Available online at www.sciencedirect.com
ScienceDirect
Electronic Notes in Theoretical Computer Science
ELSEVIER
Electronic Notes in Theoretical Computer Science 224 (2009) 141–149
www.elsevier.com/locate/entcs

Integrating Multiple Approaches for Interacting with Dynamic Data Structure Visualizations

James H. Cross II, T. Dean Hendrix, and Larry A. Barowski[1]

Department of Computer Science and Software Engineering
Auburn University
Auburn, Alabama 36849-5437 USA

Abstract

jGRASP 1.8.7 has integrated three approaches for interacting with its dynamic viewers for data structures: the debugger, the workbench, and a new text-based interactions tab that allows individual Java statements to be executed and expressions to be evaluated. While each of these approaches is distinct and can be used independently of the others, they can also be used together to provide a complementary set of interactions with the dynamic viewers. In order to integrate these approaches, the jGRASP visual debugger, workbench, and viewers had to be significantly redesigned. During this process, the structure identifier, which provides for the identification and rendering of common data structures, was also greatly improved by examining the examples from 20 data structure textbooks. The overall result of this integration effort is a highly flexible approach for user interaction with the dynamic data structure visualizations generated by a robust structure identifier.

Keywords: Program visualization, data structures, jGRASP, debugger, workbench, interactions.

1 Introduction

Visualizations of data structures have been used in limited ways for many years. To overcome a major obstacle to widespread use, namely lack of easy access and use, the jGRASP[2] IDE provides a structure identifier that attempts to identify and render traditional abstract visualizations for common data structures such as stacks, queues, linked lists, binary trees, heaps, and hash tables [2]. These are dynamic visualizations in that they are generated while running the users program in debug mode. This technique helps bridge the gap between implementation and the conceptual view of data structures, since the visualizations can be based on the users own code.

[1] Email: crossjh | hendrtd | barowla @auburn.edu
[2] The jGRASP research project is funded, in part, by a grant from the National Science Foundation.

doi:10.1016/j.entcs.2008.12.058

jGRASP 1.8.7 provides numerous ways for the user to interact with the data structure visualizations: (1) the debugger, (2) the workbench, and (3) a new text-based interactions tab. Each of these is briefly described below and then more fully with examples in the sections that follow.

The most common way to open a viewer on a data structure object is via the debugger. The user simply sets a breakpoint on a statement near the creation of the data structure, and runs the program in debug mode. After the program stops at the breakpoint, the user single steps as necessary until the object is created, and then opens a viewer on the object by dragging it from the debug tab. As the viewer is opened, the structure identifier determines the particular type of data structure and then renders it appropriately. When the user steps through the executing program, the viewer is updated to show the effects.

In addition to interacting with the data structure visualizations via the debugger, jGRASP allows the user to interact with viewers via the workbench. Objects can be created and their methods invoked via menus and buttons on the UML class diagram and/or the source code edit windows. When the user invokes a method on the object (e.g., to insert a node into the data structure), the visualization is updated to show the effect of the method.

The third approach for interacting with the viewers is a text-based interactions tab, which is essentially a Java interpreter. That is, when the user enters a Java source statement and presses the ENTER key, the statement is immediately executed. If an expression is entered, it is evaluated and its value is displayed. This means that while at a breakpoint, the user can interact with any of the variables in the debug tab or any variables on the workbench. In addition, if the user creates an instance of a class via the interactions tab (e.g., LinkedList list = new LinkedList();), list is shown on the workbench. A viewer can be opened in the usual way by dragging the reference from the debug tab or workbench. Once opened, the viewer is updated as appropriate when statements or expressions are entered in the interactions tab. In the following sections, we discuss related work, and then we provide an extended linked list example to illustrate the integration of the debugger, workbench, and text-based interactions with the viewers for data structures. This is followed by examples of other common data structures generated by jGRASP. A brief summary of the integration effort is presented, including future integration tasks, and then we close with concluding remarks.

2 Related Work

Both the method and degree of user interaction with software visualizations have been shown to be primary contributors to effectiveness. Research indicates that passive modes of interaction with visualizations, e.g., only watching an animation of an algorithms behavior, are not as effective as more active engagement strategies, e.g., having the user manipulate elements of the visualization or respond to prompts [12,7]. The context in which the visualization appears has also been shown to play a vital role in effectiveness [5]. These issues are now widely seen as fundamental to