Barcelona, Spain, October 7-8, 2002

Graph-Based Tools (GraBaTs 2002)

Tom Mens, Andy Schürr and Gabriele Taentzer

Abstract

Graphs are well-known, well-understood, and frequently used means to depict networks of related items. They are successfully used as the underlying mathematical concept in various application domains. In all these domains tools are developed that store, retrieve, manipulate and display graphs as underlying data structures, despite of the fact that in most cases these graphs have a different name such as object diagrams, (meta) class diagrams, hyper documents, semantic webs etc. It is the purpose of this workshop to summarize the state of the art of graph-based tool development, bring together developers of graph-based tools in different application fields and to encourage new tool development cooperations. Motivation Graphs are an obvious means to describe structural aspects in various fields of computer science. They have been successfully used in application areas such as compiler compiler toolkits, constraint solving problems, generation of CASE tools, pattern recognition techniques, program analysis, software engineering, software evolution, software visualization and animation, and visual languages. In all these areas tools have been developed that use graphs as an important underlying data structure.

Since graphs are a very general structure mechanism, it is a challenge to handle graphs in an effective way. Using graphs inside tools the following topics play an important role: efficient graph algorithms, empirical and experimental results on the scalability of graphs, reusable graph-manipulating software components, software architectures and frameworks for graph-based tools, standard data exchange formats for graphs, more general graph-based tool integration techniques, and meta CASE tools or generators for graph-based tools. The aim of the workshop on graph-based tools (GraBaTs) is to bring together developers of all kinds of graph-based tools in order to exchange their experiences, problems, and solutions concerning the efficient handling of graphs.

The GraBaTs workshop was held for 1 1 2 days. Its schedule contained in addition to the afore-mentioned session on graph transformation tools, an invited talk by Tiziana Margaria (University of Dortmund, Germany) on ETI, an 1571-0661/$ – see front matter © Elsevier B.V.
10.1016/S1571-0661(05)80523-7
Workshop Issues

The workshop aims at bringing together tool developers from different fields, dealing with graphs from different perspectives. In the following, we give an overview on the most important perspectives.

Meta-modeling by Graphs

For a long time, the syntax and static semantics of most visual modeling or programming languages was only defined by means of characteristic examples and informal descriptions. To improve this situation, the visual language community invented grammar-based formalisms for the definition of the syntax of their languages, such as constraint grammars, graph grammars, relational grammars, etc. Unfortunately, it turned out that the grammar-based definition of visual languages is rather complicated compared with the meta-modeling approach developed in parallel. The Meta-modeling approach for the definition of visual languages uses a combination of class diagrams (ER-diagrams, etc.) and predicate logic expressions (Z, OCL, etc.) to define the syntax and static semantics of visual languages. It became popular with the standardization of the OO-modeling language UML and is used by various meta-modeling (meta-CASE) tools which are able to generate domain-specific CASE tools. The so-called MOF approach (Meta-Object Facility) is one attempt to come up with a meta-modeling standard. Despite of its limited expressiveness (compared with ER diagrams or UML class diagrams) MOF builds the basis for the formal definition of UML and other visual languages. All meta-modeling approaches used nowadays have one common property: they offer graph-like diagrams for the definition of the structure (syntax) of graph-like diagram languages. Therefore, meta-modeling is in fact the formal definition of graph languages by means of graphs which are instances of “meta” graph languages. As a consequence, meta-CASE tools are a special class of graph-based tools, which need at least basic services for storing, visualizing, and analyzing graphs. Graph Visualization Facilities for visualizing graph diagrams of graph-based by all kinds of graph-based tools ask whether they are e.g. used for meta-modeling or rule-based programming purposes. Furthermore, graph visualization techniques are the most important means for visualizing various aspects of software architectures, the dynamic behavior of running systems, their evolution history, and so forth. Software components developed for these purposes usually have to deal with huge graphs and need services for making these graphs persistent, for introducing abstractions based on hierarchical graph models, for computing reasonable graph layouts (efficiently), and for displaying graph visualization techniques are often employed for teaching purposes in computer science courses on “data structures and (graph) algorithms”. To summarize, almost all kinds of graph-based tools urgently need efficiently and effectively working graph visualization services, whereas graph visualization tools may profit from research activities on graph query and graph transformation engines for the computation of graph abstractions or views. We, therefore, hope that this workshop encourages researchers to start new cooperations, such as adapting graph visualization tools to the needs of graph manipulation tools or exploiting graph manipulation and transformation techniques to compute sensible abstractions of huge graphs. Graph Queries and Graph Algorithms Most, if not all, graph-based tools use to a certain degree software components (libraries, subsystems, etc.) for executing graph queries and/or various kinds of standard graph algorithms. For example, graph transformation tools rely on rather sophisticated means for computing graph matches (rule occurrences) and graph-based reverse engineering tools need rather powerful query engines for determining critical substructures of software architectures. On the other hand, quite a number of database management systems have already been developed using graphs (networks of related objects) as the underlying data model and offering query languages based on graph path expressions or even graph transformations. Vice versa, graph transformation languages like PROGRES are not only used for specifying and visualizing graph algorithms, but incorporate many elements of database query languages such as means for the construction of indexes, the materialization and incremental update of views, etc. Therefore, we like to encourage tool developers again to start cooperating across the boundaries of research areas.

Graph Transformation

Graph transformation means the rule-based manipulation of graphs. Several graph transformation approaches have emerged which differ w.r.t. to the underlying kind of graphs as well as in the way how rules are applied to graphs, i.e. graph transformation takes place. The kind of graphs used by these tools include labeled, directed graphs, hypergraphs, and graph structures. Their rules, the basic means to manipulate graphs, differ w.r.t. to the formal definition of their semantics, the way how occurrences (matches) are searched for, and how matching rules are applied eventually.

In tools, graph transformation is applied to visual languages, specification, code generation, verification, restructuring, evolution and programming of software systems, etc. Developers of graph transformation tools may profit from other workshop participants concerning more efficient realizations of basic functionality, while developers of other graph-based tools might find the graph transformation paradigm attractive to implement certain graph manipulations. The workshop may also provide insights to apply these tools to other application domains.

Common Exchange Formats for Graphs and Graph Transformation

To support interoperability between various graph-based tools, several initiatives on the development of common exchange formats for graphs
have been founded. These formats are all based on the extensible markup language XML developed
to interchange documents of arbitrary types. Preceding events like three subgroup meetings of
the EU Working Group APPLIGRAPH, a Workshop on Standard Exchange Formats, and a
satellite workshop of the 8th Intl. Symposium on Graph Drawing (GD 2000) discussed various
ideas which are currently converging to one format being GXL. During the GraBaTs workshop
a further discussion round on this topic was organized focusing especially on graph layout and
graph attributes. Another topic of interest for this discussion is an exchange format for graph
transformation systems called GTXL, which is under development and which will be built on top
of GXL. **Workshop Organizers** The Program Committee of the workshop consists of:

Luciano Baresi (Italy)
Giuseppe Di Battista (Italy)
Ulrik Brandes (Germany)
Scott Marshall (The Netherlands)
Tom Mens (Belgium) (Co-chair)
Andy Schürr (Germany) (Co-chair)
Gabriele Taentzer (Germany) (Co-chair)
Andreas Winter (Germany)
Albert Zündorf (Germany)

We are very grateful to Hartmut Ehrig for his help with the organization of the Workshop as
satellite event of the 1st Int. Conference on Graph Transformation (ICGT) and to Mike Mislove,
one of the Managing Editors of the ENTCS series. Thanks are also due to Fernando Orejas and
his local organizers at UPC in Barcelona who supplied preprints of this volume for all workshop
participants.