To appear in: R. Tamassia, I. Tollis: “Proceedings of DIMACS Workshop on Graph Drawing ‘94, Princeton (USA)”, Lecture Notes in Computer Science, Springer Verlag, 1995

# Demonstration of the interactive Graph Visualization System *daVinci*

Michael Fröhlich and Mattias Werner

University of Bremen, Institute for Formal Methods in Software-Engineering,
P.O.-Box 330 440, D-28334 Bremen, Germany
e-mail: daVinci@Informatik.Uni-Bremen.DE

**Abstract.** We present the graph visualization system *daVinci*, an interactive tool that can be used by arbitrary application programs as a user interface for graph data structures. Beside a novel automatic layout algorithm for graphs, *daVinci* offers many interactive facilities such as fine-tuning of a layout, abstractions and scaling operations. A bidirectional application interface is used for tool communication with a program that controls the graph structure.

## 1. Introduction

The visual representation of directed graphs becomes more and more important in modern computer applications. Graphs are widespread, universal data structures that can be found in many different forms and applications. In computer programs, a powerful visual representation is needed to give the user an intuitive image of an internal graph structure. Handling the application by interacting with the graph visualization (as with a graph editor) is even better to give the user a more precise feedback about the program. Although graph visualization is important for many domains, the use of this technique is not very common in today's computer applications. Frequently, a user has to deal with uncomfortable textual interfaces or poor ad-hoc drawings of graphs. The reason for this deficit is the great effort to implement a satisfying graph layout. Furthermore, flexible and powerful graph visualization tools, which are reusable and easy to integrate with applications, are hard to find. These facts were the motivation for developing the interactive graph visualization system *daVinci*.

In the past decade, a lot of powerful visualization tools for directed graphs have been developed such as dot [1], GraphEd [2], the prototype of Henry [3], VCG [4], and many others. Unfortunately, most of the available systems are designed as isolated viewing components and do not provide distinct methods for communicating with application programs. They can only be used to display graphs, but not to develop them interactively in conjunction with a controlling application program. Systems demonstrating the opposite are Edge [5] and GraphView [6]. Both tools offer interfaces to graph generating programs, but without providing the manifold capabilities of *daVinci's* application interface. Our approach joins two requirements: a connected application keeps full control on the displayed graph and the user gets the impression to handle the application by interacting with the graph visualization.

## 2. Graph Layout

*daVinci* loads a cyclic or acyclic digraph from an ASCII term representation. By using attributes in the term, one can define the visualization, for example by specifying text, font, color or shape of nodes. Icons of arbitrary size are supported as well as multiline strings for all kinds of graphical objects.

*daVinci's* layout procedure for hierarchical graph visualizations is based on the heuristic algorithm of Sugiyama et al. ([7]) and was improved in certain details to obtain better visualizations. For example, *daVinci* is able to draw the world model graph (used in [7] for illustration) with only one third of the edge crossings found in the layout of the original algorithm (refer to [8] for details). To optimize the layout quality and speed of trees, a new linear time tree layout algorithm, based on the work of Juutistenaho ([9]), is implemented. We have improved this algorithm to reduce the number of tree traverses from two down to one.

## 3. Functionality

In contrast to other tools with static visualizations, *daVinci* provides several interactive operations to work with the generated graph drawing. Standard representations, calculated by graph layout algorithms, are usually good, but rarely perfect, so a user should have **fine tuning** capabilities to get optimal results (e.g. for presentations or publications). After processing the layout algorithm initially, it is possible to adjust the position and order of graph nodes and edges interactively with the mouse. Fine tuning in *daVinci* is an embedded function, so based on a fine tuned graph, the layout algorithm can be started again to obtain better results.

In case of large graphs, abstraction techniques are needed to cope with complexity. Two **interactive abstractions** are implemented in *daVinci* at the moment: subgraph- and edge hiding. Subgraph hiding is used to collapse the subgraph of a selected node. With edge hiding all edges of a selected node can be omitted in the visualization. With interactive abstractions a user is able to suppress uninteresting or obstructing parts to simplify the visualization. To get an overview, one can reduce the scale of a visualization to any rate. After using **scaling**, all the other interactions (e.g. fine tuning, abstractions) are still available.

The state of a visualization can be saved in a file to preserve interactions for a later session. In the same way, a graph can be saved in Encapsulated PostScript Format for printing. Informations about a graph (e.g. number of nodes, edges, crossings, etc.) can be displayed on demand.

## 4. Application Interface

It was the intention to create a generic and reusable tool which could be used by application programs as a comfortable user interface for graphs. One fundamental concept is the strict separation of tasks between both systems. The connected application, which is an autonomous process, is exclusively responsible for controlling and modifying the graph structure, and the only task of *daVinci* is to display this graph on the screen. So in opposite to usual graph editors, *daVinci* is not able to alter the graph by itself. To realize this concept, a bidirectional, pipe-based application interface is provided for tool communication.

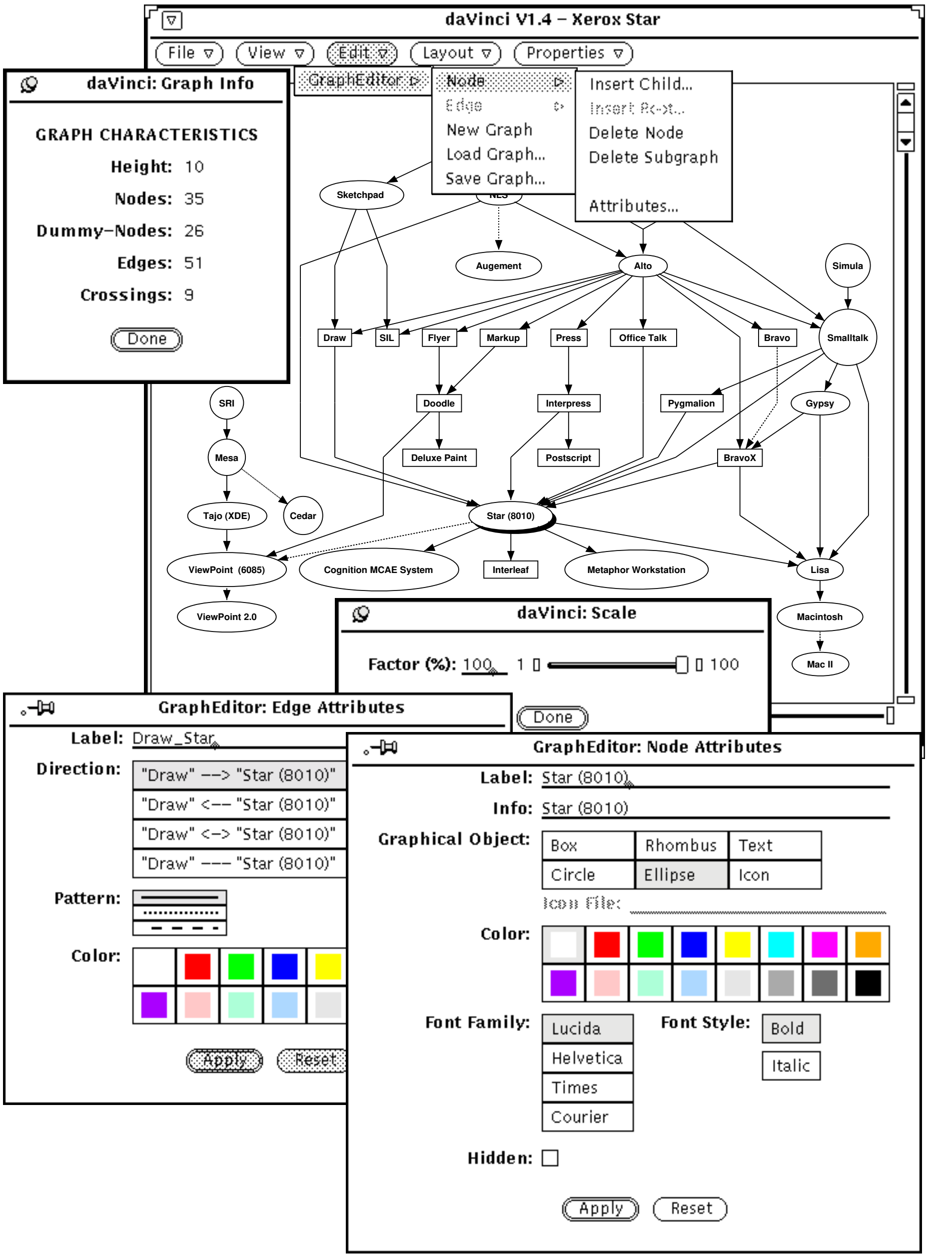


**Fig.1.** This figure shows a graph visualization with *daVinci*, representing dependencies of systems related to the Xerox Star project. Two subwindows are visible for setting the scale and for displaying additional informations. The lower two windows belong to an autonomous graph editor program which is connected to *daVinci's* application interface. The editor has added some extra menus to the user interface of *daVinci* for manipulating the displayed graph interactively (see menu “Edit”).

The first pipe is used by the application to send commands to *daVinci*. Many different commands are available for transfering graphs, adding application specific menus to *daVinci's* user interface, starting dialogues with the user, showing status informations or controlling the visualization system. On the second pipe, an application is informed about events triggered by the user such as selection of nodes, edges and previously created menus or user input in a dialogue. An application has to interpret the events in its own context. Depending on the event, the application can modify its internal data structure and send back an updated graph to display the changes. This way any program can offer the user the opportunity to control the application by interacting with a graph visualization.

More than 500 educational and commercial sites are already using *daVinci*. Existing applications are manifold and cannot be enumerated in this paper. More informations about *daVinci* can be found in [8] or are available by the authors.

# References


1. E. Koutsofios, S. C. North: “Drawing Graphs with dot - dot User's Manual”. Technical Report, AT&T Bell Laboratories, Murray Hill, New Jersey (1993).
2. M. Himsolt: “GraphEd: A Graphical Platform for the Implementation of Graph Algorithms”. In these proceedings.
3. T. R. Henry: “Interactive Graph Layout: The Exploration of Large Graphs”. Ph. D. Thesis, Technical Report No. **92-03**, University of Arizona (1992).
4. G. Sander: “Graph Layout through the VCG Tool”. In these proceedings.
5. F. Newbery-Paulisch: “The Design of an Extendible Graph Editor”. Lecture Notes in Computer Science No. **701**, Springer Verlag (1993).
6. B. Birgisson, G. Shannon: “GraphView: An Extensible Interactive Platform for Manipulating and Displaying Graphs”. Technical Report **No. 295**, Computer Science Department, Indiana University, Bloomington, Indiana (1989).
7. K. Sugiyama, S. Tagawa, M. Toda: “Methods for Visual Understanding of Hierarchical System Structures”. IEEE Transactions on System, Man, and Cybernetics, Vol. **11**, No. **2** (1981) pp. 1047-1062.
8. M. Fröhlich, M. Werner: “The Graph Visualization System *daVinci* - A User Interface for Applications”. Technical Report No. **5/94**, Department of Computer Science, University of Bremen, Germany (1994).
9. A. Juutistenaho: “Linear Time Algorithms for Layout of Generalized Trees". Technical Report No. **A-1994-6**, Department of Computer Science, University of Tampere, Finland (1994).