Co-located Collaborative Software Visualization

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

Stuart Marshall
James Noble
Victoria University of Wellington
New Zealand
{stuart,kjx}@ecs.vuw.ac.nz

Robert Biddle
Carleton University, Canada
robert_biddle@carleton.ca

ABSTRACT

Most software visualization tools are designed from a single-user perspective and are bound to the desktop, IDEs, and the web. Few tools are designed with sufficient support for the social aspects of software engineering such as collaboration, communication, and awareness. Our research aims at supporting co-located collaborative software analysis using software visualization techniques with multi-touch tables. The research will be conducted via user experiments which will inform the design of multi-touch software visualization applications and further our understanding of how developers work together with co-located collaborative tools.

Categories and Subject Descriptors

H.1.2 [**User/Machine Systems**]: Human Factors; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces—*Evaluation/methodology*

Keywords

Co-located collaboration, multi-touch, software visualization

1. INTRODUCTION

Understanding the complex structure and behaviour of large software systems is an important task for software maintenance. Software visualization aims to help with techniques to visualize the structure, behaviour, and evolution of software [2].

Understanding software is often a social activity and involves teams of software developers. The field of Computer Supported Cooperative Work (CSCW) explores how users behave with digital tools during collaborative work such as understanding software. Few studies have explored how tools support collaborative software understanding [5] and collaborative software visualization [8].

Multi-touch is an interactive technique that allows single or multiple users to collaborate to control graphical displays with more than one finger on various kinds of surfaces and devices. The goal of our research is to investigate whether multi-touch table interaction techniques are more effective for co-located collaborative software visualization than existing single-user desktop techniques. The research will involve developing a multi-touch software visualization tool and collecting qualitative data via user experiments.

2. SOURCEVIS

Effective software visualization techniques help users understand one or several fundamental aspects of complex software systems [2]. One aspect of software systems that would benefit from visualization support is software metrics, such as those that capture the static information around package, class or method size and interdependency. Polymetric Views are techniques that can visualize software metrics data [6]. A recent study of ours asked 14 participants to identify key measurements and comparisons of the package and classes from the Java Standard API version 1.6 using a modified version of one of the Polymetric Views [1]. The results indicated that users were able to effectively use the modified Polymetric View to explore the example domain.

We are developing a multi-touch software visualization tool called *SourceVis* that implements Polymetric Views to support co-located collaborative software visualization. Figure 1 shows an early prototype displaying packages and classes of a software system. Users can move and resize packages, and select classes to see the inner details.

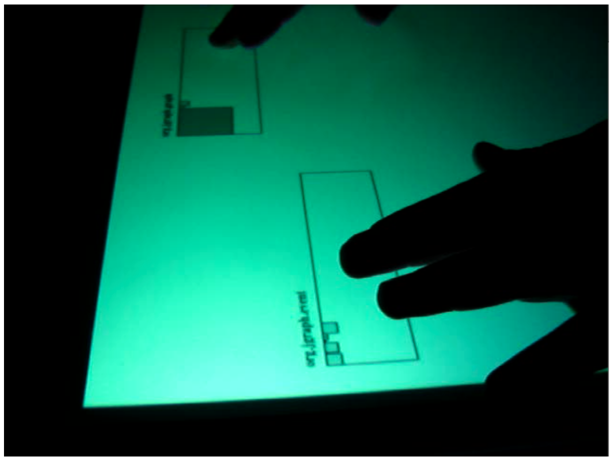

Figure 1: SourceVis.

3. USER EXPERIMENTS

We will use a qualitative research approach for our research methodology as it is well suited for giving an overview of a situation and to examine how and why certain users behave in certain environments. Some qualitative approaches include observational studies conducted as part of the design process, in situ interviews, and field studies. The CSCW community has adopted these approaches to understand the different behaviours of users with tools during collaborative work. Some researchers have used these approaches to explore the design of tabletop displays for co-located information visualization [3] and collaborative design [7].

The design of SourceVis and the user experiments will follow an iterative cycle using a grounded evaluation process to validate the design decisions [4]. We will recruit professional and student software developers for the user experiments from local mailing lists and from within our department. The user experiments will involve within-subjects testing. Each experiment will have up to three users working as a group using different tools. The tools include SourceVis on a multi-touch table that is 1077mm (width) x 673mm (depth) x 930mm (height) (approximately 50 inches) and desktop tools that implement Polymetric Views using a 20 inch computer display. Each group will start with either SourceVis or one of the other tools and then switch around. The purpose of a within-subjects test is to see whether the different tools influence the groups behaviour, and how the tools may lead to different kinds of discoveries. We will record the actions of the participants with screen capture software and video recording.

A representative sample of Java applications from the Qualitas Corpus will be used in the experiments [9]. Different applications will be used with each tool. This is to avoid the bias of users becoming experts from learning the structure of one application with a tool and then using the knowledge gained about that application with another tool.

The tasks for each experiment will involve answering questions about the structure of a Java application. Example questions could include: "what are the largest packages and classes in the application?", "what class has grown the most between the different versions of the application?", and "what classes are coupled the most with other classes?"

4. CHALLENGES

A significant barrier to exploring multi-touch table applications is the cost of the necessary hardware and software. Commercial multi-touch tables such as Microsoft Surface, Mitsubishi DiamondTouch, or Smart Touch Tables cost many thousands of dollars and are well beyond the reach of most consumers and research labs. We have addressed this problem by building our own low cost rear diffused illumination multi-touch table using Community Core Vision.

Desigining multi-touch table applications requires catering for users viewing the application from different orientations and angles as users need to stand at the table to interact. The application needs to be designed to accommodate the large size of the touch surface such that objects can be easily manipulated and text is readable.

SourceVis will support many of the user defined set of multi-touch gestures [10]. Some of these gestures include tap for select, dragging an object with one finger, rotating and resizing an object with two fingers, manipulating the background by zooming in and out and scrolling up and down, and grouping objects by drawing a shape around them with one finger. During the iterative development cycle domain specific gestures for software visualization will become apparent and where necessary these gestures will be implemented within the visualizations. There will also need to be gestures for users to switch between different views.

To provide comprehensive documentation in addition to the visualizations the source code from the Java applications and associated Javadoc will need to be augmented. For example a user might want to to select a package or class in the visualization but also see the source code or associated Javadoc page at the same time.

Acknowledgments

This work is supported by the New Zealand Foundation for Research Science and Technology for the Software Process and Product Improvement project, and TelstraClear.

5. REFERENCES

[1] C. Anslow, J. Noble, S. Marshall, E. Tempero, and R. Biddle. User evaluation of polymetric views using a large visualization wall. In *Proceedings of the Symposium on Software Visualization (SOFTVIS)*. ACM, 2010.

[2] S. Diehl. *Software Visualization: Visualizing the Structure, Behaviour, and Evolution of Software.* Springer Verlag, 2007.

[3] P. Isenberg. *Collaborative Information Visualization in Co-located Environments.* PhD thesis, University of Calgary, 2009.

[4] P. Isenberg, T. Zuk, C. Collins, and S. Carpendale. Grounded evaluation of information visualizations. In *Proceedings of the AVI Workshop on BEyond time and errors: novel evaLuation methods for Information Visualization (BELIV)*. ACM, 2008.

[5] A. J. Ko, R. DeLine, and G. Venolia. Information needs in collocated software development teams. In *Proceedings of the International Conference on Software Engineering (ICSE)*. IEEE, 2007.

[6] M. Lanza and S. Ducasse. Polymetric views-a lightweight visual approach to reverse engineering. *IEEE Trans. on Soft. Eng.*, 29(9):782–795, 2003.

[7] S. D. Scott, M. S. T., Carpendale, and K. M. Inkpen. Territoriality in collaborative tabletop workspaces. In *Proceedings of the Conference on Computer Supported Cooperative work (CSCW)*. ACM, 2004.

[8] M.-A. D. Storey, C. Bennett, R. I. Bull, and D. M. German. Remixing visualization to support collaboration in software maintenance. In *Proceedings of the Frontiers of Software Maintenance (FoSM)*. IEEE, 2008.

[9] E. Tempero, C. Anslow, J. Dietrich, T. Han, J. Li, M. Lumpe, H. Melton, and J. Noble. Qualitas corpus: A curated collection of Java code for empirical studies. In *Proceedings of the Asia Pacific Software Engineering Conference (APSEC)*. IEEE, 2010.

[10] J. O. Wobbrock, M. R. Morris, and A. D. Wilson. User-defined gestures for surface computing. In *Proceedings of the International Conference on Human Factors in Computing Systems (CHI)*. ACM, 2009.