Petri Nets in Secondary CS Education

Michael Rücker
Humboldt-Universität zu Berlin
Department of Informatics
ruecker@informatik.hu-berlin.de

Niels Pinkwart
Humboldt-Universität zu Berlin
Department of Informatics
pinkwart@informatik.hu-berlin.de

ABSTRACT

Petri nets are a graphic and dynamic model type for interactive and distributed systems. They have been proposed for secondary CS education before but existing discussions are superficial at best. We present the results of a qualitative evaluation of two 90-minute lessons on Petri nets. The aim was to generate several working hypotheses for future research.

Categories and Subject Descriptors

K.3.2 [**Computers and Education**]: Computer and Information Science Education—*computer science education*

Keywords

Computer science education, modeling, Petri nets, qualitative evaluation

1. INTRODUCTION

Petri nets are a graphic and dynamic model type suited to represent interactive, embedded and concurrent systems. For a comprehensible introduction, see [4]. Petri nets have been proposed for secondary CS education before, e.g. in [6] and [3] (p. 248-254). However, these discussions hardly go beyond a short introduction, some example nets and the general claim that those examples are suitable for the CS classroom. They largely lack methodical considerations or empirical evidence to back up this claim. In fact, the only practical report on an actual school lesson on Petri nets that we were able to find comes from the final exam of a German trainee teacher [5]. Therefore, we believe that the methodical and practical aspects of using Petri nets in secondary CS education require further investigation.

In the following, we present a qualitative evaluation of two 90-minute lessons that were conducted at a German grammar school. The goal was to provide some practical insight into using Petri nets in secondary CS education and to generate several working hypotheses for future research.

WiPSCE Nov 05-07 2014, Berlin, Germany
Copyright 2014 ACM 978-1-4503-3250-7/14/11 ...$15.00
http://dx.doi.org/10.1145/2670757.2670766

2. METHODICAL CONSIDERATIONS

Using Petri nets in a constructivist way, i.e. having students create and analyze their own models (cf. the general literature on modeling in science education, e.g. [1, 2]), requires an editor tool that allows students to easily create, simulate and revise their models. In the absence of a tool that suited our needs, we developed *Versys*, a Petri net editor specifically designed for the use at schools. Written in Java, *Versys* is platform-independent and portable. It is published under the GNU General Public License v3 and is available at `https://cses.informatik.hu-berlin.de/software/details/versys/`. At any time, *Versys* allows to start either a manual or automatic simulation. This requires the model in the editor to be syntactically correct at all times. Therefore, *Versys* does not allow syntactically incorrect operations, like connecting two places via an arc.

3. LESSON DESCRIPTIONS

The first lesson was conducted in a year-9 CS compulsory elective course. It revolved around embedded systems. The course consisted of 13 students (around age 15), who did not have any prior knowledge on embedded systems or Petri nets. First, the concept of embedded systems was defined and Petri nets were introduced as a suitable means to model such systems. Then students were required to simply recreate a provided model in *Versys*. In the second half of the lesson, students were presented with a LEGO-Mindstorms model of a ventilation system and were required to model the system's behavior as a Petri net.

The second lesson was conducted in a year-12 CS A-level course. It revolved around the concepts of mutual exclusion and deadlock. The course consisted of six students (around age 17) who did not have any prior knowledge on theses concepts. They did, however, have some prior knowledge on Petri nets, which were introduced to them during a project week about half a year prior to this study. Thus, the beginning of the lesson merely included a short recap. Subsequently, students were introduced to the concept of mutual exclusion by means of a corresponding Petri net. In the second half of the lesson, the students were given a textual description of the system of dining philosophers and were required to model the system's behavior as a Petri net.

4. EVALUATION METHODS

For each of the two 90-minute lessons, three different data collection methods were employed. As a first measure, a participant observation was conducted. The duration of activities, various student and teacher utterances as well as specific student actions were documented in a lesson protocol. The protocol was also supplemented by the students' models produced during the lesson. Second, the teacher was asked to give an assessment of the lesson. For this, an interview was conducted afterwards. The teacher can be seen as a second participant observer – yet, being the regular CS teacher of the course, offered a different perspective and was able to provide some initial insight into the underlying learning processes. The interview was recorded and then transcribed. Finally, as a third measure, a student test was conducted at the beginning of the next CS lesson. It was designed to test the students' knowledge on the general syntax and terminology of Petri nets, their understanding of the firing rule, and their ability to create and/or analyze Petri net models.

The obtained lesson protocols, interview transcripts and test results were then coded and triangulated. While space does not permit for a detailed description of the analysis methods here, the entire (German) material, including detailed lesson plans, interview guidelines, test tasks and grading guidelines, and used codes, are available at request.

5. FINDINGS

In both lessons, only a small amount of time was dedicated explicitly to the syntax of Petri nets before students started using the provided editor tool to create and simulate their own models. Since the tool did not allow syntactic mistakes, it is not surprising that many year-9 students demonstrated only a very limited knowledge of Petri net syntax during the test. However, this need not be interpreted as negative.

Hypothesis 1. Given a suitable editor tool, Petri nets can be used in a modeling task without an extensive formal introduction of how they work.

Syntactic correctness does, of course, not imply semantic correctness. During class as well as in the test, students apparently used two different approaches to determine whether their models are semantically correct. The older students in year 12 focused primarily on the static net structure. In the test, some even forgot to place any tokens on their created models at all. In contrast, the year-9 students focused much more on the visible movement of tokens. While running a simulation, one student noted that tokens got lost. His model was missing an arc. However, instead of changing the net's structure, he tried to solve the issue by experimenting with different markings. In the test, almost all year-9 students demonstrated a good understanding of the firing rule.

Hypothesis 2. In order to evaluate the correctness of a Petri net model, younger students focus more on the visual movement of tokens, older students more on the static net structure.

In one of their test tasks, the year-9 students were expected to analyze an embedded system with the help of a model. In essence, they were expected to explain why a certain transition cannot be enabled under certain conditions. Regardless of their correctness, the students' answers can be divided into two groups: those that used the model as a basis for their argument, and those that did not. While some students argued using their everyday knowledge, most tried to answer the question by pointing out places and transitions. This suggests that those students were able to distinguish between the model and the real system. Furthermore, in the year-12 course, the students' models of the dining philosophers used two very different, but valid modeling approaches. This led to a discussion among some of the students as to which was the better way to model the system. This clearly shows that they were able to distinguish between the model of the dining philosophers, of which there were two versions, and the modeled system itself.

Hypothesis 3. Petri nets are intuitive enough to serve as comprehensible illustrations, yet abstract and versatile enough to foster a distinction between model and reality.

6. DISCUSSION AND FUTURE WORK

The study presented here can be but a first step in a more systematic investigation of using Petri nets in secondary CS education. The above findings do not yet claim to be objectively valid or directly generalizable to other learning arrangements. The aim was to generate several working hypotheses as a starting point for future research.

However, if these hypotheses were to be validated, each of them would make for a strong case of using Petri nets in CS education. Hypothesis 1 indicates that Petri nets may support a task-based learning style. Models are not created for their own sake, but as tools to solve specific problems. Petri nets may allow for such a use without extensive introductory overhead. Hypothesis 2 highlights the potential benefits of dynamic models. In lower grades especially, dynamic models like Petri nets, which allow for visual simulations, may reduce cognitive load as opposed to static models like structograms or flow charts. Hypothesis 3 addresses a very fundamental issue, since fostering a distinction between model and reality is a general challenge when teaching models in the science classroom.

Given these considerations, we are convinced that the further investigation of Petri nets in secondary CS education is a worthwhile and potentially very fruitful effort.

7. REFERENCES

[1] R. K. Coll, B. France, and I. Taylor. The role of models/and analogies in science education: implications from research. *International Journal of Science Education*, 27(2):183–198, 2005.

[2] J. D. Golbert and B. C. Buckley. Introduction to model-based teaching and learning in science education. *International Journal of Science Education*, 22(9):891–894, 2000.

[3] P. Hubwieser. *Didaktik der Informatik: Grundlagen, Konzepte, Beispiele*. Springer, Berlin, 3 edition, 2007.

[4] W. Reisig. *Understanding Petri Nets: Modeling Techniques, Analysis Methods, Case Studies*. Springer, Berlin, 2013.

[5] T. Rotering. Petrinetze als Modellierungswerkzeug auf dem Weg zur Implementation von Steuerungsprozessen in der Sekundarstufe II. Staatsexamensarbeit, 2010. http://rotering-net.de/edu/files/staatsarbeit2.pdf, last retrieved: 11.07.2014.

[6] A. Schwill. Beschreibung paralleler Abläufe mit Petri-Netzen. *Log In 13*, 5:45–51, 1993.