Proceedings of the 52nd Hawaii International Conference on System Sciences | 2019

Teaching Conceptual Modeling 2019 (TECOMO'19) Welcome from the Minitrack Co-Chairs

Bastian Tenbergen
Department of Computer Science
State University of New York at Oswego
bastian.tenbergen@oswego.edu

Marian Daun, Jennifer Brings
paluno – The Ruhr Institute for Software Technology
University of Duisburg-Essen, Essen, Germany
[marian.daun | jennifer.brings] @paluno.uni-due.de

1. Description of a Minitrack

Over the past several years, software systems have increased in complexity, size, and criticality. Conceptual models, i.e. graphical representations, have proven to be a useful tool for the industry to cope with the corresponding challenges. It has widely been shown that using graphical representations aids in communication, simplifies prioritization of artifacts, enables code-generation, fosters quality assurance, and assist knowledge discovery. Hence, conceptual models are widely used, for instance, as sources for discussions with stakeholders in requirements engineering or architecture design. Furthermore, models become tightly integrated into the development process, particularly in model-driven development or in model-based engineering.

Therefore, computer science curricula, industry consultants, and educators at large have begun focusing on teaching the application of conceptual models during software development. However, there remain open and recurring questions regarding what differentiates a "good" conceptual model from an inadequate one, how to use conceptual models of different types in conjunction with one another in a meaningful way, or simply how to avoid ambiguity and vagueness. This Minitrack focusses on how to teach the use of conceptual models to students, as students often struggle with selecting the right level of abstraction, strive for aesthetics and must understand the rules of the used graphical language, while trying to separate "inventing" the system from "describing" the system.

This Minitrack explores challenges, experiences, approaches, ideas, and new impulses in teaching conceptual modeling. In a highly interactive atmosphere, where the challenges of teaching conceptual modeling can be discussed, positioned, and addressed, we sought thought-provoking and highly constructive discussions among a broad audience and presenters in order to jointly identify promising educational approaches. We want to try out proposed approaches, foster empirical studies, and facilitate collaboration between industry and academia in teaching conceptual modeling.

This Minitrack accepted contributions focused on, but not limited to the following topics:

- Teaching approaches for conceptual modeling
- Experience reports, especially challenges, difficulties, pitfalls, and negative experiences with learning success, project/assignment outcome, or the application of teaching approaches
- Assignment/Project ideas, experiences, and instructional support for student work
- Methods of instruction, e.g., flipped classroom, problem-based learning
- Case studies and case examples from industry and academia
- Proposals for and/or results of empirical studies on conceptual modeling
- Methods and strategies of feedback and grading of student work
- Conceptual modeling curricula and course structures
- Teaching semantics, content, correctness, adequacy, aesthetics, and consistency of models, levels of abstraction, model integration, and code-generation
- Teaching model quality, metamodeling, and a structured modeling process
- Teaching modeling frameworks, languages, and diagram types

2. Program Committee and Review Process

Each paper submitted to the Teaching Conceptual Modeling Minitrack underwent a thorough review, by at least four experts in the field. To ensure comparable high-quality reviews each paper was reviewed by experts for conceptual modeling as well as experts in the field of software engineering education. Furthermore, each paper was reviewed in a double-blind fashion, strictly controlling for conflicts of interest (see Principle 4 in http://www.acm.org/about/se-code#full). The following individuals served as the program committee:

- Mikio Aoyama, Nanzan Univ. (Japan)
- Steven Clyde, Utah State Univ. (USA)
- Sergio A. A. Freitas, Univ. de Brasília (Brasil)
- Regina Hebig, Chalmers Univ. (Sweden)
- Rogardt Heldal, Chalmers Univ. (Sweden)
- Elke Hochmüller, FH Kärnten (Austria)
- David Kung, Univ. of Taxas at Arlington (USA)
- Steve Tockey, Construx Software (USA)

3. Inaugural Program

In its year, TeCoMo was hosted as a mini track at the 52nd Hawaii International Conference on System Sciences 2019 (HICSS-52) as part of the Invited Track "Software Engineering Education and Training." This track has a long, successful history as a standalone conference known as CSEE&T and took place for the first time as part of the HICSS conference. From more than 50 contributions submitted to all Software Engineering Education minitracks, two submissions were accepted to TeCoMo.

In [1], Dominick Bork reports on a meta model for conceptual modeling that meets the educational objective metatypes outlined in Bloom's Taxonomy. Bork suggests that in order to be successful at conceptual modeling, one must master Bloom's principle cognitive educational process: remember, understand, apply, analyze, evaluate, and create. Using the taxonomy, the author describes what these processes mean with regard to conceptual modeling and metamodeling, respectively. The resulting framework was applied to a case example. The results from the case example application can be used to improve conceptual modeling education in a meaningful way.

In [2], Justin MacCreery and Bastian Tenbergen contribute to the body of knowledge on students' conceptual model quality. Based on the Krogstie and Lindland's framework for conceptual model quality (i.e. where the quality of a model depends on semantic, syntactic, and pragmatic aspects), the authors report quantifiable measurements on the usage and error frequency of language features in UML sequence, class, activity, and state machine diagrams in four software engineering courses at the baccalaureate level. In addition, the authors quantified the semantic quality based on criteria derived from UML's notational rules and inter-diagram correspondence rules and describe the inter-model usefulness with regard to the modeling task experienced by the respective modeling team. As some of the most interesting findings, the authors report that student modelers only use a small feature set of the UML language with very high frequency of semantic errors (i.e., missing association labels in UML class diagrams). Nevertheless, the authors found that the benefit of conceptual models for the respective development team is high in the sense that despite the limited language features and high number of semantic inaccuracies, the act of producing conceptual models significantly aided the engineering process.

We hope that the first edition of TeCoMo lays the foundation for more research in this particular area to come forward. As TeCoMo is intended as a platform for knowledge and experience exchange, we invite the community to continue where TeCoMo 2019 left off.

References

[1] Bork, D., "A Framework for Teaching Conceptual Modeling and Metamodeling Based on Bloom's Revised Taxonomy of Educational Objectives."

[2] MacCreery, J., Tenbergen, B., "On the Syntactic, Semantic, and Pragmatic Quality of Students' Conceptual Models."