Web 3D Software Visualisation

Craig Anslow
Victoria University of Wellington, New Zealand
craig@mcs.vuw.ac.nz

Synopsis

Software visualisation is the use of the crafts of typography, graphic design, animation, and cinematography with modern human-computer interaction and computer graphics technology to facilitate both the human understanding and effective use of computer software [3]. However, I am are more focused on program visualisation which is the visualisation of actual program code or data structures in either static or dynamic form.

The potential use of 3D graphics for program visualisation is significant and mostly unexplored [3]. The use of 3D graphics is not to enhance the beauty of a program visualisation; instead it provides additional information to that of 2D. A great deal of experimentation is needed to better understand the strengths and weaknesses of using interactive 3D graphics for software visualisation.

I am interested in visualising the structure and behaviour of object-oriented software over the web in 3D. Previous work I was involved with describes an architecture for supporting the visualisation of software in a distributed environment [2, 1]. The design supports components in multiple object-oriented languages and configurations (e.g. complete programs or just code fragments), and provides user control for all parts of the visualisation process. The architecture requires tools to develop and deliver visualisations from execution traces [1] that are easy to learn and are intuitive to understand for developers.

I have built a web-based software visualisation tool called VET3D (Visualising Execution Traces in 3D) that transforms XML execution traces into X3D [4] – the Web3D Consortium's open standard for web based 3D graphics – visualisations. X3D is an XML language for 3D content delivery on the web and combines both geometry and runtime behaviour into a single XML file. X3D can be displayed in a native X3D browser or a web browser that has an X3D plug-in.

VET3D allows users to make queries from a web browser. Users can test drive[2] remotely executing software components to produce an XML execution trace, transform XML execution traces using XSLT into X3D visualisations, and display X3D visualisations in a web browser such as node/link diagrams.

[1] Execution traces contain static and dynamic information of software components including the events that happened during the execution of a component

[2] Test driving is defined as specifying a sequence of method invocation and field access/modifications and then executing the sequence on a software component.

Separating the test driving and creation of visualisations steps allows users to test drive components and then create visualisations in the future. Users can view stored X3D visualisations without having to test drive remote software or transform XML execution traces. New types of visualisations can be created by adding in new style-sheets. Existing visualisations can be modified to add in new controls such as sliders to display only a certain number of nodes in a visualisation.

In summary I have built a tool that can produce visualisations of software in X3D over the web. The visualisations will help assist developers to understand the structure and behaviour of software.

References

[1] S. Marshall, K. Jackson, C. Anslow, and R. Biddle. Aspects to visualising reusable components. In T. Pattison and B. Thomas, editors, *Information Visualisation 2003*, *CRPIT Vol 24*. Australian Computer Society, 2004.

[2] S. Marshall, K. Jackson, R. Biddle, M. McGavin, E. Tempero, and M. Duignan. Visualising reusable software over the web. In P. Eades and T. Pattison, editors, *Information Visualisation 2001*, *CRPIT Vol 9*. Australian Computer Society, 2001.

[3] J. Stasko, M. Brown, and B. Price. *Software Visualization*. MIT Press, 1998.

[4] Web3D-Consortium. *X3D Specification*. `http://www.web3d.org/x3d/specifications/`, 2005.