XIT – A Multi-Layered Tool for User Interface Design

Jürgen Herczeg, Hubertus Hohl, and Thomas Schwab*
Research Group DRUID

Department of Computer Science, University of Stuttgart
Breitwiesenstr. 20/22, D-7000 Stuttgart 80, Federal Republic of Germany

*Author's new address: Robert Bosch GmbH, Abt. ZWI, Postfach 10 60 50, D-7000 Stuttgart 10

Keywords: User Interface Toolkits, Graphical User Interfaces, Construction Kits, Visual Programming, Rapid Prototyping

Abstract

XIT is a tool for building user interfaces for the X Window System. Different system layers are provided corresponding to different abstraction levels on which user interfaces may be constructed by user interface programmers, application programmers, or end users. The system includes toolkits with object-oriented programming interfaces based on Common Lisp and CLOS as well as graphics-oriented, interactive tools. This paper describes the different components of XIT and how they are employed to design user interfaces.

1. Introduction

Today, building user interfaces based on window systems is most efficiently performed with *user interface toolkits*. They consist of a set of common building blocks, such as buttons or menus, also called *widgets*, which may provide a certain "look and feel" to end users. Many examples of them have been implemented for the X Window System [7], mostly based on C [5]. All too often these toolkits do not provide appropriate means to define new building blocks, to modify existing building blocks, or even to build and modify user interfaces at run time. Especially the latter aspect can play an important part in the implementation of user interfaces, a design process claiming more and more resources in the development of complex applications.

XIT (X User Interface Toolkit) is an object-oriented tool for building user interfaces for the X Window System implemented in Common Lisp and the Common Lisp Object System (CLOS) [9]. It is based on CLX [8] and CLUE [4], which still are too low-level programming interfaces. XIT has been designed following the concepts of the USIT system [1], developed at the University of Stuttgart, and makes them available in a more standardized software environment. Furthermore, its functionality has been extended towards an integrated *user interface design environment*, which on different system layers provides user interface toolkits and graphics-oriented, interactive tools.

2. The System Layers of XIT

XIT provides tools on four different system layers:

- A *Low-Level Toolkit* for building user interface objects from basic building blocks.
- A *High-Level Toolkit* providing common user interface building blocks.
- A *Construction Kit* to build user interfaces interactively out of building blocks from the toolkits.
- A *Metasystem* to modify user interfaces, built by the toolkits or by the construction kit, at run time.

The system layers correspond to different abstraction levels on which user interfaces may be built and modified. Accordingly, their tools are designed for different types of users. The low-level and high-level toolkits provide convenient, object-oriented programming interfaces. They are generally used by user interface programmers with good programming skills. The construction kit and the metasystem are interactive tools with graphical user interfaces. They are designed for users with less programming skills or even end users. In the following, we will give a more detailed description of these system components and their use.

2.1. The Low-Level Toolkit

In XIT, each user interface object, in the following called *interaction object*, is implemented by *interaction nodes* organized in a *part-of* hierarchy. Leaf nodes determine the visual contents of the interaction object. Inner nodes serve as "containers" for their parts and determine the overall shape and behavior of the interaction object.

The *low-level toolkit* of XIT provides a set of basic visual objects, such as text and bitmap elements as well as graphical elements like lines, arrows, or rectangles. They form the leaf nodes of the part-of hierarchy. Furthermore, the low-level toolkit defines mechanisms to combine these basic objects to compound interaction objects, which may again be combined to more complex objects, and so forth. For example, a text menu may be composed of several text elements; the menu itself may be part of a property sheet. *Layouting objects* define how the parts of an interaction node are arranged, e.g., one below the other in case of the text menu example. Each interaction node may be connected to an application object, which is represented or may be manipulated by the associated object on the user interface. A *reactivity table* for each interaction node describes the behavior of the node by mapping user interface events, e.g., keyboard entries or mouse button clicks, onto actions of the underlying application.

Thus, each interaction node is basically characterized by the following attributes: *structure* (parts and part-of), *geometry* (position, size, layout, etc.), *presentation* (visual contents, e.g., text or bitmap), *reactivity*, and *application connection*. Due to the modular and homogeneous definition of interaction nodes, it is quite easy to build even complex interaction objects from simpler ones.

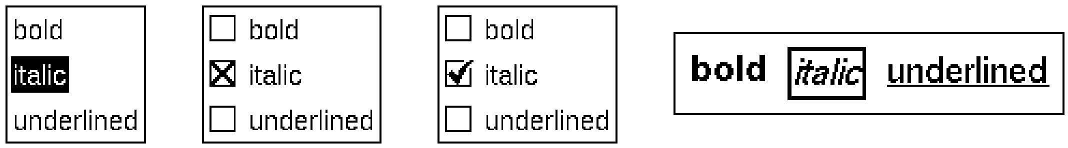
The low-level toolkit is generally used by experienced user interface programmers to implement system modifications and extensions, e.g., to define new interaction objects, not supplied by one of the higher level tools of XIT.

2.2. The High-Level Toolkit

The *high-level toolkit* consists of a set of predefined, commonly used interaction objects, e.g., icons, buttons, menus (including single-choice and multiple-choice menus), switches, sliders, and property sheets. Each of them provides various parameters to be adjusted for a particular application, for example, to show up a special "look and feel". Figure 1, for example, shows how a multiple-choice menu containing three options to change text style may be created with XIT. Four different examples are presented how the menu may look, depending on the setting of some of its attributes (font, layout, highlighting of selection).

Furthermore, the high-level toolkit implements tools like window margins, titles, scrollbars, popup windows, shrinkable windows, as well as utilities like a mouse documentation line. They may be combined with either predefined or user-built objects. Figure 4 gives an impression of the extensive facilities of the toolkit.

This system layer provides most of the building blocks commonly used in graphical user interfaces. It has been designed for application programmers with good programming skills but not necessarily deep knowledge of the lower levels of XIT.




```
(make-window 'multiple-choice-text-menu
             :view-of a_text_field
             :reactivity '((:part-event
                            (call :view-of change-text-style *part-value*)))
             :selection '(:italic)
             :parts '((:text "bold"
                       :view-of :bold)
                      (:text "italic"
                       :view-of :italic)
                      (:text "underlined"
                       :view-of :underlined)))
```

Figure 1: Creation of a Multiple-Choice Menu

2.3. The User Interface Construction Kit

The *user interface construction kit* is a tool for interactively building user interfaces by means of *visual programming*. It allows to select graphically presented building blocks from the toolkits and compose them to complex objects of the desired user interface. By means of the metasystem described below, all attributes of the interaction objects involved may be inspected and modified. With the construction kit, a user interface may be built up independently from the underlying application. At any time, however, it may be plugged into a running application to explore its final functionality. When the constructed user interface shows up the desired behavior, program code may be generated automatically.

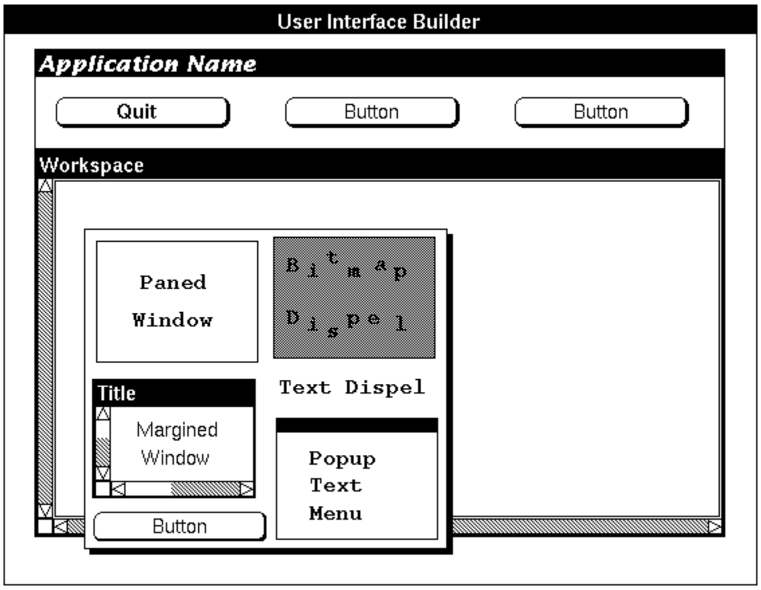


Figure 2: The User Interface Construction Kit

Figure 2 shows the construction kit building an application window consisting of a title bar, three buttons, and a scrollable workspace. A menu containing building blocks to be included overlaps the application window.

The *user interface construction kit* is especially useful either for *rapid prototyping* or for user interface designers with less programming skills or only few knowledge of the toolkits. To reduce complexity, however, it does not provide their full functionality.

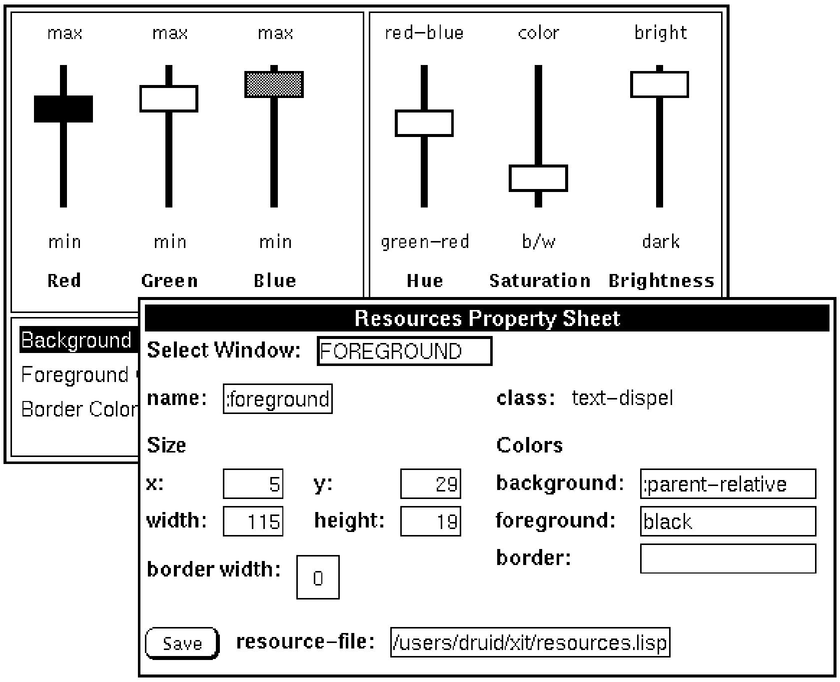


Figure 3: The Metasystem

2.4. The Metasystem

The *metasystem* is an important tool for the interactive design of user interfaces. It may be used to inspect and modify attributes of interaction objects in a running application via menus and property sheets. This may be performed by the user interface programmer, for example, as a supplement to the construction kit, or even by the end user to adapt the appearance and functionality of the user interface of an application to special needs, without knowledge of the system internals of XIT. For a set of *window resource attributes*, such as fonts and colors, these modifications may be saved on external files and reloaded, whenever the application in question is used again. Different users may thus keep different versions of the user interface for the same application. Figure 3 shows the metasystem of XIT changing the background color of a selected text element by manipulating sliders.

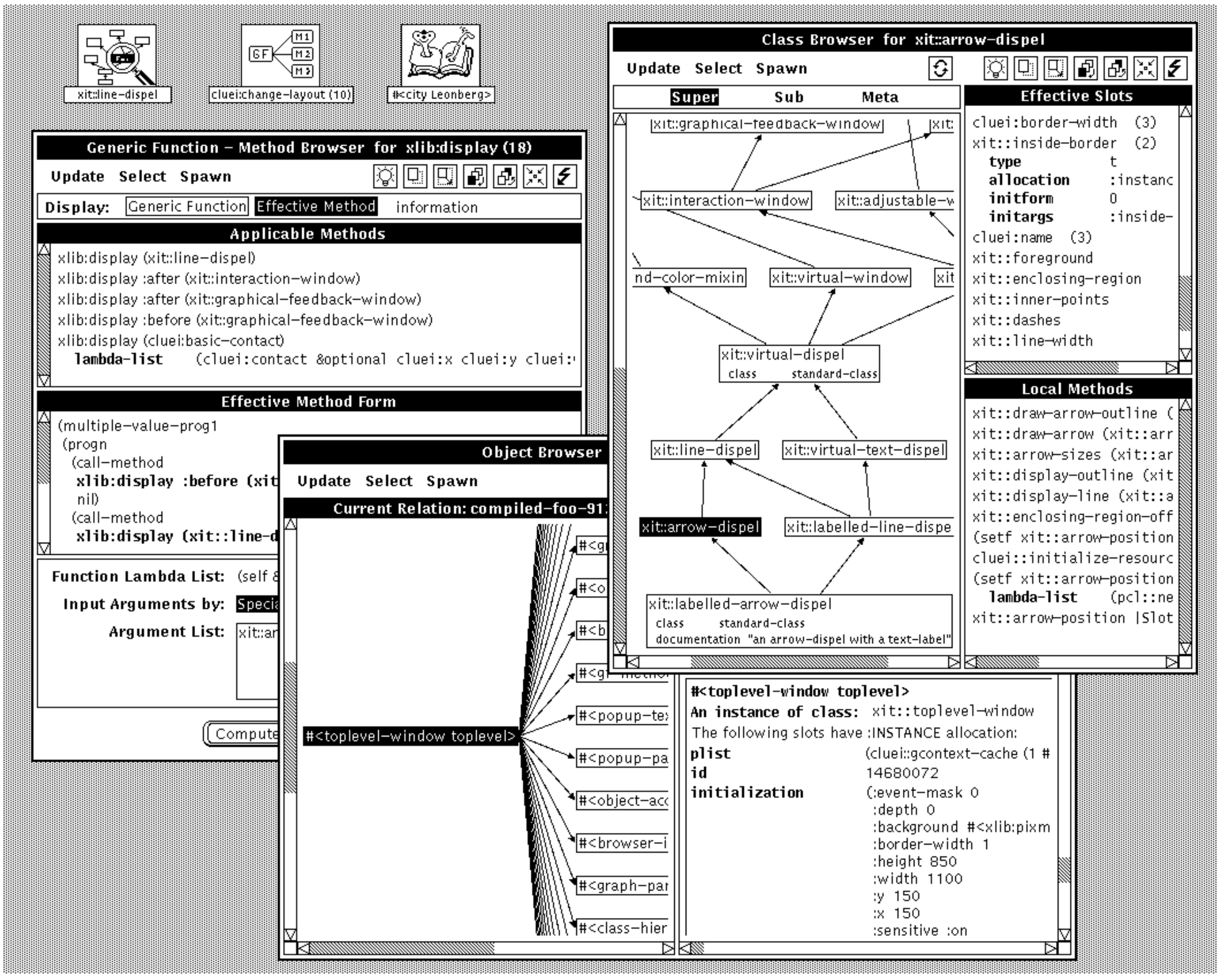


Figure 4: A Browsing Tool for Object Structures

3. Applications and Future Work

Over the last two years, the low-level and high-level toolkits of XIT have been successfully used by students and within several research projects at the University of Stuttgart to implement user interfaces for systems in different application domains, e.g., a query component for an object-oriented database in a hypermedia application [3], a system for the design of CNC machines [6], and a graphics-oriented browsing tool for object structures, including several CLOS browsers (cf. figure 4) [2]. For further development of XIT and to explore how visual programming tools may affect the design of user interfaces, we will encourage and support the use of the graphics-oriented, interactive tools, which have been built recently and are still subject to modification and extension. Especially tools for the interactive construction and design of user interfaces can effectively support programmers to implement graphical user interfaces for complex applications.

Acknowledgments

We would like to thank Michael Herczeg, who by implementing the USIT system did much groundwork for the implementation of XIT, Matthias Ressel, who contributed to the development of the system, and Michael Wichert, who implemented major parts of the construction kit and the metasystem. Special thanks to Rul Gunzenhäuser for making this work possible.

References

[1] Michael Herczeg. USIT: A Toolkit For User Interface Toolkits. In *Proceedings of Third International Conference on Human-Computer Interaction*, Boston, Massachusetts, September 1989.

[2] Hubertus Hohl. Ein graphik-orientiertes Werkzeug zum Browsing in Objektstrukturen. Diplomarbeit Nr. 727, Institut für Informatik, Universität Stuttgart, 1990.

[3] K.-H. Jerke et al. Combining Hypermedia Browsing with Formal Queries. In *Proceedings of INTERACT'90, IFIP Conference on Human-Computer Interaction*, 1990.

[4] Kerry Kimbrough and LaMott Oren. *Common Lisp User Interface Environment.* Texas Instruments Incorporated, Dallas, TX, July 1990.

[5] SUN Microsystems. *OPEN LOOK Intrinsics Toolkit Widget Set Programmer's Guide*, June 1990.

[6] Alexander Negele and Christian Rathke. Konex+ — An Interactive Design Expert. In H.-J. Bullinger, editor, *Proceedings of the 4th International Conference on Human-Computer Interaction, Stuttgart*, September 1991.

[7] R.W. Scheifler and J. Gettys. The X Window System. *ACM Transactions on Graphics*, 5(2):79–109, April 1986.

[8] Robert W. Sheifler and LaMott Oren. *CLX - Common LISP X Interface.* Texas Instruments Incorporated, Dallas, TX, 1988.

[9] Guy L. Steele Jr. *Common LISP: The Language.* Digital Press, Digital Equipment Corporation, second edition, 1990.