Multi User Interface Problems in Current CAD Systems

Jaroslaw Szewczyk and Adam Jakimowicz  
Faculty of Architecture, Bialystok University of Technology  
ul. Krakowska 9, 15-875 Bialystok, PL

Abstract

The general problem of software and hardware interface has been discussed since the first personal computers had appeared on the market. Interface solutions, proposed by the CAD software producers themselves, have been deeply criticised. In fact, learning how to use the software means how to operate its interface. In the process of software developments, interfaces became 'second languages', which are to be learnt by the user.  
This is a real problem especially for the beginner. The problem is even more complicated in case of interfaces for many users. They should help in the processes of computer networks based cooperative design. In this paper we try to juxtapose the main features of user interfaces in CAD, especially ones for collaborative work. The most important problems were detected, and conclusions were made with an aim to research interfaces adequate for architect’s work in the future.

1. An interface in CAD

While describing interface we mean user interface in CAD, i.e. part of CAD software which is responsible for interaction between one or more users (designers) and CAD software on one or more computational machines (workstations, servers, laptops or any other kind of electronic machines using by designers in their work).

Rapid development of both hardware and engineering software helped designers in their work, giving them a great variety of powerful applications. Most of such applications offer a lot of tools (commands), and users often have problems with learning them and working with these tools efficiently. There are some problems concerning “common” interface (for both single- and multi-user work), and some ones which are very important while collaborative work, especially in multidisciplinary engineering environment.
Common interface problems are:

1.1. **Overloaded interface.** Evolution of traditional interface led to creation a WIMP (Windows-Icon-Menu-Pointer) model of communication between users and computer. 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 advanced applications: especially CAD programs, because their commands' number is too large to use these applications efficiently. Thus interfaces are overloaded with commands. For example: a quantity of key-ins in MicroStation 95 (with Master Piece added) was 1900 (This number was counted from MicroStation Key-In Browser database). AutoCAD R14 offered nearly 700 commands (including system variables) with thousands of options. A lot of commands in CAD software (especially the most frequently used ones) have a variety of options, so total number of commands and options has to be multiplied.

1.2. **Small visible area of computer screen and its low resolution.** An architect has very specific needs for his working environment. His workplace should help him with his perceptive abilities. An architect would not be helped with his work unless he can see his design project shown on a wide, large surface - the maximum one that he can perceive. And the possibly clear one, too. Good quality pictures in magazines have resolution of 1200 dpi (1200x1200 = 1 440 000 dots per square inch). 15-inch monitors had resolution of 640x480 pixels i.e. 72ppi. 72x72 = 5184 points per square inch - this is 278 times worse! How to place some hundreds of tools with thousands of options on a small foggy surface?

![Figure 1: It is possible even to hide a workspace opening some software toolbars.](image)
1.3. **Workspace-versus-interface-versus-user battle of screen.**

Computer screen can be too small and too large at the same time. The bigger the screen - the easier is to design and locate there the properly big and readable set of tools (see the above paragraph), but moving among the tools and the working space takes more time. A user concentrates on his workspace (on a design), then separates from it, reads the information shown by the system's interface, and finally finds the right palette with the correct tool from among the many others. This cycle distracts the attention and requires independent controlling of both GUI space and working space, but GUI and working space need quite different user actions. They have really no common features: GUI is two-dimensional and working space is in most cases three-dimensional. When user works with a larger desktop, he has also a larger problem with tension between GUI and the workspace. It is helpful for the user to see wide screen with his design detailed, but he is able to concentrate his attention on the one point of the screen actually. Summing up: the possibly big working space is needed to see the design clearly, and a big desktop also helps user with interface managing, but - because of the mentioned reasons - when the screen is large, user can have difficulties with efficient controlling his project space mixed with interface's elements. *(Less is more...)*. Authors' suggestion is to integrate project space with interface elements to avoid distraction of user's attention and enabling him to concentrate on a design mainly. Some 3D modelling software packages have already propose such solutions (e.g. Caligari TrueSpace).

![Figure 2: TrueSpace with a 3d toolbox, located within a workspace](image-url)
1.4. Flat interface's space.
Interfaces are mostly two-dimensional: flat. Palettes and strips with icons, main text menu, context menus and picture menus form a layered interface. The layered interface structure approximates interface to the idea of 3D, but the third dimension is limited. Placing several thousands of options of 700 AutoCAD commands in any flat structure in a way to make it functional is not possible. Isn't it another reason for integrating interface with 3D working space rather than forming two different self-existing structures?

1.5. Separated design space.
Design space is separated from the CAD system interface and limited by two-dimensional direct draw surface. Linking design space with extended 3D interface elements (like spatial context menu) will make the interface more intuitive.

1.6. Lack of coherence of design environments: between different CAD systems. It would be needed to have a common data structure when using some different software packages. A user needs known-to-him hierarchy of layers, elements' styles, objects etc.

1.7. (Un-)Clarity of project data structures.
Software should optimise data structures and organise user's virtual working environment in the way to make it the most useful. This task is very important when dealing with large, complex design projects. In these cases controlling and managing layers' and other settings' structures exceeds the user's power. User can know that none but he must manage information structure and make it consequent and clear, but in fact interface can play a great role with managing data structures.

Figure 3: Different types of data displayed paralelly
In the multimedia CAD systems, it is not so important just to receive information, but to interact with the data. So the problem how to reach and work with the data becomes crucial. Documents or other single design elements can be grouped and being relocated between different projects, being accessible partially or totally, and organised according to different hierarchies. A designer (looking at 15 or 17 inches screen), should be able to recognise and work with structures of the objects, the project consists of. Different design objects can be defined around a single element. For example a wall can be defined and read as a drawing element, as a definition of complex entity, database, file, user, group, etc. At the same time all objects are interrelated and structured. So, in architectural design, the software should make the conceptual work on a quite high level of abstraction possible, to help acquire a selective view of the chosen aspect of the computer elaborated design. During the design process one cannot make many decisions at the same time. From the point of view of the CAD systems user, architectural project is fact a large, text - graphic database, where the software serves to visualise structure the data, and at the same time should empower creative thinking. However, the more types of data is being produced, the more complicated its structure becomes, so it also becomes harder to make a readable record of this structure in the project. So CAD systems are the tools to modify the complex objects, which in fact are carriers of large amounts of data. The task of the software interface is to enable a user an easy access to the whole structure of data, its visualisation, and intuitive data processing. Therefore interface problem means not only the menu problem, nor the problem of the number of available commands. The essence of the problem is the possibility to visualise the data needed and extracted from the project at a given moment, and at the same time to hide all the extra information, which is still is a part of the project as a whole. The problem becomes even more complex, when we understand, that it indispensably concerns design methodology. Contemporary GUIs unconsciously want to answer all methodologies, but they do not do so at all. Actually they strongly impose the undefined design methods – which are just derived from the software operational syntax. So does it have a sense to produce different interfaces for different design methodologies? To what extent shall we relate design methods to the possibilities of the software interface?  

1.8. (Un-) Clarity of interface information structures. 
Interface structures must be consequently clear. When unimportant tools dominates on the screen, it damages intuitiveness of visual information structure. The problem can be compared with another problem of spaghetti code - such term defines (in cybernetics) application programming code which structure is very complicated and then it becomes impossible for programmer to work with this code. So the same problem is also now with GUI. Instead of ordered, ergonomic Interface, a user fights against spaghetti Interface and he in unable to deal effectively with all this tools chaos.
1.9. **No limits with interface complexity.**

Most of CAD software contain API (Application Programming Interface), i.e. built-in programming languages and script interpreters, compilers and debuggers, which can help user with modifying (what means: complicating) interface's and whole application's structure. A programmer can use them not only to change interfaces, but to modify program „engine“ and to create new applications. Any CAD software offers user more and more programming tools, such as C++, Visual C++, Visual Basic, Delphi or Power Builder. AutoCAD can be an example: AutoCAD in DOS versions had tools for menu compiling and creating macros, it had DIESEL interpreter, built-in AutoLISP translator and integrated programming environment ADS/ARX. Now AutoCAD has more programming tools than anymore: Java, Visual Basic for Applications (VBA), Visual LISP, Visual C++ (via Object ARX API), Delphi, ActiveX Automation and COM technologies (that is why numbers of items in AutoCAD R14 Help (Menu Help\Help Topics\Index) was about 14500). API enables an interface to be enlarged many times!

1.10. **Interface language:** GUI language is mostly the language of symbols and metaphors, such as key words, shortcuts and (mainly) icons. It has its grammar and a vocabulary. Both grammar and words and rules for communication must be as intuitive as possible. A user needs a common language for all known-to-him interfaces. This means that all applications used by the project team must use a common language.

1.11. **Interface metaphors:** Graphic symbols have been used in technical drafting for ages. Now graphic symbols are not only parts of drafting but they form an interface. In MicroStation SE a user can choose between over 420 pictograms (icons) representing some most often used commands set in quite different tools (access to icons database through: MicroStation SE Main Menu->Tools->Tool Boxes->New->List Tools Alphabetically). Graphic symbols and all keywords are metaphors. They include specific meanings. Metaphors can be found also in hierarchy of pieces of visual information, relations between them and in many other GUI elements. To see how novices can interpret icons' metaphors we made an experiment at Faculty of Architecture, in Bialystok: young students (after second semester) received a form with 38 icons chosen from MicroStation SE. Students were not familiar with this application. In most cases they were not familiar with any CAD program except of a modelling one, 3D Studio Max R2. They had to guess the correct command represented by each icon. We received back about 40 forms with about 1500 answers. The metaphors of icons were not correctly understood in most cases (both when icons were simple and when they were complex and difficult). Students had no idea about collaborative work, designer-to-designer communication, interdisciplinary work etc., but often think that CAD software enables user making renderings and animations only.
1.12. **A very high number of complex and different computer actions.**

Users must learn computer behaviour. New software, not CAD only, likes to behave strange. It isn’t easy to learn all the behaviours: our beta-version of Windows NT 2000 Workstation has 184 files with cursor definitions, 34 animated cursors among them. Learning computer’s behaviours needs some efforts: Windows NT 2000 has 337 files in /WINNT/Help folder (in my Windows 95 OSR2 there are only 3 cursor files and 127 menu files).

1.13. **Explorativity of interface.**

GUI must be explorative. A user should be able to work with CAD software without knowing all interface commands. He should be able to learn interface during work. For this, some applications (like ArchiCAD) have interface levels for novice users and intermediate and advanced designers. Some other CAD programs have extended help, tutorials or wizards. There are also packages, which offer many interfaces to choose. We do not always need the same commands when designing with CAD – so it is not necessary to have them all at hand all the time.
1.14. **Redundancy of the interface.**

Some CAD software (especially AutoCAD) has redundant interface, i.e. user can find the same commands in different parts of an interface. As an example, there are possible ways of input process for Zoom Dynamic command in AutoCAD R13c4:

- from the icon menu, Zoom toolbar
- from the icon menu, Main toolbar
- from the pull-down menu
- from the side menu
- from the keyboard
- from the keyboard aliases

the command can be replaced with the use of Aerial View window

the command can be assigned to a hot key (through modification of Accelerators section in the menu file)

Interface redundancy can make chaos in user's mental imagination of commands. The above trial to detect main problems with a single user interface, introduces us to even more difficult issue of the interface for many users, working in a collaborative team.

2. **Multi-user interface**

An architect deals with more complex problems while collaborative work. There are some specific problems concerning multi-user interface among them. These problems are:

2.1. **Collisions between data structures shown on the desktop:** Any virtual architectural project can be represented as a set of object data structures. Each of these structures has its own elements, objects, hierarchy etc. Some of such structures gather architectural information, some others can integrate data required by engineers, interior designers, electricians, urban planners, CAD managers and so on. Each of the participants of design work requires his own data structures, specific to his profession. Moreover, some co-workers may need showing project on a various degrees of abstraction and in different scales. That is why users need the same object shown as parts of different data structures or even as artefacts in different virtual spaces. It is possible that each user may need an object in a specific-to-user way, and everyone can require his own data structure or set of data structures. The more people are involved in design process, the more complicated data structures are, and dealing with all the information becomes nearly impossible. It is essential that the interface should manage data structures effectively, i.e. an interface has to filter redundant data and to show only the data, which is necessary in current work.

2.2. **Extremely complicated menu structures.** A good CAD software for groupwork doesn't aid users with design only, but helps all users with all their
actions, such an organising and managing project data structures, communication among users and sharing project among them, exchanging data between different applications, saving archive files and managing temporary information. CAD user must work on many areas and levels. His tools have to help him with it. He needs a large set of different tools, organising in many toolbars and he probably uses a lot of workspaces. How to create simply and clear structure of interface, when the number of tools exceeds some thousands and when number of groups of actions (such as drafting, modelling, modifying, writing, rendering, networking, interface managing, project managing, team managing, working with databases and many others) is about fifty? It seems that some kind of precise modularity of the interface would be helpful. These modules could be as follows: drawing – designing – modelling, communicating, distributing – data managing, etc.

2.3. Problems with acting in virtual collaborative design space. Not only the lack of such space is the problem, but the lack of methods of acting in such space. Architects need to be artefacts’ (models’) creators, space explorers and modifiers, actors in communication processes, owners of their individual spaces and data structures. Such actions influence the interface. Interface becomes the medium for any user actions, and the problem is whether the medium is adequate for these actions. Here again we touch strongly the problem of design methodology. The awareness of design method and goals should not be left behind, at least the interface should not influence the methodology directly.

2.4. Problems with communication between spaces. The interface unites virtual design spaces, single user interface spaces, communication spaces and user mental spaces. It must translate the languages of these spaces, i.e. ways of acting in the spaces, and show effects of all the processes in an understandable way.

2.5. Interfaces for communications. In this group of problems the main problems are: how to make communication between co-workers easier and more natural? How to communicate with another designer without leaving project - is it possible to pay attention to the project only while communicating with others? How to enable specialists to communicate with each other, although they need quite different project structures and different interfaces? Communication becomes efficient and natural, when it involves more than just one of the available senses.

2.6. Standardisation of data and interface structures: levels, layers, layer groups, stories, groups of objects, project files names, annotations formats and text formats, dimension settings, special data settings (construction, piping, electrical), objects’ definitions, command names, shortcuts and hot keys, programming languages, menu icons, cursors and many others.

2.7. Interface’s role in synchronisation of collaborative work. The interface has to enable users working in not 3D only, but in multidimensional
space, with many space levels and with a parameter of time. This parameter helps with synchronisation users' work and spaces fluently in defined time periods. Project history seems to be essential, like in Bentley's Project Bank.

2.8. **Project history (journal).** Design team needs each of the changes made by each user tracked and logged so that one can audit what has happened throughout a project. A project journal enables answering questions, such as: "Who placed this element on that design?" and: "Who has ever modified this part of design and when?" It increases total amount of information available for user, so that it can enlarge collisions between data structures shown on the desktop (especially when every design element has its own history saved).

2.9. **Change merging and reviewing conflicts.** Users need interface allowing them to modify a drawing at the same time. Parallel work of some designers on one project can make conflicts between users' changes of one design - especially in huge project, while communication is poor (for instance while working on laptops without connecting). It is essential in collaborative work to avoid collisions between project changes made by users, to solve problems with bad project versions etc. Interface’s role is to help user with reviewing project, and to visualising every user’s changes, conflicts, project versions etc.

2.10. **The necessity of designing and developing individual interfaces for the most advanced collaborative design methodologies.** Maher and Simoff have written about the variation of the virtual meeting place, which is the model of the product being designed, as a virtual space, where collaborators meet, communicate and work. This kind of collaborative synchronous work seems to be the most interesting and desired way of designing in disseminated online groups. However, we face a general lack of the efficient interface solutions for this kind of environments. Maybe the problem could be solved, as Asanowicz writes, by the intelligent integration of the voice recognition systems into the interface and the system itself. The needed menus, data, etc., could be evoked in the virtual space by natural human voice, and learning of a specific vocabulary to communicate with the system should not be a problem...

3. **Conclusion**

Now a designer, working with CAD, deals with small desktop and a great number of commands enabling him work, communication with co-workers, managing project and managing work, viewing results, organising project data structures and interface structures, and many others.

Collaborative work increases the number of interface problems significantly. Some problems appear while multidisciplinary work (huge amount of data, conflicts between different standards etc.), when attempting to synchronise group work (change merging, acting in virtual space), and there are some problems with
communication process or concerning any other aspect of multi-user engineering environment.
The interface problems can't be solved in a simple way, but defining and structuring them can help software vendors with developing newer versions of their applications. This can also help users, designers understand better the complexity and nature of digital design environments. It may also help in training young architects and students. But in our opinion the most expected chance it brings is the possibility to create new design paradigm. This paradigm appears with a fast development of computer based patterns of collaborative design networks. At the moment - there is still is mostly a technical problem with it: how to make a well working 'network machine', which for the users means 'understandable and easy interface'. However, such an interface does not seem to be possible, knowing how complex tools CAD systems became. The problem of the 'interface as a second language' can be concluded with the issue how difficult this language should be.

The work was a part of rector's grants W/WA/1/00, W/WA/4/00 for the research project realised at the Faculty of Architecture, Bialystok Technical University.

References

2. Internet: http://ox.compsoc.net/~swhite/history.html ("History of Windowing Systems")
3. Internet: http://projectwise.bentley.com/
4. Internet: http://sigart.acm.org/conferences/iui/iui98/ (IUI 1998 - International Conference on Intelligent User Interfaces)
6. Internet: http://www.acm.org/cacm/AUG96/antimac.htm
8. Internet: http://www.bel.cs.ucla.edu/~dondi/cmsi628/lectures/history/onepage.html
18. Internet: http://www.viecon.com/


• [29] Szewczyk, J., 1999 „Bariera interfejsu a projektowanie wspomagane komputerowo”. Zeszyty Naukowe Politechniki Białystockiej (Architektura, Zeszyt Nr 18), Białystok