WS GTaD-2003 - The 1st Workshop on Graph Transformations and Design
ed Grabska, E., Seite 6-7, Jagiellonian Universität Krakau

Support of Conceptual Design in Civil Engineering by Graph-based Tools

Bodo Kraft, Manfred Nagl
Aachen University of Technology (RWTH), Software Engineering
{kraft|nagl}@i3.informatik.rwth-aachen.de

Abstract

1. Graph-based Tools

At Computer Science III Department of RWTH various tools for supporting development processes have been build in the past, for software engineering, mechanical engineering, chemical engineering, process control, telecommunication systems, and authoring support. This paper talks about a rather new application domain, namely *civil engineering.*

The tools we have been building are intelligent, incremental and integrated, where integration falls into two categories. In *a priori integration*, new tools are developed for the purpose of tight integration. In the *a posteriori* case given tools are integrated by providing new tools functionality. In this paper both integration approaches are handled.

In all tools mentioned above we use a *graph-based tool construction procedure*: internal data structures of tools are modelled as graphs, changes due to command invocations are specified by graph rewriting systems. Then, there are two different ways for constructing tools.

In the *research-oriented* branch, we derive tools automatically from specs, using the PROGRES system for specification development, a code generator for producing code out of the spec, and the UPGRADE visual framework environment into which the code is embedded. The resulting tools are efficient demonstrators for proof of concept purposes. Theses tools are, however, based on our academic development infrastructure. In the *industrial* branch we use the specification as a guideline for implementation, there by making use of available platforms and utilities. Usually, we use the research-oriented branch for a priori integration and the industrial one for the a posteriori case. This is also done in this paper.

2. Approach for Conceptual Design in Civil Engineering

Conceptual Design means that design results are elaborated on a coarse and abstract level without regarding details which are later included in constructive design (in other disciplines called detail engineering). On the other hand, the main goal of conceptual design is to take the various *levels of semantics* for a design problem into consideration: (a) domain specific knowledge, as standards, economy rules, security constrains, or common and accepted design rules, (b) experience knowledge in form of best practice or of using previous design results and, finally, (c) specific user behaviour knowledge or wishes, where users are customers or architects, respectively.

The *essentials* of our *conceptual design approach* are that (i) explicit knowledge can be used, formulated, or enhanced, (ii) change support is specifically supported,

where changes can happen on the level of knowledge as well as for design results, (iii) a lot of consistency checks are included in order to report errors as soon as possible, and (iv) that a smooth connection to constructive design is aimed at. The approach specifically pays off, if (v) different classes of buildings are regarded and, within a class, different designs of buildings and different variants thereof.

Our approach consists of two parts: (A) we realize a graph-based demonstrator by which an experienced architect (knowledge engineer) can specify knowledge of the three areas mentioned above by tools. Any of these areas is studies by one ore two examples. For the usual architect, there are further tools for developing constructive designs. These designs are immediately checked against the underlying knowledge. For the realisation of these tools, which we call *conceptual experimentation platform*, we use the enhanced machinery already sketched above which has been developed in the group within the last 15 years.

The experience platform cannot be used in an industrial context, as it is heavily based on the realisation machinery developed in an academic context. Therefore, a second demonstrator, called *conceptual design tool extension*, uses an industrial design tool (ArchiCAD by GRAPHISOFT) and extends it by semantical concepts as rooms, areas of housing, semantical relations, and alike. Again, it is possible to specify conceptual knowledge and to check a design result against this knowledge. This demonstrator is also graph-based, where graph-based is related to the concepts the implementor has in mind during tool extension.

For the *coupling* of both *demonstrators* we extract conceptual knowledge from the conceptual experimentation platform after it has been evaluated and its usefulness has been approved. This knowledge is then stored in the tool extension in an appropriate database form to be used for consistency checks of design results against the database.

3. Outline of the Paper

We discuss both approaches and demonstrators in more detail. For the experimentation platform we explain the underlying semantical concepts, how knowledge is represented, how conceptual design results look like, and how the consistency checks are handled. The *functionality* of knowledge tools, conceptual design tools, and checkers are *specified* by *graph rewriting*. Especially, we sketch how we gained flexibility for the specifications in the sense that the underlying knowledge can easily be changed or extended by the knowledge engineer. This implies a method for graph rewriting specifications different from that in the past, where the acquisition of knowledge is only to be found in the tool construction process.

In the second part, describing the industrial demonstrator, we concentrate on how semantial concepts and underlying knowledge can be incorporated within an industrial tool. Thereby, we try to make the solution as independent as possible of the underlying tool to be extended. Finally, we explain how conceptual and constructive design can be integrated by an incremental floor generator, which generates an initial form of a constructive design from the conceptual design, to be further elaborated.

Finally, we give a rough sketch of the outside behaviour of both demonstrators.