Gravisto: Graph Visualization Toolkit

Christian Bachmaier[1], Franz J. Brandenburg[1],
Michael Forster[1], Paul Holleis[2], and Marcus Raitner[1]

[1] University of Passau, 94030 Passau, Germany
`{bachmaier,brandenb,forster,raitner}@fmi.uni-passau.de`
[2] Ludwig Maximilian University Munich, 80333 Munich, Germany
`paul.holleis@ifi.lmu.de`

Abstract. *Gravisto*, the Graph Visualization Toolkit, is more than a (Java-based) editor for graphs. It includes data structures, graph algorithms, several layout algorithms, and a graph viewer component. As a general toolkit for the visualization and automatic layout of graphs it is extensible with plug-ins and is suited for the integration in other Java-based applications.

Overview: Gravisto is a new approach towards an extensible graph visualization toolkit. Entirely written in Java, *Gravisto* runs on all Java2 1.4 platforms, including Linux, Solaris, MacOS X, and Microsoft Windows. *Gravisto* can be obtained under the terms of the GNU General Public License (GPL) from [2].

Architecture: Gravisto consists of three layers; see Fig.1. The basic layer contains the graph data structures. The editor layer uses the basic data structures and provides managers for easy extension and customization. The top layer comprises all plug-ins, either delivered with *Gravisto* or from third-parties.

Fig. 1. System architecture

As several components must be notified about changes of the data structure, e. g., a view component in a plug-in, *Gravisto* employs the Observer Design Pattern: the *Event Manager* allows a component to register as a special type of event handler, depending on the events it likes to receive.
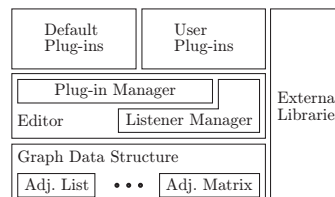
Interfaces: Gravisto provides a powerful plug-in mechanism with a comfortable plug-in manager. Most non-core functionality is realized as plug-in, including algorithms, node and edge attributes, graphical user interface components, input and output serializers, attribute inspectors, node and edge shapes, tools, or complete views. The idea behind this paradigm is to facilitate extensions of *Gravisto*; thus encouraging people to contribute to the project. Furthermore, this concept allows easy customization of the editor for different application scenarios, e. g., a biologist, working with biochemical pathways, does not need most functionality of the standard graph editor, but, for instance, the nodes and edges
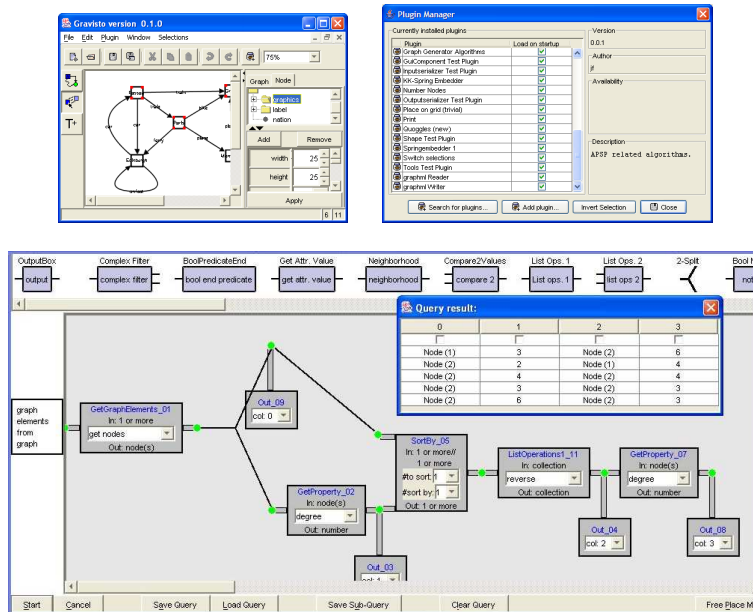
Fig. 2. *Gravisto*'s editing view, its plug-in manager, and the *Quoggles* interface

must be linked to additional information in a data base; thus all unnecessary plug-ins can be excluded and a custom plug-in for the data base connection may be added. Of course, the toolkit includes all plug-ins for the basic functionality and many more. Plug-ins for the widely used GML and GraphML file formats allow data exchange with non-Java applications and other graph drawing tools.

Queries on Graphs: An innovative feature is *Quoggles*, [3], a plug-in implementing an extensible, graphical query system for graph properties. The idea for *Quoggles* arose from the 10th Graph Drawing Contest, 2002, Category C, [1]. The query itself is composed into a graph. The input to the query are the set GE of all nodes and edges of the queried graph. For example, the query "$GE \rightarrow GetGraphElements(nodes) \rightarrow GetProperty(degree) \rightarrow Arithmetic(avg)$" calculates the average degree of all nodes in the graph. The query shown in Fig. 2 sorts the nodes according to their degree.

References

[1] F. J. Brandenburg. Graph-drawing contest report. In M. T. Goodrich and S. G. Kobourov, editors, *Proc. Graph Drawing, GD 2002*, volume 2528 of *LNCS*, pages 376–379. Springer, 2002.

[2] *Gravisto*. http://www.gravisto.org/. University of Passau.

[3] P. Holleis. Design and implementation of an extensible query system for graphs. diploma thesis, University of Passau, 2004.