Visualization, Simulation and Analysis of Reconfigurable Systems

Claudia Ermel\textsuperscript{1} and Karsten Ehrig\textsuperscript{2}

\textsuperscript{1} Institut für Softwaretechnik und Theoretische Informatik
Technische Universität Berlin, Germany
claudia.ermel@tu-berlin.de
\textsuperscript{2} Department of Computer Science
University of Leicester, UK
karsten@mcs.le.ac.uk

Abstract. Meta-modeling is well known to define the basic concepts of domain-specific languages in an object-oriented way. Based on graph transformation, an abstract meta-model may be enhanced with information on concrete visualization of objects and relations, and the language syntax is defined by a graph grammar. Moreover, graph transformation can also formalize the semantic aspects of models, thus providing a basis for model validation by simulation.

Apart from editing and simulating the behavior of a system, there may be necessary reconfiguration operations which change the underlying system structure at runtime. In this paper, we focus on the interrelation of simulation and reconfiguration operations using formal verification techniques based on graph transformation. Our approach is demonstrated by the definition of a domain-specific language for building, simulating and reconfiguring small railway systems, using the Tiger tool environment. For further verification, we define a model transformation from the railway domain to Petri nets.

Keywords: Graph transformation, model transformation, reconfigurable system, visualization, simulation, analysis.

1 Introduction

Domain-specific modeling (DSM) aims to model a system at the same level of abstraction with the domain itself. This reduces mental mapping by moving the modeling language closer to the domain as perceived by designers, and improves the model quality compared to using generic modeling languages. The disadvantage of DSM is that for each domain a different visual modeling tool is needed. Here, meta CASE tools can help (like e.g. MetaEdit+ \cite{1}), which generate e.g. a visual editor on the basis of a definition of the visual domain-specific language.

\textsuperscript{*} This work has been partially sponsored by the IST-2005-16004 Integrated Project SENSORIA (Software Engineering for Service-Oriented Overlay Computers).
Two main approaches to visual language definition can be distinguished: grammar-based approaches or meta-modeling. Using graph grammars \[4\], multidimensional representations are described by graphs. This allows not only a visual notation of the concrete syntax, but also a visualization of the abstract syntax. While the concrete syntax contains the concrete layout of a visual notation, the abstract syntax abstracts from the layout and provides a condensed representation to be used for further processing, e.g. behavior simulation or system reconfiguration. Graph rules are used to manipulate the graph representation of a language element. Meta-modeling (see e.g. \[2\]) is also graph-based, but uses constraints instead of a grammar to define a visual language. While visual language definition by graph grammars can borrow a number of concepts from classical textual language definition, this is not true for meta-modeling.

Graph transformation can also formalize the semantic aspects of models. There are numerous formal graph-transformation-based semantics definitions \[3\]. In this paper, we use graph transformation not only to construct and visualize domain-specific visual models, but also to simulate dynamic model behavior. Apart from operations for editing, there may be necessary operations to change the underlying system structure at runtime (i.e. during simulation). Systems allowing to be changed have become an important topic in recent years since the adaption of a system to a changing environment plays a significant role e.g. in computer supported cooperative work, multi agent systems or mobile networks. In our approach, such reconfiguration operations are modeled by reconfiguration rules, and the corresponding systems are called reconfigurable systems.

As running example, we model a toy railway system. The visualization shows different kinds of tracks and switches which can be glued at connection points. Simulation rules allow to move a train to an adjacent track, respecting the switch modes. Reconfiguration rules allow to toggle between two modes of a switch. Graph transformation as a formally defined calculus \[1\] offers well-founded theoretical results that support the formal reasoning about graph-based models at all levels. We apply formal graph transformation techniques to reason about the independence of simulation and reconfiguration steps. For further verification, we define a model transformation from the railway system language to Petri nets. We apply the TIGER environment \[5\] for generating visual editor plug-ins in ECLIPSE \[6\] from graph grammars. TIGER is based on the graph transformation engine and analysis tool AGG \[7\].

The paper is structured as follows: Section \[2\] reviews the concepts for the graph-grammar based definition of visual languages, demonstrated by a domain-specific language to model small railway systems. In Section \[3\], concepts for simulation and reconfiguration of discrete-event systems by graph transformation are discussed, and the railway system is coming to life by operations for moving trains and changing switch modes. Section \[4\] applies verification techniques to analyze the interrelation of reconfiguration and simulation steps. Furthermore, a model transformation to Petri nets is defined, which allows to verify further dynamic system properties.