X3D-Earth in the Software Visualization Pipeline

Craig Anslow, Stuart Marshall, and James Noble

Victoria University of Wellington, New Zealand
{craig,stuart,kjx}@mcs.vuw.ac.nz

Abstract

We need tools and new ways to visualize our software execution traces and to deploy and integrate the visualizations into users' environments. We are currently implementing existing information visualization 3D metaphors using X3D to visualize our execution traces, however most of these metaphors are principly node-link diagrams or graph structures and are not easy to interpret nor understand. We want to use real world metaphors which users can easily relate to, such as a 3D city metaphor. One tool for implementing the 3D city metaphor could be the open source X3D-Earth application. In this paper we list the requirements for an X3D-Earth application which could be used for visualizations of static and dynamic information of software.

Keywords Software Visualization, Execution Traces, X3D-Earth

1. Introduction

Software visualization [19] 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. We are interested in visualizing the static and dynamic information of software.

The mapping of static and dynamic information to an image is defined through a metaphor which specifies the type of visualization. We are currently evaluating the effectiveness of X3D in the software visualization pipeline by implementing existing information visualization 3D metaphors such as information landscapes [1] and information cubes [15].

Eick [7] claims that the information visualization resarch challenge is how to invent new visual metaphors for presenting information and developing ways to manipulate these metaphors to make sense of the information. The information landscape and information cube are essentially node-link diagrams or graph structures which use geometric shapes. These kind of graph metaphors are not intuitive to interpret and users must learn the semantics of them in order to understand the information the visualization is displaying. Hence we would like to investigate using metaphors from the real world where users already have an understanding of how to manipulate the metaphor and make sense of the information.

We propose to use a 3D city metaphor for visualizing software. One tool for implementing the 3D city metaphor could be the open source X3D-Earth application. Using the X3D-Earth engine in the software visualization pipeline will enable developers to create 3D web visualizations of software in a much easier manner. The aim of this paper is to document the requirements for an X3D-Earth application for use in software visualization.

This rest of the paper is organised as follows. In section 2 we look at the software visualization pipeline and implementations of existing node-link 3D visualizations in X3D. In section 3 we list our requirements for an X3D-Earth application. In section 4 we mention related work and in section 5 we discuss possible companies that may be interested in contributing to an X3D-Earth project. We conclude our ideas in section 6.

2. X3D in the Software Visualization Pipeline

For the software visualization pipeline we have created the Visualisation Architecture for REuse (VARE) [13]. VARE is designed for web-based visualizations of remotely executing object-oriented software. VARE is a client-server architecture. Essentially a user selects a software component and then uses an engine to test drive [1] the component. The engine generates an execution trace as output which contains all information required to describe a program execution such as the order of object creation, method invocations, field accesses, and field modifications. The execution trace is then used as input to a transformer to be transformed into a visualization.

We have built a number of tools based on the VARE architecture that test drive C++ [13] and Java [11] programs, map the outputs to XML execution traces [12, 3], and then transform the execution traces into SVG [6] and X3D [2] visualizations using XSLT. Figure 1 summarizes how we use XML in the software visualization pipeline. We now elaborate on some of the visualizations we have created in X3D using existing 3D information visualization metaphors.

Figure 2 shows the layout of all the events from an execution trace of a Java program as an information landscape. The information landscape is essentially 2 1/2 D rather than 3D. A red box represents the main class, blue spheres object creation, green boxes as method calls, white cones as method returns and end of the object. The image starts at the red box and is animated from right to left.

Figure 3 shows the same information as Figure 2 but displayed as an information cube. The information starts with the main method red box, followed by the first object creation blue sphere at the top of the cube then continuing along the links to each of the object creation blue spheres. The spheres are transparent and the events that an object executes are encompassed within the sphere, i.e. method calls, method returns and end of an object.

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

X3D Earth Requirements'06 November 14–15, 2006, Naval Postgraduate School, Monterey, California, USA.
X3D Earth Requirements Workshop.

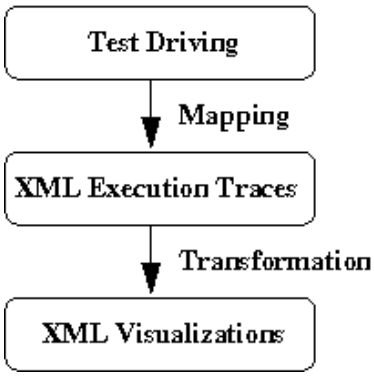

Figure 1. XML in the software visualization pipeline

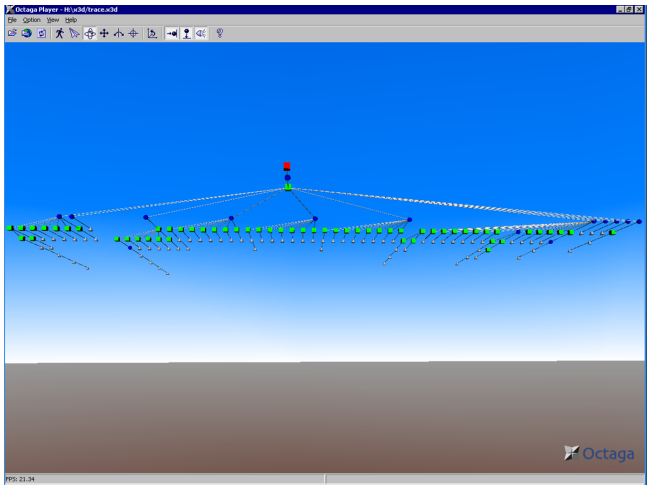

Figure 2. Information Landscape

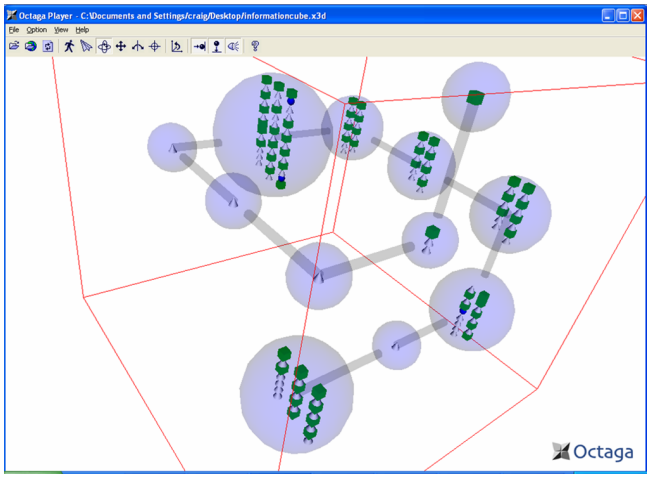

Figure 3. Information Cube

We are continuing to implement other 3D metaphors such as cone trees [16] and compound treemaps [5]. However most of these 3D metaphors are simply node-link diagrams or graph structures and are not easy to understand or interpret. We are interested in exploring alternative real world metaphors where users already have an understanding of how to manipulate the metaphor and make sense of the information. We next look at visualizing software using a 3D city metaphor.

3. 3D City Metaphor for Visualizing Software

Most node-link diagrams and graph structures do not have an intuitive interpretation and users must learn the semantics of them in order to understand the visualization. We believe visualizing software as a 3D city metaphor may improve a users' understanding of software over other 3D information visualization metaphors. The 3D city metaphor is a natural (real world) metaphor where users already understand and know how to use their perceptual and spatial skills. We will not be able to determine if a 3D city metaphor is actually better than other metaphors for visualizing software until extensive usability studies are conducted. However the purpose of our research is to evaluate the effectiveness of X3D in the software visualization pipeline by implementing existing 3D information visualization metaphors.

One tool for implementing the 3D city metaphor could be X3D-Earth. We now list the requirements for an X3D-Earth application to visualize software based on previous X3D software visualization experience.

One of the requirements of our visualization architecture is that it be in a distributed network. Our main focus is for our tools to be web enabled, so X3D-Earth must be able to be viewed in a web browser. We want to be able to transform our XML execution traces into X3D-Earth files, so providing tools or stylesheets for this transformation process would be essential.

For navigation within X3D-Earth we need to at least provide the basic X3D navigation options of walk, slide, examine, fly, pan, and look-at a node. Since it is a 3D world we want to go inside buildings, vehicles, walk on land, and dive under the sea. Essentially any way a human can navigate in the real world we would like to be able to replicate in X3D-Earth.

For static information we want to be able to represent packages, classes, fields, objects, method calls, method returns, field access, and field modifications in a software visualization. An example way to represent these items in an X3D-Earth visualization would be to map countries as packages, cities as classes, buildings as methods, fields as levels in a building, and roads that link cities. The layout of the countries and cities could show inheritance hierarchies based on the capital city and population demographics and clustering of cities could show coupling between classes.

For dynamic information vehicles could be used to travel through oceans, rivers, over roads to show the interactions between objects. When many vehicles are moving at once this could represent high communication amongst certain objects and show diferent threads of control. When vehicles colide or bridges collapse sounds or animations could be played to represent when an error or exception has occured in a program. Displaying natural disasters such as earthquakes or volcanoes erupting could represent programs crashing.

There are various tasks [18] that we would like to be able to do in X3D-Earth. We want to be able to create an overview of the earth and view the countries (packages) that exist. We want to have functions for zooming in or filtering out countries, cities or buildings of interest. When viewing a certain aspect in the 3D world such as a country or city we would like to provide details-on-demand. This could include clicking on a country (package) and then seeing the list of cities/towns (classes) that exist or even the names of pa-

rameters that are passed to a method. We would like to be able to show relationships amongst packages and classes which could be represented as all the countries in the commonwealth for a certain application or sister cities to represent classes that communicate with each other from different packages. When viewing or exploring X3D-Earth we would like to be able to keep a history of the paths that a user or visualization has taken and to save viewpoints of interest. This could include the animation of vehicles moving between cities to show different threads of a program. Finally we would like to be able to extract information which would be relevant for offline viewing or storing for later retrieval. Extracting sub collections of X3D-Earth files could show how software (countries/cities) evolves over time.

4. Related Work

We are interested in visualizing software using a 3D City metaphor and implemented with X3D-Earth. Some researchers have also explored using the 3D city metaphor. Knight and Munro [9] created a source code comprehension tool – Software World – within a desktop virtual reality system using C code, see Figure 4. They also created a software component visualization tool – Component City – using XML and VRML [4]. Both of these systems list a set of mappings from real world ideas to software abstract ideas. However Neither Software World nor Component City addresses dynamic information from code or components. Panas et al. [14] have created a 3D city for software production visualization using 3D Studio Max and they intend to create a city using OpenGL4Java, see Figure 5.

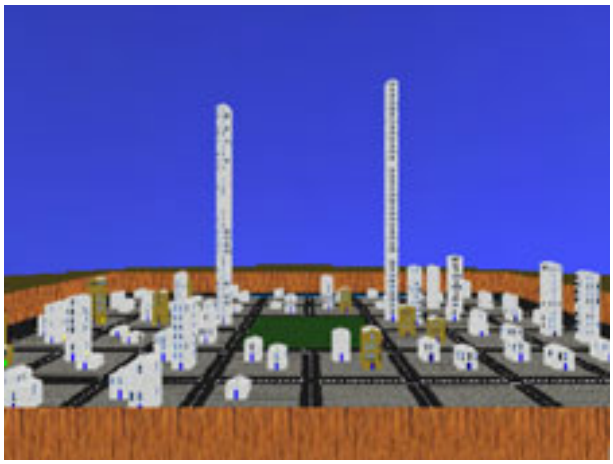

Figure 4. Software World [9]

Figure 5. 3D city metaphor for software production visulization [14]

Santos et al. [17] have used VRML for mapping information information onto 3D virtual worlds that convey network data for network monitoring and management, see Figure 6. Kot et al. [10] have used a 3D game metaphor that uses Doom3D for a source code comprehension tool, see Figure 7. Finally Irwin and Churcher [8] have also used XML in the visualization pipeline to produce VRML visualizations to display metrics of object-oriented software.

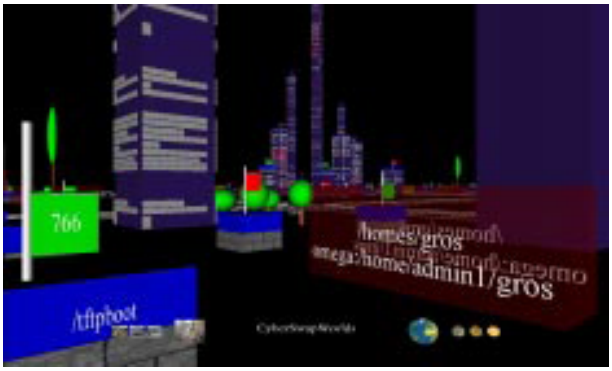

Figure 6. Example of the city metaphor for network data visualization [17]

Figure 7. Source code comprehension tool that uses Doom3D [10]

5. Discussion

In New Zealand there are number of companies working on 3D mapping technolgies that the Web3D consortium could have potential strategic partnerships with. Some of these companies are now listed.

Terralink International provides Geographic Information Systems (GIS) and mapping solutions. Terralink was once owned by the New Zealand Government but has since been privatised in 2001. Animation Research Holdings Ltd, a New Zealand company recognised globally for its innovative 3D animation achievements is an 80% shareholder. Terralink provides all the information for map publications in New Zealand and have various web applications to view this data. Terralink recently provided a tertiary education grant to a masters student in the School of Architecture and Design at Victoria University for a 3D, interactive, web enabled, multilayered model of Wellington City. The scholarship is valued at

($15000 NZD). The recipient is currently using Google Earth as an engine to display urban environments.

MetService provides weather and information presentation services to customers around the world. It has produced Weatherscape XT, which is the world's permier weather graphics system used by leading broadcasters such as BBC News, BBC World, Nine Network Australia, TG4 Ireland and international CNBC stations. Weatherscape XT uses 3D rendering technology, 3D graphics animation, computer technology and meteorological science to provide an automated weather presentation and production system.

ProjectX Technology is an international online map services company. They are a startup company and have existed since July 2005. They have developed two very useful tools to make online mapping easy. The first is ZoomIn Mapping System which is a toolkit that uses Ruby On Rails and can visualize your location based data. The second uses this system and is called ZoomIn. ZoomIn is a local search website for Australia and New Zealand. ZoomIn allows you to locate addresses, find businesses or services and interesting places you have never heard about or seen.

6. Conclusion

Our research focuses on evaluating the effectiveness of X3D in the software visualization pipeline. We are building tools that can produce X3D visualizations from execution traces over the web and have implemented some existing 3D information visualization metaphors. These metaphors are mainly node-link diagrams or graph structures and are not easy to interpret nor understand. We propose to use a real world metaphor such as the 3D city metaphor which users already relate to. One tool for implementing the 3D city metaphor could be the open source X3D-Earth application. In this paper we have listed some of the requirements for an X3D-Earth application that could support visualizations of static and dynamic information of software.

Acknowledgements

This work was partly funded by a DSTools grant. Thanks to James and Stuart for providing valuable feedback on this project. The X3D visualizations are viewed using the free Octaga player. Thanks to Malcolm Munro for providing an electronic copy of Claire Knight's PhD thesis.

About the Authors

Craig Anslow is currently doing a MSc thesis degree in computer science in the area of software visualization at Victoria University of Wellington, New Zealand. The thesis is to investigate how appropriate X3D is for the use in the software visualization pipeline. Stuart Marshall is a lecturer in computer science at Victoria University of Wellington. James Noble is a Professor in computer science at Victoria University of Wellington.

References

[1] K. Andrews. Visualising cyberspace: information visualisation in the harmony internet browser. In *INFOVIS '95: Proceedings of the 1995 IEEE Symposium on Information Visualization*, page 97, Washington, DC, USA, 1995. IEEE Computer Society.

[2] Craig Anslow, Stuart Marshall, James Noble, and Robert Biddle. Vet3d: a tool for execution trace web 3d visualization. In *OOPSLA '06: Companion to the 21st ACM SIGPLAN conference on Object-oriented programming languages, systems, and applications*, pages 655–656, New York, NY, USA, 2006. ACM Press.

[3] Craig Anslow, Stuart Marshall, James Noble, Kirk Jackson, Mike McGavin, and Robert Biddle. Program trace formats for software visualisation. Technical report, School of Mathematics, Statistics and Computer Science, Computer Science. Victoria University of Wellington, 2006.

[4] Stuart M. Charters, Claire Knight, Nigel Thomas, and Malcolm Munro. Visualisation for informed decision making; from code to components. In *SEKE '02: Proceedings of the 14th international conference on Software engineering and knowledge engineering*, pages 765–772, New York, NY, USA, 2002. ACM Press.

[5] Neville Churcher, Lachlan Keown, and Warwick Irwin. Virtual worlds for software visualisation. In A Quigley, editor, *SoftVis99 Software Visualisation Workshop*, pages 9–16, 1999.

[6] Matthew Duignan, Robert Biddle, and Ewan Tempero. Evaluating scalable vector graphics for use in software visualisation. In *Proceedings of the Australian Symposium on Information visualisation*, pages 127–136. Australian Computer Society, Inc., 2003.

[7] Stephen G. Eick. Visualizing online activity. *Commun. ACM*, 44(8):45–50, 2001.

[8] Warwick Irwin and Neville Churcher. Xml in the visualisation pipeline. In *VIP '01: Proceedings of the Pan-Sydney area workshop on Visual information processing*, pages 59–67. Australian Computer Society, Inc., 2001.

[9] Claire Knight and Malcolm Munro. Comprehension with[in] virtual environment visualisations. In *IWPC '99: Proceedings of the 7th International Workshop on Program Comprehension*, page 4, Washington, DC, USA, 1999. IEEE Computer Society.

[10] Blazej Kot, Burkhard Wuensche, John Grundy, and John Hosking. Information visualisation utilising 3d computer game engines - case study: A source code comprehension tool. In *Proceedings of the 6th Annual Conference of the ACM Special Interest Group on Computer-Human Interaction (CHINZ 2005)*, pages 53–60, Auckland, New Zealand, 2005. ACM Press.

[11] Stuart Marshall. *Test Driving Reusable Components*. PhD thesis, Victoria University of Wellington, 2006.

[12] Stuart Marshall, Kirk Jackson, Craig Anslow, and Robert Biddle. Aspects to visualising reusable components. In *Proceedings of the Australian Symposium on Information visualisation*, pages 81–88. Australian Computer Society, Inc., 2003.

[13] Stuart Marshall, Kirk Jackson, Robert Biddle, Michael McGavin, Ewan Tempero, and Matthew Duignan. Visualising reusable software over the web. In *Proceedings of the Australian Symposium on Information visualisation*, pages 103–111. Australian Computer Society, Inc., 2001.

[14] Thomas Panas, Rebecca Berrigan, and John Grundy. A 3d metaphor for software production visualization. In *IV '03: Proceedings of the Seventh International Conference on Information Visualization*, page 314, Washington, DC, USA, 2003. IEEE Computer Society.

[15] J. Rekimoto and M. Green. The information cube: Using transparency in 3d information visualization. In *Third Annual Workshop on Information Technologies & Systems (WITS'93)*, pages 125–132, 1993.

[16] George G. Robertson, Jock D. Mackinlay, and Stuart K. Card. Cone trees: animated 3d visualizations of hierarchical information. In *CHI '91: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 189–194, New York, NY, USA, 1991. ACM Press.

[17] C. Russo Dos Santos, P. Gros, P. Abel, Didier Loisel, N. Trichaud, and J. P. Paris. Mapping information onto 3d virtual worlds. In *IV*, pages 379–386, 2000.

[18] Ben Shneiderman. The eyes have it: A task by data type taxonomy for information visualizations. In *IEEE Visual Languages*, number UMCP-CSD CS-TR-3665, pages 336–343, College Park, Maryland 20742, U.S.A., 1996.

[19] John T. Stasko, Marc H. Brown, and Blaine A. Price. *Software Visualization*. MIT Press, 1998.