaneveld, B.: The many faces of the mathematical modeling cycle. *J. Math. Model. Appl.* **1**(6), 3–21 (2012)
20. Polya, G.: *How to Solve It: A New Aspect of Mathematical Method*. Princeton University Press, Princeton (2008)
21. Robins, A., Rountree, J., Rountree, N.: Learning and teaching programming: a review and discussion. *Comput. Sci. Educ.* **13**(2), 137–172 (2003)
22. Shulman, L.S.: Those who understand: knowledge growth in teaching. *Educ. Res.* **15**(2), 4–14 (1986)
23. Spodniakova Pfefferova, M.: Computer simulations and their influence on students' understanding of oscillatory motion. *Inform. Educ.* **14**(2), 279–289 (2015)
24. Taub, R., Armoni, M., Ben-Ari, M.M.: Abstraction as a bridging concept between computer science and physics. *ACM* (2014)
25. Van Overveld, K., Borghuis, T., van Berkum, E.: From problems to numbers and back. In: *Lecture Notes to 'A Discipline-Neutral Introduction to Mathematical Modelling'*. Eindhoven University of Technology, Eindhoven (2015)
26. Werner, L., Denner, J., Campe, S., Kawamoto, D.C.: The fairy performance assessment: measuring computational thinking in middle school. *ACM*, Raleigh, North Carolina, USA (2012)
27. Wilensky, U.: Computational thinking through modeling and simulation. White paper Presented at the Summit on Future Directions in Computer Education, Orlando, FL (2014). <http://www.stanford.edu/~coopers/2013Summit/WilenskyUriNorthwesternREV.pdf>
28. Wilensky, U., Brady, C.E., Horn, M.S.: Fostering computational literacy in science classrooms. *Commun. ACM* **57**(8), 24–28 (2014)
29. Wilensky, U., Rand, W.: *An Introduction to Agent-Based Modeling: Modeling Natural, Social, and Engineered Complex Systems with NetLogo*. MIT Press, Cambridge (2015)
30. Wing, J.M.: Computational thinking. *Commun. ACM* **49**(3), 33–35 (2006)