The Non-Technical Factors of Software User Interface Development

József Tick

Budapest Tech
Doberdó út 6, H-1034 Budapest, Hungary
tick@bmf.hu

Abstract: The design of user interfaces is one of the corner points of the phases of software development regarding user satisfaction. In order to reach an optimal realisation it is practical to build up a quite intensive interaction with the future system users already during the phases of analysis, design and implementation, as well as during testing. In the course of interaction with a future user, the goal is not only to determine the interface to be designed, but also to have a detailed identification of the user. This paper deals with an other viewpoint of user interface development, namely the non-technical features. It focuses on their importance and provides a guideline how to consider them in the development process of software systems.

Keywords: Software Engineering, User Interface Design

1 Introduction

During the development of software systems the development of user interface has become one of the most critical fields. The fact that user interface is that part of a system which "can be seen" by the user and through which the user has an interaction with the certain system, makes this work even more difficult. The efficiency of the use of the whole software system primarily depends on the efficiency of the user interface. However, the efficiency in this case depends not exclusively on technical parameters but quite a lot of other factors have a great impact on the rapid and error free handling of a system as well.

Parallel with efficiency, user satisfaction is also a quite complex feature. This feature has proven to be quite characteristic in certain application fields, primarily in the field of "Office, home, and entertainment applications" [1], even more it can play a determining role when selecting from competitive products.

This paper first analyses the most important features expected from user interfaces then gives an overview on the factors to be considered during analysis and design.

Finally, it recommends a special guideline to be followed in the process of software user interface development.

2 The Most Important General Requirements Set to User Interfaces

The most important demands made on software user interfaces are non informatics, and not even technical ones, but they are basic criteria in connection with usage and handling. The system usage can be put into three categories regarding frequency and regularity:

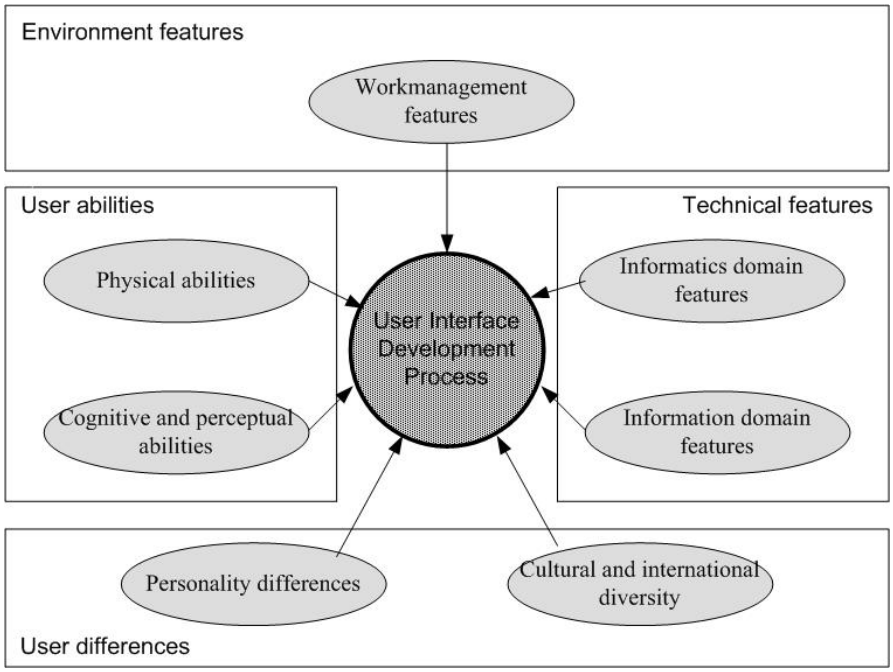
- Short-term, unique, rare usage, which requires high support from and guiding by the system. In this case the user cannot have a routine.
- Non-regular, casual usage where the user does not have a routine yet, but long-term memory already supports the user in handling. A strong support in handling is not justified, even more it can often disturb the user.
- Regular, frequent usage where handling becomes a routine, and a quick and effective handling describes this type of usage. Support can be required only on great demand.

According to [1, 2, 4], in each case of each usage above the most important general features that can be required related to user interfaces can be summarised as follows:

- Easy handling of a user interface and easy navigational possibilities among interface elements.
- Easy learning of a user interface, building on previous knowledge (informatics, mathematics, cultural background etc.) regarding notation- and rule-systems, clear logical structure of the interface and simple solutions.
- Consistency of a user interface. The total interface-system should build on the same set of elements and system of rules.
- Flexibility of a user interface. It should meet the requirements of all kinds of users with different levels of knowledge ranging from beginners to skilled users.
- Adaptability of a user interface, which modifies the behaviour of the user interface according to the frequency of usage, the user's skills, and the differences in users' skills and abilities (the intensity of user support, information windows, short cuts).

3 Features to be Considered when Developing User Interface

During the development of user interface, the features to be considered can be put into four categories (see Figure 1).



This paper does not focus on the trivial, that is the technical field, but it aims to highlight user characteristics and differences between users that are often played down, but which are as important as technical features.

These non-technical features can be given by different parameters that are sometimes hard to determine. Parameters of the different fields can be summarised as follows:

- Physical abilities include not only static physical dimensions. Regarding software systems these cannot be used at all. From this aspect, visual perception, response time as well as the speed, the dynamics and the resolution of movement are of high importance. Considering the interaction with a system, fine tuning of the above parameters, the optimal setting of times and the possibility of their modification raise a crucial question. Think of messages appearing on the screen, the possible problems occurring when moving the mouse, the fine or tiring or even the warning effect of colours in some cases. In the case of graphical applications (especially CAD systems), 3D effects as well as the perception of depth are of special importance. In case of software

products developed for a widespread use, a large deviation of these parameters can be expected and a management of people with physical disabilities must be provided for.

- Cognitive and perceptual abilities include important elements to be considered during the realisation of user interfaces of software systems. From the aspect of design I consider the following elements of primary importance:
    - Short-, and long-term memory, learning
    - Problem solving, decision making
    - Scanning and search abilities
    - Perceptual abilities and perceptual levels
    - Concentration skill, brain-fag

  These parameters play an important role in building up a user interface, in planning a correct position of certain elements and in selecting the right colour-palette of an interface. It must be noted here, that these parameters must be considered not only in case of handling a system at the moment but during the development of routine and comfort feeling as two important feelings through learning and long-term memory. In order to reach success with a product, the development of these two perceptions must be consciously made from both technical and marketing aspects.

- Personality differences: A valid method for the measurement of personality differences does not exist in the market. The most widespread method is the Myers-Briggs Type Indicator (MBTI) [5] based on the personality classification made by Jung. On the basis of this the user can be characterised by four categories:
    - Extroversion vs. introversion
    - Sensing vs. intuition
    - Perceptive vs. judging
    - Feeling vs. thinking

  It is true that the consideration of personality can be of high importance, however, only in case of a certain, quite special application of software systems, the so-called “Life critical systems” it has a determining importance during usage.

- Cultural and international diversity is not a determining feature for traditional stand-alone systems. However, in case of web-based systems, especially if the system is a multilingual one and the target community is not a previously determined nation, then these non-technical features can be highly important in

user interfaces. In case of the latter types of systems the following fields must be seriously worked out:

- Language independent realisation (easily changeable text files, displaying of information, error message, etc.)
- Support of different national sets of characters
- Changeability of date and time formats
- Changeability of measurements, possibility of conversion
- Correct handling of names, forms of address and titles
- Changeability of writing direction (from left to right, from right to left and vertical)

The list is not at all and cannot even be complete, but it is important to call attention to the fact that in case of software products intended to be used in circumstances mentioned above, the realisation of a user interface is often just a surface phenomenon and a radical modification in the software product must stand in the background.

4 Guideline for User Interface design implementing non-technical features

Software engineers often make a mistake when they regard their own user knowledge, background knowledge and skills as starting point during the development of user interfaces and interactions. They take into account what they think logical, aesthetic and easy to be handled. These types of software products are easy to be used by informatics engineers, but an actual end-user feel uncomfortable and insecure during the software usage. A lack of comfort feeling leads to a user dissatisfaction, which can be disadvantageous for the software product in the long run. A significant part of software developments that have already fallen flat can be rooted from a discomfort caused by the user interface, and its difficult and complicated usage.

Several recommendations have already been published to help user interface development. Most of them are rather technical ones and they are closer to the structure of the software, and recommend different realisation patterns [1, 3]. These are very important and must be taken into consideration. Their implementation nowadays, in a quite standardised user interface market, is, of course, necessary. This paper's recommendation however, by going beyond, tries to approach the aspects of user interface development from the side of non-technical features.

The “7 golden rules” described in the recommendation below are based on the features outlined in the previous chapter.

1 **Determine the target community of software users.** In the early phase of software development the community of users of a software product to be developed should be determined by cooperating with the customer. As far as it is possible the community must be narrowed and if it is possible it should be a homogeneous one. We should avoid the category of “everybody uses it” because “nobody will use” this type of product. We should avoid extreme concepts, think of the fact that a “stupid-sure” product can exclusively be used by “stupid users”.

2 **Prepare a profile of an average member of the target group.** Let’s characterise an average member of the target group, who does not exist in reality, but has all the features and characteristics that are typical in the group. Take into account the linguistic peculiarities, the age, the mental abilities, the skills in the application domain, the routine in computer handling and the decision making abilities.

3 **Choose a small numbered control group.** Considering that the whole community of users cannot be involved in testing and the average user is a fictitious person, furthermore, that we need real people during the development, we have to choose a 10-15 member group to test user interfaces. The group must be formed in a way that it well represents the whole community of users.

4 **Make the control group do tests to specify the profile of an average user.** Let’s make a set of tests and make the control group do them. Taking test results into consideration the profile of an average user can be refined. Before starting user interface development the elaboration of an exact user profile should be part of the analysis in order to have an exact picture about the requirements raised to user interfaces.

5 **Draw up a conceptual plan about the user interface.** Knowing the software requirements and the features of users a conceptual plan should be drawn up before a detailed development of a user interface starts. The plan should be reviewed by the customer and if it is necessary it should be modified in several iterations until the customer’s ideas are also enforced and he/she accepts the concept.

6 **Make a prototype in parallel with a detailed design and test it with the control group.** After having agreed on the conception, a detailed design may become a reality. It is practical to prepare a prototype of the user interface side by side the detailed development and to make the prototype be tested by the control group. Depending on the results we should strive for reaching user satisfaction by modifying the prototype and keeping the best efficiency.

During the iteration a more and more exact user interface is reached and included in the detailed plan.

7 **Intensive testing of the realised system by the control group.** The implemented and tested – i.e. the functionally operating – software must be intensively tested by users. We should aim at ensuring valid situations while testing should be planned in a way that beyond measuring the features of short term use (fast and error free usage) we can gain information about long term use (routine formation, monotony, errors coming with tiredness) as well. Valid situations include damage prevention as well. We should examine how robust the system is, the efficiency of information windows and error messages, and the ability of warding off incorrect handling.

The above written rules are based on the experience of practitioners and are concluded from the experience of several projects. In case of smaller sized projects the task is simplified but cannot be ignored. The elaboration of a profile and a test of usage are needed even in case of special systems developed only for a few users.

During the whole development process interaction is of high importance regarding user interface. We have to have a continuous contact with the customer and the control group must be involved quite often. Only in this case further additional modifications, a redesign before delivering and other failures can be avoided during the development.

Conclusions

The development of user interface of software products is a badly handled field, it does not weigh as it should. Rather the technical features are considered, software developers, feeling that their work is done, give only general recommendations in the field of non technical features. The „omission" of users in the phases of analysis, planning and testing is a bad mistake and we pay for it both in the maintenance phase and in the default of further developments. The guideline recommended in this paper can result in more successful software developments, more effective systems and more satisfied users.

References

[1] Shneiderman, B.: Designing the User Interface – Strategies for Effective Human-Computer Interaction Third Eddition, Addison-Wesley, 1998

[2] Sommerville, I.: Software Engineering, sixth edition, Addison-Wesley, 2001

[3] Elberts, R. E.: User Interface Design, Prentice Hall, 1993

[4] J. Tick: „User Interface Redesign based on User Behavior Analyses" Proceedings of the IEEE International Conference on Computational Cybernetics, ICCC 2003, Siófok, Hungary, August 29-31, 2003, ISBN 963 15418 3, pp. 141-144

[5] Shneiderman, B. Software Psychology: Human Factors in Computer and Information Systems, Little, Brown, 1980