Available online at www.sciencedirect.com
ScienceDirect
Journal of Visual Languages and Computing
18 (2007) 99–125

Journal of
Visual Languages
& Computing

www.elsevier.com/locate/jvlc

Visual programming with analogical representations: Inspirations from a semiotic analysis of comics

Mikael Kindborg*, Kevin McGee

Department of Computer and Information Science, Linköping University, SE-58183 Linköping, Sweden

Abstract

Analogical representations based on pictures of domain objects can be used in visual programming to provide a close mapping between the program and the resulting runtime display, which can make programming easier for children and other users. The use of graphical rewrite rules with before and after pictures is an example of this approach. Graphical rewrite rules have some similarities with comics strips, which also use picture sequences of graphical objects to describe dynamics in a static form. However, the visual language of comics is not used to its full potential in visual programming. We discuss how a semiotic analysis of comics can be used to address some of the limitations of graphical rewrite rules. We use a visual programming system we have designed to illustrate that comic strips can express more general computations and be more intuitive and flexible than traditional graphical rewrites. Our conclusion is that the visual language of comics has a strong potential for increasing the expressiveness and flexibility of visual programming with analogical representations of domain objects, while maintaining a direct mapping between the program representation and the runtime representation.

Keywords: Visual programming; Comics; Graphical rewrite rules; Semiotics; Children

1. Introduction

Making programming easier, for both children and adults, is an important motivation for the exploration of new visual programming methods. Our focus in this paper is on

*Corresponding author.
E-mail addresses: mikki@ida.liu.se (M. Kindborg), kevmc@ida.liu.se (K. McGee).
URL: http://www.ida.liu.se/~mikki/comics/.

1045-926X/$ - see front matter © 2007 Elsevier Ltd. All rights reserved.
doi:10.1016/j.jvlc.2007.01.002

event-based visual programming with analogical representations of graphical domain objects. This is an approach that aims at making programming more accessible by using a program representation that looks similar to the result that is seen when the program is running.

We will start with a brief survey of work that has been done to make programming easier, especially by making it more visual. Next, we identify several important limitations of graphical rewrite rules, which is an existing approach to visual programming with analogical representations. We then introduce the visual language of comics, and discuss how the semiotics of comics could be used to make this kind of visual programming more flexible and expressive. We use a system called ComiKit that we have developed to illustrate how the issues with graphical rewrite rules could be addressed by using comic-book signs. Finally, we discuss design issues and future development of comic strip programming systems.

2. Survey of related work

Considerable research has been directed towards lowering the barriers to programming for novice programmers (for a recent survey, see Kelleher and Pausch [1]). Techniques include simplifying the syntax (e.g., Logo [2] and Hands [3]), using a ''graphical'' syntax (e.g., ''tiles'' in Squeak eToys [4]), programming by example and by demonstration [5,6], tangible programming (e.g., using physical bricks and cards to program simulations and games [7]), programming with high-level behaviours [8], visual programming with diagrams (e.g., Prograph [9]), and other visual programming techniques, for example, iconic and form-based languages [10–14]. Complete visualisations that use the same visual formalism for both the static program specification and the animated execution visualisation have been explored in Pictorial Janus [15].

One approach that has been studied in the field of visual programming is the use of analogical representations [16], that is, representations that have a structural similarity to and ''look like'' what they represent. One example is a map, which has a visual structure that mirrors what it represents. The idea in visual programming is to make the ''source code'' of the program analogous (''look similar'') to the resulting runtime result. A strong visual similarity and a direct mapping between the program representation and the runtime representation could potentially reduce the mental transformations needed to create and to understand a program, thus making programming more accessible.

Graphical rewrite rules, also called visual before–after rules, are an example of a program representation that features pictures of domain objects. A rule contains the same graphical representations of domain objects that can be seen on the display screen when the program is executed, and therefore the visual appearance of a rule can be claimed to look similar to the runtime representation.

Examples of systems that use graphical rewrite rules are Stagecast Creator [16] (originally called KidSim [17]), AgentSheets [18–23], ChemTrains [24], BITPICT [25], Cartoonist [26], and Visuit [27]. Other examples of visual programming systems that represent programs with panel sequences are Pursuit [28], C-SPRL [29], and Icicle [30]. PLAY [31] is a system for children that uses comic-strip-like sequences of panels to represent animations. Each panel is composed of an ''iconic sentence'' that consists of a character, an action, and a modifier like a direction specification.