Logo as a Scripting Language in a Component Architecture for Exploratory Software

Conference Paper · August 1999

CITATIONS
0

READS
11

1 author:

George Birbilis
University of Patras
19 PUBLICATIONS **23** CITATIONS

SEE PROFILE

Logo as a scripting language

in a component architecture for exploratory software.

G. Birbilis

Computer Technology Institute

The demonstration is about the potential of Logo as a scripting language in a component architecture for exploratory software. Logo is reincarnated as an autonomous software entity which when embedded in a component-oriented environment takes the role of the central commander, able to manipulate the other components' behavior through suitable Logo procedures and primitives.

In effect this can be used as a new paradigm for creating exploratory microworlds, by providing a mechanism that allows Logo and the formalization that goes with it, to be directly embeddable in traditionally "alien" software settings. To do that, Logo is given a new face and a new role: in a component-oriented world Logo can be thought of as a component itself, at the same conceptual level as any other component, with the special characteristic that it can directly affect other components' behavior through common Logo programming.

A Map component, for example, could zoom-in to a specified area by a specified factor through suitable Logo commands that are supported by the Map component. The benefits are twofold: the component-oriented approach inherits Logo's expressive power, and Logo is reincarnated as a core tool for formal expression in modern software environments. Zooming-in on the map component, for instance, can be done either visually when the focus is strictly on the effect of the action or by means of symbolic descriptions which provide accuracy, can be used as tools and as objects and can have parameters providing them with behavioral properties.

In this architecture, each component carries with it a set of parameters which determine its functionality vis a vis the variant and invariant features. These parameters are expressed as a set of Logo primitives which are included in the Logo code as soon as the component is inserted in the environment. The language can also be used to join up components in a static or dynamic way.

An interesting research issue is the ways in which the definition of the component parameters is influenced by the fact that they will operate through the Logo language.