ORIGINAL ARTICLES

Enhancing Usability by Improving Software Visualization and Navigation

Ali Sajedi, Hamidreza Afzali, Zahra Zabardast

Software Engineering Department, Lahijan Branch, Islamic Azad University, Lahijan, IRAN.
Department of Management, Roudsar and Amlash Branch, Islamic Azad University, Roudsar, IRAN

ABSTRACT

Massive amounts of information are dealt with in the current generation of software applications. Numerous commands and situations are to be issued by the user in these systems. Visualizing, finding and triggering the desired functionalities in an application are important subjects in User Interface (UI) design process. In this paper, some new aspects in managing both hidden and visible concepts of form based UIs are offered to have more friendlier and reliable UI. First, the automatic arrangement and enhancement of visual objects in the forms is proposed to make the UI customizable for each user automatically. Then, making hidden objects and commands -such as shortcuts and application agents in the switch bar window- visible is offered to reduce bewildered situations and time to trigger a command/application. Finally, the main factors of a UI are evaluated based on the offered suggestions.

Key words: Human-Computer Interaction, User Interface Design, Information Visualization, Software Engineering

Introduction

Not long time ago, computer software applications were not user-friendly enough. Users needed a lot of training and they sometimes had to learn complicated commands to be able to work with those software applications. Because of the hardware limitations of processing, there was no Graphical User Interface (GUI) in software environments as we see today and software was usually designed in text mode. Enormous advances of computer hardware in recent decades made it possible to use more processing power for software interfaces and the dream of having Graphical User Interfaces came true. This had a great effect on the market of software applications, as developers could offer software with much more different interfaces and let users to choose between a number of softwares with same abilities but different interfaces.

User Interface Design or shortly called UID is the design of computers, applications, mobile communication devices and web sites with the focus on the user’s interaction and experiences (Arroyo, E., et al, 2006; Dutoit, A.H. et al, 2005). Today softwares have much more complicated environments, but the interfaces are designed in a way that even a novice user can easily learn how to work with them. Thus, having a user-friendly interface in addition to complete functionality is an important factor to choose softwares.

Different methods of direct Human-Computer interaction are available today by the means of different input and output devices such as keyboard, mouse, joystick, light pen, barcode reader, scanner and printer. Nowadays, computers are used widely and there are a lot of embedded computer systems everywhere. User interface design is used widely in different types of projects, from home and office applications to today’s modern cars (Störrle, H., 2002).

In Graphical User Interface design, the developer has to design shapes of objects like buttons so that the user is able to interact with the graphical interface. This means dealing with drawing shapes on the graphical screen and recognizing different types of user’s interactions with special parts of the shapes (e.g., clicking and double-clicking on an item or dragging it to somewhere else in the screen). Drawing such shapes on the screen can be easily done by few lines of code, but handling user interactions with the shape is a time consuming action that may need hundreds of lines of code. Besides, this repetitive and time consuming action has to be done for a lot of shapes that is used in the application. Instead of writing hundreds of lines of codes we may use UI design packages to manage these time-consuming events. Mac App system was one of the UI design packages that reduced coding time up to 80%. Some of the most important benefits of using UI design tools may be mentioned as Increasing design speed, simplicity of changes, concentrating on essential details of program rather than controlling the interface and uniform design (Medina, P. et al, 2007). UI design packages are used widely to produce user interfaces even by web designers. A well designed user interface for a website is an important factor that has a great effect on increasing the number of visitors.

Corresponding Author: Ali Sajedi, Software Engineering Department, Lahijan Branch, Islamic Azad University, Lahijan, Iran
E-mail: Sajedi@Liau.ac.ir
The main goal of the User Interface design is to make software more user-friendly. A new method in
designing a system is to design in a way people want to work and interact with the system, instead of
forcing new defined ways. User Centered Design (UCD) is a process in which the needs, wants and
limitations of the user are considered in all stages of the design (Gaffar, A. et al, 2005). In this process the
ways users usually use an interface is required to be analyzed and the validity of the assumptions have to be
tested on actual users in the real world.

Learnability and usability of software applications is an important factor which has to be considered in
designing software applications (Sajedi, A., 2008). Effective UI design is discussed in some papers recently
(Sajedi, A. et al, 2008). Usability guidelines for a UI are also studies and defined in some research projects
(Sajedi, A. et al, 2008). The focus in this paper is on some suggestions to improving both the learnability
and usability of softwares. Some important aspects of software applications’ user interface is studies in the
next sections.

1- Important aspects of UI DESIGN:

Three important aspects of User Interface of software applications are discussed in this section.

2-1- Metaphors:

Facing familiar items and concepts in user interface of softwares has great effects on user’s mind in
memorizing different concepts and also in simplicity of use. Contexts can be infused to users by choosing
good metaphors. The Recycle Bin in Microsoft Windows is a good example which is seen in the real world
with similar application as well. Another example of metaphor is implemented in Media Player softwares.
Because of the clear resemblance between Media Player software interface and real music players, even a
novice user can easily work with it. Because of the appropriate used metaphor, it would not be hard for the
user to guess how to use the Media Player basic functions such as play, stop, forward and backward. It this
case, processes and environments may also be matched with the real world. For example, this match may be
considered in categorizing items in the forms or setting the order of related processes. Of course using
metaphor is not always useful. Sometimes it is better to use simpler functions.

Two important factors should be considered to choose a proper metaphor. As softwares are designed
for a wide range of users that are usually unknown, it is important to use metaphors that are familiar to
everyone. It is also important to spread the metaphor all through the software application. So it may be used
everywhere the similarity is found.

2-2- Help Systems:

In addition to a good environment design, what makes software more learnable is a good Help system.
It should be context sensitive and users should be offered a simple, step-by-step and complete explanation
of what they need to do. Tooltips and Help Browsers may be mentioned as examples of common help
systems.

Wizards are also useful options as they can also help users to achieve what they need, step by step. As
is seen in today’s help systems, the ability of searching words and commands in the total Help system to
find the needed information makes the Help system more useful.

2-3- Flexibility, Efficiency and Simplicity:

Usability of a software application is the most important matter for the end-users. It may be discussed
from different points of views as flexibility, efficiency and simplicity of use.

Computer’s productivity should be considered less than user’s as in some cases users may cost more
that computers. It is important to reduce the waiting time of the user as much as possible. Of course, in
organizations, the overall efficiency of the system is important, and the efficiency of the users and groups
should not be considered as the main goal. After making sure about the most possible efficiency of the
whole system, users and groups’ efficiency may then be considered.

The fundamental architecture of a system is what makes it efficient. Therefore, having efficient forms
with suitable relationships between them is desirable (Molina, P.J., H. Tratteberg, 2005).

Most common softwares are developed to be used by a wide range of users, so they should be designed
suitable for both novice and expert ones. This means to benefit from both the powerful functionality of the
software application and the ease of use.

As mentioned before, another important issue is the simplicity. In order to achieve this goal, some
simple but complete forms should be designed at first. After that, the required links between them may be
created (Molina, P.J., H. Tratteberg, 2005).
2- The Proposed Suggestions:

Some items containing concepts are embedded in windows forms and environments used by users and even the expert ones may not be aware of or do not have access to them. These concepts have to be used and offered to improve users’ accessibility. The suggestion is to visible these invisible concepts to help the user select and use the best way of doing an action. In the following sections of the paper, some of these concepts are discussed and some suggestions are offered. Besides, some suggestions are made to improve learnability and usability based on the aspects mentioned in the previous section.

3-1- Automated Form Arrangement:

During the history, one of the important wants of mankind has always been to change the world around in a way that everything may be accessed easier. In the software world, a user may deal with a large number of objects in the forms that the proper ones may be selected based on the needs, situation and goals. If the most frequently used objects can be accessed faster than other ones, this would improve the speed of the user. A simple instance of this idea can be seen in some operating systems where the most recent used applications and documents are listed in main menu of the operating system.

It would be great to use this idea in softwares and office automation systems to help employees work faster. These means offering easier access to more important objects and even hide less-important ones. The suggestion is to sort objects based on the frequency of use. Some other useable techniques are to reform, resize or change the color of objects to help users find the desired ones faster. A data mining approach is helpful in more complicated environments as they need more technical solutions to handle this matter. Besides monitoring the usage of objects, the time of use may also be logged in a database for later analyses to determine the best position arrangement for them in the form. Some heuristic consideration should be done by the software to handle this arrangement properly. The process of analyzing data and attaining results for a proper arrangement is a complicated process that needs artificial intelligence considerations and may be discussed separately in some future research.

3-2- Shortcut Keys Visualizing:

Some experienced users that want to use software environment abilities faster, prefer to use the keyboard more frequently because of the available shortcut keys which a lot of them can be found in software applications. But novice users may just know a few of them. In most software environments, two common interaction devices are used; keyboard and mouse. As switching between interaction devices is time consuming, using shortcut keys can help users work faster without changing the interaction device. Therefore, a mechanism should be developed to inform users about the available shortcut keys of the environments.

You may find a basic form of this idea in some menus where a character is underlined to inform the shortcut key of the menu item. As seen in Figure 1, a better graphical example of this is also done by a team of developers in Microsoft working on Microsoft Office. Shortcut keys of Ribbon items would be shown graphically by pressing Alt key. Of course this is not a complete implementation of the idea and ideal is to have such graphical visualization of shortcut keys all over the environment. For example after triggering a functionality item from menu, the shortcut key can be shown to help the user memorize it. The Help system should also be consisting of sections to inform users about available shortcut keys in each environment and a complete guide about them.

![Microsoft Office 2007 shortcuts.](image)

Several shortcut keys are also assigned by the operating systems to special tasks so the idea can also be considered in designing operating systems. The layered structure offered by (Sajedi, A. et al, 2008) may also be considered in this case.
Visibiling the available shortcut keys can be said to have impact on both learnability and usability of softwares applications for both novice and expert users.

3-3- Program Selection Visualization:

In recent graphical operating systems there is the ability to switch between in-memory applications by selecting them graphically. As an example, in Microsoft Windows the switch bar would appear if the user press Alt + Tab keys. This bar is used to visible an invisible conceptual relationship between the currently running applications by showing their icons in right order. The list of applications is always sorted by the activation time of them and resets every time an application activates. Showing applications with icons may cause someone to select the desired app with difficulties. Specifying the right program to switch needs a lot of attention because the list is dynamic and placements are changing. As shown in Figure 2, a new idea is used in the modern graphical environments of Windows Vista Aero. A real-time image of the running application is shown in the switch bar. This is an amazing idea, but what about using several programs with the similar interface theme? What about the computer systems with graphical limitations? They are not able to use this feature for sure.

![Vista Aero switch bar.](image)

The new idea is to focus on applications, not their order of activation and confuse users to choose the right one. Currently, even experienced users may need to switch between applications several times to select the right one, because specifying the right one is not easy especially when they are running several instances of an application.

Suppose you are doing a research and several instances of Microsoft Word documents are open. You keep switching between them to check your writing. Soon, you would forget the order of them and cannot find the right document in the first try. It just offers you same icon for all of them. But you still need to remember the order of them or you have to read the names one by one to find the right one that means holding the Alt key and pressing the Tab key frequently. Among several possible solutions, two suggestions are offered in this section that may be used as elective options.

The first suggestion is assigning a sequenced index number to each application when it is running. Consider that, this is different from the Process ID that is assigned to each new process by the operating system. This indexing is only applied to user applications and would be shown on the icon in the Switch bar as well as the Title bar. Therefore, as the index number is fixed, users can easily find documents at a glance by the index number. It may also possible to select an application window by typing its index number in a specified text box. The index number starts from one, increases by one and will be reset to one if all running tasks end. As you may see in Figure 3, index number 0 is also assigned to Show Desktop function.

![A sample proposed Switch window.](image)
The Second suggestion is grouping similar applications together and show icons in a hierarchical structure (Figure 4) using the topmost level to show the main applications and the next level to show the instances. However, the indexing may also be applied. The advantage of this suggestion is realized more when switching between similar applications. This way, similar applications may also be closed in groups.

Fig. 4: A sample categorized Switch window.

It is also possible to change the order of icons in the list (for example by running time). To have both advantages simultaneously, other solutions should be considered such as showing the activation order by means of a circular threaded link set between the icons or having an extra numbering on the icons in the second solution.

3-  Conclusions and Future Work:

For a regular program, there are 5 main factors for UI design (Shneiderman, B., 1992):
1- Time to learn (average time for a user to learn to interact with the system)
2- Speed of performance (how long does it take to perform relevant benchmarks)
3- Rate of errors by users (average number of errors produced by a user or type of users)
4- Retention over time (how much can a user retain his/her knowledge about working with the system after a period)
5- Users satisfaction (this parameter can be determined with questionnaire or conversation)

The effect of our suggestions in improving the UI quality based on these parameters is evaluated in Table 1.

Table 1: The effect of our improvements on the main factors of the UI (H=High, M=Medium, L=Low).

<table>
<thead>
<tr>
<th></th>
<th>Automated Forms</th>
<th>Shortcut Keys</th>
<th>Switch Bar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time to learn</td>
<td>H</td>
<td>H</td>
<td>M</td>
</tr>
<tr>
<td>Speed of performance</td>
<td>L</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td>Rate of errors by user</td>
<td>H</td>
<td>L</td>
<td>M</td>
</tr>
<tr>
<td>Retention over time</td>
<td>M</td>
<td>M</td>
<td>H</td>
</tr>
<tr>
<td>User satisfaction</td>
<td>H</td>
<td>H</td>
<td>H</td>
</tr>
</tbody>
</table>

Although the above results are all theoretical, the great impact of the suggestions and improvements in the software applications is obvious. For example, visibiling invisible objects in the form of suggestions of this paper rapidly show the users the way of interacting with the system between hundreds of various commands and possibilities. This reduces the triggering time especially for the expert user.

In case of novice user, these invisible objects (and many other ones) prevent him/her from mixing up, but the expert user shows great signs of increasing the speed; hence the usability aspect is somehow satisfied. In fact, most of these objects and capabilities hide from the users’ view. Visibling invisible objects help to solve this problem.

Although there are several other guidelines to improve the software’s quality, here the focus was on the most important ones with much more tendencies to the impact on usability aspects. Presented suggestions help to create friendlier and more useful interfaces considering the evaluation factors.

Currently, there are many objects and capabilities in the interfaces, even the web based and mobile forms. However, many mobile cell phones and computer users don’t know most of the invisible objects till the time they change their hardware/software. Visibling them is another idea to identify the system’s hidden interacting capabilities.

Having intelligent forms is another technique that can be used for complicated forms, especially in the limited screens such as mobile cell phones. In the web based forms it can reduce send/receive data to increase speed.

Some important guidelines for designing a friendly UI are discussed and suggestions are made for improving the quality of the UI design.

In addition, the suggestions can be expanded in a separate research paper to improve software’s quality.
References