Intelligent User Interface for Computer Aided Architectural Design

Jaroslaw Szewczyk

Keywords
User interface, design environment

Interface development: more intelligence or more... menu?

The first versions of AutoCAD enabled an architect to operate all of the most frequently used commands through the use of mouse and side menu. The AutoCAD 2.6 side menu offered 16 main items and 157 commands of the submenu. Total mastery of approx. 200 commands (including a dozen or so of instructions excluded from the menu and some of system variables) was equal to a very good proficiency in the program.

AutoCAD R9 (the successor of version 2.6), which entered the market in 1987, had an additional pull-down menu. This menu contained 47 commands set in 7 groups. Since then, every next version of AutoCAD offered more and more enlarged menu structures.

CAD programs which work in Windows environment, took from it many new interface elements, such as picture menu, context menu and windows with various command options. AutoCAD R14 was a typical example.

AutoCAD R14 offers the following: 301 system variables, 291 main commands in pull-down, side and icon menu and text commands from the keyboard. The commands involve subcommands additionally, therefore, the given number of commands is to be multiplied. AutoCAD R14 *Edit Command Aliases* editor contains 681 main commands (including system variables, excluding subcommands); most of them can be used in many ways. AutoCAD Help Index includes approximately 14500 topics!

Summing up: development of CAD software in 90-ties was accompanied by uncontrolled process of menu structure development and considerable increase of commands number. This "interface explosion" caused many problems [4].

Interface problems in CAD

Command level – two thousands commands on fifteen inches?

From the user viewpoint, there are following main problems:

- Small **visible surface** of monitor and its **low resolution**. 15-inch monitors had resolution of 640x480 pixels i.e. 72ppi. How to place 600 commands on a small foggy surface? (Just to compare: good quality pictures in magazines

have resolution of 1200 dpi. This is 278 times better!)

- **Size of a screen**. The bigger the screen – the easier to design the properly big and readable set of tools, but moving among the tools and the working space takes more and more time. A user concentrates on his workspace (on a design), then separates from it, reads the information projected by the system, and finally finds the right palette with the right tool from among the many others. Tension between GUI (2D) and working space (3D) distracts the attention and requires independent controlling of both these elements. On the other hand, maximum working space is needed.
- Interface is **overloaded with commands**. Evolution of traditional interface created a WIMP (Windows-Icon-Menu-Pointer) model of communication. This model is a base of majority of interfaces used. It is a subtle model but not the one that would meet the needs of many complicated applications: especially CAD programs.
- Interface is two-dimensional. It is **flat**. Palettes and strips with icons, main text menu, context menus and picture menus form layered interface. Layered interface structures approximates interface to the idea of 3D, but the third dimension is limited. Placing several thousands of options of 600 AutoCAD commands in any flat structure in a way to make it functional is not possible.

The simplest solution is creating different working environments, **virtual workspaces**, with a limited set of commands. The criterion is frequency of using a command while working. This criterion is taken into account while separating modules of menu for "beginners" and "advanced" users (like in ArchiCAD). Better model of shaping interface is **context – oriented interface**. The idea is simple. Only the commands connected with the current work are available. Context menu is particularly useful in object-oriented applications, because activities in these programs can be anticipated.

Data level – the problem of a needle in a haystack

CAD systems became tools for modification of complex objects that contain a lot of data. From the viewpoint of a CAD user, an architectural project is a big, complex, graphic-text database. Therefore a good interface should enable easy access to the whole data structure. Interface should help user with visualisation of data and intuitive modifications of information.

When a user deals with numerous data and when many multileveled connections appear between the elements of information, operations on it often exceed his perceptive capacity. Users need information extracted from the project to be clearly visualised. Otherwise, work on a big, complex model becomes impossible. That is why the interface problem is not only the problem of a menu and quantity of accessible commands, but it is also a data problem. It is important to hide surplus data, which are not needed in current work, but which are an integral part of a project as a whole.

CAD systems based on the object technology are in fact based on object database technology. However, object database technology has some shortcomings: it does not facilitate differentiation of the degree of data abstraction. An architect, a constructor and a surveyor require different descriptions of the same data. Everybody perceives the data of the project from his own viewpoint and needs some elements projected clearly on the screen, some switched off, and other shown more detailed. Every user needs data structure organised in a different way. Moreover everybody needs to see a project in a various scales. So members of a project team need an architectural project to be an open collection of data, with which one can work on a various grades of abstraction. CAD system is to enable efficient management of contents of data structures and visualise each of them on a various stages of detail regardless of others structures [3].

Summing up: Interface should help user with cre-

ating, modifying, separating and showing data of various types, easy passing to various degrees of abstraction in a project, preserving coherence of processed information and preserving connections between elements of project.

The solution: This group of problems (especially surplus data problem) can be solved by use of **drawing filters,** which appear in some object-oriented and parametric CAD software.

Context level – a needle in two haystacks...

In a graphical user interface, both the data and the possible operations on the data are visible to the user. In CAD systems, both the number of data and the number of accessible commands exceed a user perceptive power. Therefore an interface should enable:

- removing the surplus data from a project (data structures filtering)
- removing the surplus commands (commands structures filtering: only the most useful commands are visible at the moment)
- synchronisation of user and computer actions (interface tool and CAD engine should be integrated)

The solution is interface with **intelligent agent programs** [2]. Now, some complex applications contain autonomous or semiautonomous agents. An interface agent is a program that affects the objects in a direct manipulation interface with no explicit instruction from the user. There are two directions of agent development: interface agents and autonomic agents. Interface agents help a user with interactive use of interface. Autonomic agents act regardless of a user. Both kinds of agents can integrate a CAD engine with an interface.

Promises: Intelligent CAD environments

Traditional model of interface is a conversational one. Work with complex CAD models require totally different interface, the one that would help user with formulating of complex tasks, in a way understandable by a computer. Interface should analyse the task and suggest means to its realisation – it should do the job previously done by a user. In this way intelligent interface shall minimise number of interactions between a user and a computer, which are necessary to build up a model.

Autonomous agents of interface can change its "conversational" model into a more effective one. Agents adapt the interface to the user's preferences. They can learn user's way of thinking. As a result of agent's actions**, the consecutive building of the optimal, intelligent work environment takes place.**

Reality: Educational and mental barriers

CAD programs will not have the intelligent, interactive interface unless users see such need. Cost is the barrier (number of code lines of such interface may be up to 80% of the whole application code! [1]).

Users of CAD programs can be divided into two groups: novices and those who have got already used to their ways of work. Experienced users do not want drastic changes in their way of design. Novices, especially students who are not burdened by redundant habits may be interested in working with really intelligent tools. Students may break the established patterns of thinking. The university centres should play great role in this process.

References

1 Gray P., Took R. (eds), **Building interactive systems: architectures and tools**. Springer, London, 1992

2 Lieberman, Henry. **Autonomous Interface Agents**. In: CHI 97 Electronic Publications: Papers, http://www.acm.org/sigchi/chi97/proceedings/paper/hl.htm

3 Naja, H. **Multiview databases for building modelling**. In: *Automation in Construction* No5 (June) 1999 pp.567-579

4 Szewczyk, Jaroslaw. **Interface problem in CAD systems**. In proceedings of the 5th International Conference on Computer in Architectural Design, TU Bialystok, April 1998

Jaroslaw Szewczyk
Technical University of Bialystok
jarsz@cksr.ac.bialystok.pl