SHriMP Views: An Interactive Environment for Information Visualization and Navigation

Margaret-Anne Storey
Dept of Aeronautics and
Astronautics, MIT
mstorey@mit.edu

Casey Best
Dept of Computer Science
University of Victoria
cbest@csr.uvic.ca

Jeff Michaud
Dept of Computer Science
University of Victoria
jmichaud@uvic.ca

Derek Rayside
IBM Centre for Advanced Studies
IBM Toronto Laboratory
drayside@acm.org

Marin Litoiu
IBM Centre for Advanced Studies,
IBM Toronto Laboratory
marin@ca.ibm.com

Mark Musen
Medical Informatics
Stanford University
musen@smi.stanford.edu

ABSTRACT

The SHriMP (Simple Hierarchical Multi-Perspective) visualization technique was designed to enhance how people browse and explore complex information spaces. SHriMP uses a nested graph view to present information that is hierarchically structured. It introduces the concept of *nested interchangeable views* to allow a user to explore multiple perspectives of information at different levels of abstraction. SHriMP combines a hypertext following metaphor with animated panning and zooming motions over the nested graph to provide continuous orientation and contextual cues for the user. In this demo, we show how these ideas are proving useful in the areas of software visualization, knowledge management and flow diagram visualization.

Keywords: Information visualization, information navigation, software tools, knowledge management, flow diagrams.

INTRODUCTION

Visualizing and navigating large amounts of information is a challenging problem faced in many knowledge domains. Over the past few years, the potential sources of information and the demands for integrating vast amounts of information have increased dramatically. Much effort is being expended on how to merge and integrate disparate heterogeneous sources of information as they may have very different underlying models, formats, and levels of granularity.

At the user interface level, the current methods of how to present these multiple, complex and large sources of information to knowledge engineers are unsatisfactory. Users have to understand the data sources acquired, often build models of the acquired data and may have to merge multiple knowledge bases and ontologies.

In this demo we present a visualization technique that, although originally developed for software visualization, has been customized to a variety of knowledge domains. In particular, we present three instantiations of the visualization methods and how they are applied to software engineering, knowledge engineering and flow diagram visualization. SHriMP has been designed using the Javabean component technology to enable easier customization and integration with other tools.

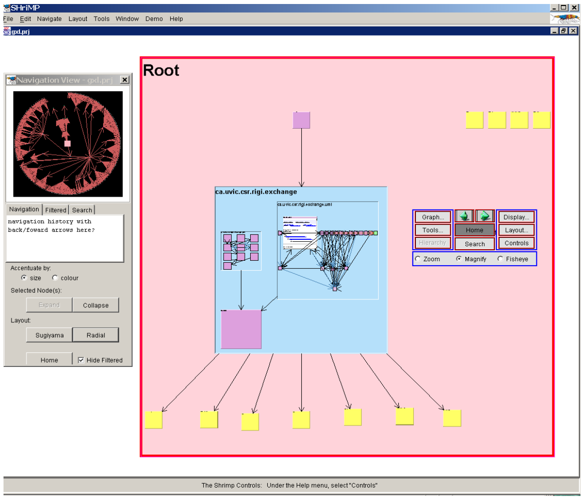
NESTED INTERCHANGEABLE VIEWS IN SHriMP

The SHriMP visualization technique was designed to enhance how people browse, explore and interact with complex information spaces [1]. SHriMP uses a nested graph view to present hierarchically structured information. It introduces the concept of *nested interchangeable views* to allow a user to explore multiple perspectives of information at different levels of abstraction. SHriMP combines a hypertext following metaphor with animated panning and zooming motions over the nested graph to provide continuous orientation and contextual cues for the user [2]. These features result in an environment where the user can interact directly with the information space enhancing his or her understanding of the information structures, thus promoting further exploration.

SOFTWARE VISUALIZATION

The first scenario shows how SHriMP can be used for exploring and navigating a Java program. In order to visualize a program in SHriMP, the user has to first parse the Java source code and documentation files to extract artifacts from the program (such as packages and classes) and dependencies between the artifacts. The artifacts and dependencies are modeled using a nested graph. The nesting relationships are used to model the implementation architecture of the program, such as when a package contains other packages and classes. Other tools are used to generate HTML'ized versions of the source code and documentation (javadocs). The graphical view, source code view, and javadoc view, are some of the interchangeable views available at different levels of abstraction. Fig. 1 shows how the concept of nested interchangeable views is used by a user to visualize Java programs. We believe that SHriMP is useful for programmers browsing their own code as well as for developers exploring unfamiliar software.

CHI 2002, April 20-25, 2002, Minneapolis, Minnesota, USA.
ACM 1-58113-454-1/02/0004.



navigation history with back/foward arrows here?

Accentuate by: ● size ○ colour

Selected Node(s):

Layout:

☑ Hide Filtered

○ Zoom ● Magnify ○ Fisheye

The Shrimp Controls: Under the Help menu, select "Controls"

Figure 1: Visualization of a Java program

ONTOLOGY VISUALIZATION

The second scenario demonstrates our integration of SHriMP with a knowledge management tool called Protégé [3]. The Protégé environment supports the modeling of ontologies to guide acquisition of content knowledge from domain experts. An ontology defines a common vocabulary and structure of an information space for researchers and domain experts to exchange and share knowledge. In this scenario, we show how Jambalaya (the term we use to refer to the integration of SHriMP and Protégé) can be used for browsing and exploring a medical knowledge base (see Fig. 2).

FLOW VISUALIZATION

For the third scenario, we describe how SHriMP has been integrated with the IBM Websphere Studio Workbench (www.eclipse.org). SHriMP has been customized and retargeted for visualizing *flow diagrams*. Flow diagrams are used in an eBusiness project to model the dynamic aspects of a system, such as the main activities and the movement of information and data in a business process. Flow diagrams can be hierarchically composed, thus leveraging the key features of SHriMP that allow a user to navigate hierarchically composed information spaces. Fig. 3 shows a message flow diagram, similar to those created with IBM's MQSI (Message Queue System Integrator) product.

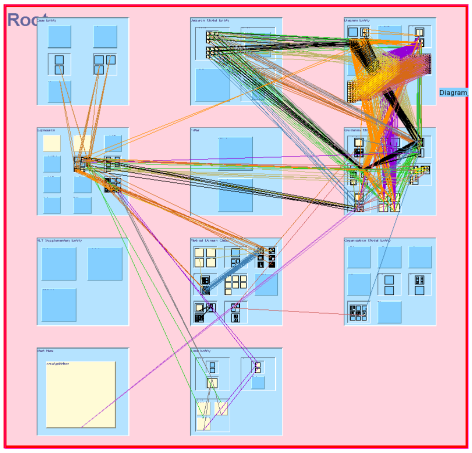
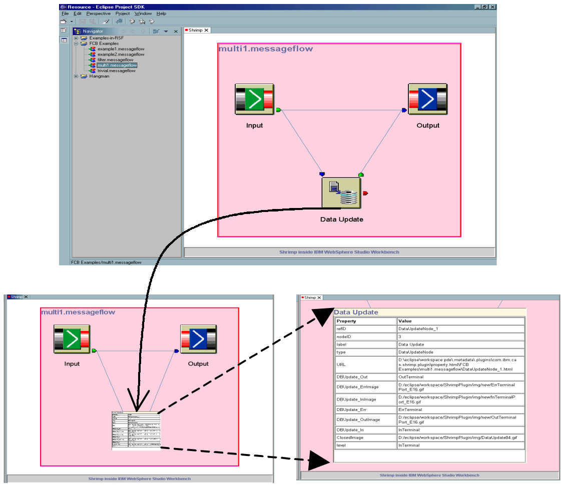


Figure 2: Using SHriMP to visualize a cancer clinical trial knowledge base in Protégé


Resource - Eclipse Project SDK
multi1.messageflow
Input
Output
Data Update
Shrimp inside IBM WebSphere Studio Workbench
multi1.messageflow
Input
Output
Shrimp
Data Update
Shrimp inside IBM WebSphere Studio Workbench
Shrimp inside IBM WebSphere Studio Workbench

Property	Value
refID	DataUpdateNode_1
nodeID	3
label	Data Update
type	DataUpdateNode
URL	D:\eclipse\workspace.pde\.metadata\.plugins\com.ibm.cas.shrimp.plugin\property.html\FCB Examples\multi1.messageflow\DataUpdateNode_1.html
DBUpdate_Out	OutTerminal
DBUpdate_ErrImage	D:/eclipse/workspace/ShrimpPlugin/img/new/ErrTerminalPort_E16.gif
DBUpdate_InImage	D:/eclipse/workspace/ShrimpPlugin/img/new/InTerminalPort_E16.gif
DBUpdate_Err	ErrTerminal
DBUpdate_OutImage	D:/eclipse/workspace/ShrimpPlugin/img/new/OutTerminalPort_E16.gif
DBUpdate_In	InTerminal
ClosedImage	D:/eclipse/workspace/ShrimpPlugin/img/DataUpdate84.gif
level	InTerminal

Figure 3: Visualizing flow diagrams in SHriMP

RELATED WORK AND CONCLUSIONS

There are some similarities with the magic lenses technique [5], but with magic lenses the user has to apply a tool to a region to change the view, whereas in SHriMP a region of information has certain views available and the views are tightly coupled with how the information is modeled and structured. Our earlier papers listed on [1] refer to other related work in information visualization. The nested interchangeable view technique has so far received favorable feedback from initial users (see [1] for references to several user studies we have conducted).

A key aspect in our research has been the lessons we have learned from applying this user interface in three different domains. Many of the ideas spawned in one domain turn out to be also applicable and powerful for the other domains. SHriMP's Javabean architecture has enabled us to integrate with other tools and techniques, providing many further avenues for research as well as providing very useful feedback on our approach.

ACKNOWLEDGMENTS

This research was supported in part by CSER, NSERC, IBM, Stanford University, the University of Victoria, the University of Texas at Austin and the Space and Naval Warfare Systems Center San Diego.

REFERENCES

1. The SHriMP Project, University of Victoria, http://www.shrimpviews.com/.
2. The Jazz Zoomable User Interface, University of Maryland, http://www.cs.umd.edu/hcil/jazz/
3. Protégé project, Stanford University, http://protege.stanford.edu
4. Rayside, D., Litoiu, M., Storey, M.-A., and Best, C., Integrating SHriMP with the IBM Websphere Studio Workbench, *CASCON 2001*, Toronto, Nov 2001.
5. Stone, M.C., Fishkin, K., and Ber, Eric. A., The Movable Filter as a User Interface Tool, *CHI 94*, Boston, Apr 1994.