Available online at www.sciencedirect.com

ScienceDirect

Electronic Notes in Theoretical Computer Science 224 (2009) 41–46

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Dynamic Evaluation Tree for Presenting Expression Evaluations Visually

Essi Lahtinen[1,2]

Department of Software Systems
Tampere University of Technology
Tampere, Finland

Tuukka Ahoniemi[3]

Department of Software Systems
Tampere University of Technology
Tampere, Finland

Abstract

Novice programmers have difficulties with their visual attention strategies when following program visualizations. This article presents work in progress on improving the user interfaces of visualization tools to support students in the visual attention problems. We introduce a user interface solution called the dynamic evaluation tree. The basic idea is to reduce the amount of separate windows of the user interface and thus make it possible to concentrate the visual attention more in one part of the screen.

Keywords: Novice programmers, Program visualization tools, Visual attention strategy, Dynamic evaluation tree.

1 Introduction

The user interfaces (UIs) of visualization tools are often build with a similar structure. Many tools seem to have the same components in their UI and similar locations for them. We feel that this is partly because the tools are offering multiple different perspectives for the example and no specific design principles are applied for the UI design. Components are in their places just because they always used to be.

However, the effectiveness of a visualization tool in its pedagogical point of view may suffer from the use of multiple components and their placement in the screen. This article references the results of an eye-tracking study by Bednarik [2]. Based on

[1] Thank you for Nokia Foundation for partly funding this work.

[2] Email: essi.lahtinen@tut.fi

[3] Email: tuukka.ahoniemi@tut.fi

1571-0661/© 2008 Elsevier B.V. Open access under CC BY-NC-ND license.

doi:10.1016/j.entcs.2008.12.047

this, we suggest a new way of integrating some of the UI components and improving the user's target of visual attention.

2 A Typical Layout of a Visualization Tool User Interface

A typical visualization tool presents multiple different kinds of actions in turns and parallel during the execution of a program or algorithm. The different kinds of actions can be for instance:

- Control reaches a new statement in the program code or algorithm.
- The values stored in the memory of the computer are referenced or changed.
- The values of expressions are evaluated.
- The program prints output and reads input.

The layout of visualization tools' UI usually presents different kinds of actions in different windows. Different tools have different names for the windows. We list some possibilities:

- *Code window:* Shows the program code or the algorithm that is executed. It typically illustrates the execution by highlighting the line of code or algorithm. It can also be named the algorithm window.
- *Memory window*: Performs most of the visual effects by drawing pictures of the variables and data structures and highlighting parts of the pictures. In the UI of Jeliot [6], this window is named *the theater*.
- *Evaluation window:* This window is activated whenever the code window executes an expression. The values of the operands, the operator, and the value of the whole expression are shown here. An example is marked with a red circle in Figure 1.
- *Console window:* Prints the input and reads the output of the program.

In addition to the most usual windows mentioned above, there can be other possibilities like *the annotation window* that explains the run of the program in writing [9]. Also the visualization tools that allow user interaction, often have a window for the control buttons.

Depending on the focus of the visualization, it is possible that some of the windows are not necessary and are thus absent. For example, algorithm visualization tools might not need the evaluation window at all since they present the algorithm on a higher abstraction level than individual expressions. Examples of tools that do not need an evaluation window are presented by Malmi et al. [5] and Naps et al. [7]. Sometimes one window of the visualization tool contains more than one kind of actions. For instance, the theater in Jeliot 3 [6] actually includes both the memory window and the execution window.

Download English Version:

https://daneshyari.com/en/article/421935

Download Persian Version:

https://daneshyari.com/article/421935

Daneshyari.com