CrocoCosmos

Claus Lewerentz, Frank Simon, and Frank Steinbrückner

Software & Systems Engineering Research Group
BTU Cottbus, D-03013 Cottbus, Germany
www.software-systemtechnik.de

1 Short Description

The ***CrocoCosmos*** tool was developed as part of the ***Crocodile*** tool set for the analysis and visualization of large object-oriented software systems. The context of our research is to support maintenance and re-engineering processes in an appropriately automated way for large programs. One aspect is program comprehension through the visualization of program structures on an architectural level. Thus, CrocoCosmos is not a general graph drawing tool but serves a dedicated purpose in a specific graph drawing application.

2 Areas of Application

Our approach to visualize large object-oriented programs (10^6 LOC, $10^3 - 10^4$ classes, 10^2 subsystems) on the basis of extracted structure and metrics data uses attributed 3D graphs. Nodes in such graphs represent structure entities like classes or packages. They are visualized by simple geometric objects (as spheres or cubes) with geometrical properties (as color or size) representing software metrics values. Relations are displayed as straight lines colored according to their relation type (method usage, inheritance). The resulting graphs for typical programs have several thousand nodes and several ten to hundred thousand edges.

The visualizations are one of the analysis tools output that complement multiple tabular and chart representations of metrics values and cross reference browsing structures. They are used both to get initial overview pictures of large systems as well as exploring particularly interesting parts of systems in detail. The first results from several case studies that were done together with industrial software developers are very encouraging [2]. The 3D visualization proved to be a very effective means to quickly detect and explain typical design weaknesses and to give restructuring recommendations on the basis of simple visual patterns.

3 Layout Algorithms and Layout Features

A central idea for drawing these graphs is the use of a generic similarity and distance concept that allows to calculate metric distances for each pair of nodes [1]. The distances may be calculated from arbitrary common property sets of the

P. Mutzel, M. Jünger, and S. Leipert (Eds.): GD 2001, LNCS 2265, pp. 446–447, 2002.

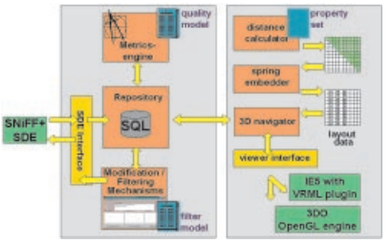

Fig. 1. Crocodile/CrocoCosmos Architecture

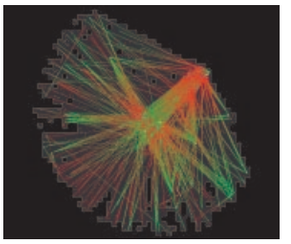

Fig. 2. Visualization of a program with 1.200 classes and 220.000 lines of Java code

nodes. To calculate a layout of the graph, the metric distances between nodes are used as weights of the edges in the complete graph. The 3D graph layout algorithm is based on a spring-embedding method that is combined with simulated annealing techniques to control the iteration processes. It takes the edge weights and produces a layout which approximately preserves the node distances on an ordinal scale in the 3D space. In the resulting graph layouts, that contain all nodes and the edges for the structural relations as call or inheritance relations the spatial relationships are meaningful and can be interpreted with respect to the problem domain.

4 Architecture

The Crocodile tool consists of a parsing frontend, that fills a relational database containing all structural and metrics data. CrocoCosmos is the visualization backend, consisting of a distance calculator, the spring-embedder, and a high-performance 3D display engine (cf. Figure 1). Except the display engine which is implemented directly on top of OpenGL, the system is written in Java.

5 Screenshot

Figure 2 shows an example depicting more than 1.200 classes of a Java program with more than 25.000 use relationships between them. The layout reflects distances according to similarities in usage relations between classes. Interaction and navigation mechanisms allow for a user-driven exploration of the graphs using 3D display devices.

References

1. F. Simon, S. Löffler, and C. Lewerentz. Distance-based cohesion measuring. In *Proc. 2nd Conference on Software Measurement (FESMA'99)*, pages 69–84, 1999.
2. F. Simon, F. Steinbrückner, and C. Lewerentz. Anpassbare explorierbare virtuelle Informationsräume. In *Proc. 3rd GI Workshop on Software Reengineering*, 2001.