Proceedings of the 8th International Conference on Applied Informatics
Eger, Hungary, January 27–30, 2010. Vol. 2. pp. 323–329.

An Overview on Software Ergonomy

Balázs Hadházi-Borsos

University of Debrecen

1. Introduction

To understand better the software ergonomy, and even why we need this science, it is important to take a short overview of its history and evolution. In a few words software ergonomy is about the quality in use of the software. According to the ISO/IEC 9126 - Software Engineering - Product Quality standard, the quality in use consists of following four characteristics[1]:

- Effectiveness: The capability of the software product to enable users to achieve specified goals with accuracy and completeness in a specified context of use.
- Productivity: The capability of the software product to enable users to expend appropriate amounts of resources in relation to the effectiveness achieved in a specified context of use.
- Safety: The capability of the software product to achieve acceptable levels of risk of harm to people, business, software, property or the environment in a specified context of use.
- Satisfaction: The capability of the software product to satisfy users in a specified context of use.

Beside the characteristics listed above, the standard also contains methods for measuring the software quality. Because these characteristics are too abstract to be measured in same way for each software product, software developers have to choose between available methods and they have to find an appropriate balance among the weight of characteristics in order to satisfy the users' needs.

Instead of measuring these characteristics for a completed software product, it is much more interesting to design a software to became ergonomic and in this way to be accepted by the users. Many methods are available for software designers and developers in order to create software more and more ergonomic. It is important to mention, that these methods can be used separately, but a good combination can result in far better solution. For example, using the Design Space test we can compare two or more possible solutions by comparing their advantages and

[1] ISO/IEC TR 9126-4 Software engineering – Product Quality Part 4: Quality in use metrics

drawbacks. The importance of the Fitt's law is remarkable, as it is very useful in measuring many of the pointer's motion on the user interface. As a conclusion, applying the theory, we can take a short overview how a user interface could be changed to achieve better user effectiveness.

2. History and evolution

At the beginning, the ergonomy was the science of the human working. Later, the ergonomy of the "human factors" was about which characteristics and attributes of a human worker have to be checked when a tool has to be changed. Nowadays it is the science of designing a system of man - tool - environment, where the tool can be actually any kind of product, which can be used by a person. In this terminology, computer software is also a tool. According to the evolution of tools, the science of ergonomy has changed too:

- **1950 → classic ergonomy**, the **ergonomy of the "knobs and scales"**. In that time the main goal was to design the user interface in such a way that a human person could use it in the easiest way. It is important to remark, the ergonomy was present moreless in military projects (i.e. user interface of airplanes and other human-driven machines).
- **1960 → system ergonomy**. The ergonomy of the productivity systems means designing the tools needed in production to achieve higher productivity. The ergonomy was introduced in huge factories, where the difference of the profit was remarkable.
- **1970 → product ergonomy**. The entered in the market. The products for everyday were designed to be used more and more comfortable. At this time the normal people met the ergonomy, the customer bought the most comfortable product.
- **1980 → cognitive ergonomy, software ergonomy**. The ergonomy spred in the whole market, and gained upon the world of computers, too. The software ergonomy means the software usability, comfortability, safety (both user and hardware) and user satisfaction. In practice we usually deal with usability only.

In the past when a computer and the software had designed, the computer itself got the highest priority, the user (actually the operator) had to accommodate to the machine. It was very hard to operate such a computer, in most cases very special training was needed.

Later the software developer get in the centre, his viewpoint was normative. The end-user was not inquired, the software was designed in the way dreamed by developer. This resulted into problems, because the developer in many cases had insufficient information about the requested software, and has almost no information about how the software should be used. In the other hand, the software product contained no documentation about how it had to be used correctly. Unfortunately, some developers still do not collect enough information about the requested software, and during software development they do not contact the end-user to avoid

unwanted results.

Nowadays the user is going to get the highest priority. In order to satisfy the users' needs, the computer could be developed (more memory, higher resolution and so on) and the software developer should find the best solution. According to the market rules, the software is a product that will be accepted by the users (customers) if it is comfortable, satisfactory, efficient, safe, ... The specialist have defined some recommendations in order to satisfy the users' needs (in generally). These are only recommendations, and they are too abstract to be counted somehow and in this way to compare two software product. But when a software product is in development process, the designers can take these recommendations into considerations. If a software is developed respecting these recommendations, the product will be more ergonomic[2]:

- Consistency in terminology and structure
- Advanced users need hot-keys, macros, short-cuts
- The spftware should provide appropriate feedback
- Use clear dialogs
- Use appropriate error handling
- Allow the undo
- The user should control the process
- Do not overload the user's short-term memory.

3. Human Data Processing

In order to develop an ergonomic software, the developers and designers apply to psychological researches. The inputs registered by our senses get to the sensory register, where the data has no meaning, and it is kept only for few seconds. The sensory register can hold big amount of data, since its capacity is big enough.

After a pattern recognition process, data is stored in short-term memory (STM), sometimes known as working memory (WM). The process of this storage is called encoding - these steps are automatic. The STM is slow, holds the data for approximately 20 seconds, but its capacity is small: can store only 7 ± 2 data.

The information will be stored in long-term memory (LTM) voluntarily only, namely if we want to store it. This step is called storage. The capacity of the LTM is huge, but the retrieval process is fast.

4. Available methods

4.1. The GOMS model

The GOMS model is very useful to determine how much time is needed to do something on the user interface using the (mouse) pointer. **Goals** are what the user intends to accomplish. **Operators** are actions that are performed to get to

[2]Hadházi-Borsos Balázs: Bevezetés a szoftver-ergonómiába, Debreceni Egyetem, 2009

the goal. **Methods** are sequences of operators that accomplish the goal. There can be more than one method available to accomplish a single goal, if this is the case then **selection rules** are used to describe when a user would select a certain method over the others.

We distinguish three operators:

- perceptual – to percept a sign (average time is ∼100ms)
- cognitive – for revelation (∼70ms)
- motor – to do something (∼70ms).

Because the GOMS model can provide valuable information about many of the user interactions, its importance is significant, but few of the drawbacks have to be mentioned, too:

- Cannot cope with user errors
- The model applies for advanced users only (they know what to do), and cannot cope with beginners
- The functionality is not considered
- Cannot takes in consideration if the user is tired or not
- The model cannot handle the user's feedback

4.1.1. Fitt's law

The modified form of the Fitt's law[3] is very useful to determine the time needed for a user to get with mouse pointer from one point to another (hit an icon, menu item).

Figure 1: $T_{POS} = I_M \log_2(D/S + 1)$

- I_M: a function of perceptual, cognitive and motor operators, and according to the measurements, its value is between 27ms and 122ms, in practice we usually work with a value of 100ms.
- D: distance between starting point and target
- S: the size of the target

4.2. The CONJOINT test

Using the CONJOINT test it can be easily measured the suitability of a software product. To do this, firstly the users' expectations are analyzed. According to the users' opinion some factors are determined. Of course, out of the users' answers, a few hidden factors are also defined for software correctness, quality, stability,

[3]Dr. Izsó Lajos, Dr. Antalovits Miklós: Bevezetés az információ-ergonómiába, Budapest Műszaki Egyetem, 2000

portability, and so on. In the same time, an importance weight can be attached to each factor. In this way, the quality of the software can be measured.

4.3. The "Design Space" test

The goal of the "Design Space" test is to define the Design Rationale, which is actually the documentation of the decisions during design time. It is possible to analyze the product in design time by comparing the concurrent possible solutions. Listing the questions with related options and criteria for these options we can attach a mark to the design. The design space contains the different designs with different marks. These potential solutions can be presented to the customer, who can choose the best one.

The great advantage of this test is to allow the designers to use their creativity. Because ideas have to be analyzed, even if many of them will not be worked out, they can be kept for a next project. But analyzing so many possible solutions can take too much time.

5. Fitt's law in practice

First of all, let's take an example: The user has to hit an area (icon). The mouse pointer is initially 10cm far from the target, which is 0,2cm large. According to Fitt's law, this motion will take 567ms. If we change the user interface by creating a little bit bigger, 0,5cm wide icon, hitting this target from same distance will take only 439ms. This means, if we hit from 10cm a 0,5cm large icon instead of a 0,2cm large icon, we save 128ms.

- $T_{0,2} = 100 \log_2(10/0,2 + 1) = 567\text{ms}$
- $T_{0,5} = 100 \log_2(10/0,5 + 1) = 439\text{ms}$

Using Fitt's law, a simple dropdown menu can be changed in many ways.

- When the menu is opening, the cursor is not positioned in the top (or bottom), but in the middle, in this way the menu items are closer, but it takes time for the user to decide which is the right direction.
- The size of the menu items can be different: the closest one is the smallest, as we get further from starting point, according to Fitt's law the menu item is larger.
- The top and bottom menu items can be "closed", in this way the mouse pointer cannot move out.
- Because the furthest menu items are closed, they should not be large, in this way we can save place on the screen for other information.

6. Valuable pixels on user interface

The most valuable area or pixel on user interface is the current pixel, because no mouse pointer motion is needed to hit it - usually this is used for context menu. If

we use a **circle menu** as context menu, every menu item can be reached in same constant time. We have to mention, a different direction is needed to reach the menu items - extra time is needed to decide the right direction, too. If the menu has to contain many items, more than one circle can be used - if the radiuses of the circles are close to each other, the time to hit an icon on inner circle will be close to time of hitting an icon on outer circle. According to Fitt's law, the icons on outer circle should be a little bit larger.

After the current pixel the most valuable pixels are the four corners of the monitor, as they have infinite extension in two directions - usually the main menu of the operating systems are placed on one of the corners.

The four edges of the screen are also very precious area, as they have infinite extension in one direction - usually an alternate menu is displayed in this region.

7. Application of Fitt's law

In this chapter we will describe an experiment, where the user interface of an application has changed in order to demonstrate Fitt's law correctness. We have the following environment:

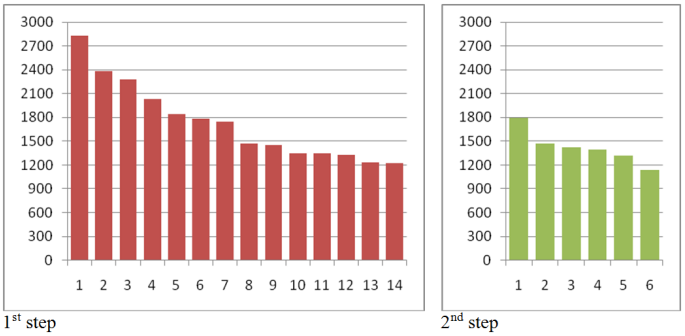
- A web application, where a list of items is displayed.
- When the user selects an item from a list, an overview screen opens.
- There are many fields on this screen, displayed with constant topology (the layout of this screen has not changed during the experiment).
- We recording the time between the overview screen is displayed and the field help icon is hit.

The data were recorded in two different steps. Between these steps, the field help icon was changed: The 13*16px icon was replaced with a new 17*16px icon. None the less this modification seems to be minor, the icon's width was increased with more than 30

Note: In practice, the users do not know exactly where the target is - first they have to take a look at the screen and have to recognize the layout, namely they have to choose the right direction. The Fitt's law cannot deal with this situation, as it is working with strictly controlled environments. Out of the direction, in real world the users start their mouse pointer motion from different points of the screen, namely the distance between the starting point and the target is not constant.

People involved in this experiment can be treated as advanced users, as they are using daily this application. So they are familiar with the mouse and with this user interface, so no special training was needed. In first step of experiment 14 people were involved, and altogether 206 data were recorded. In the second step with contribution of 6 people, 257 measurements were recorded.

The tables below show the data resulted after averaging (by user) of the two steps of experiment:



1st step 2nd step

In the first step, with small icon the slowest user's average time is 2829ms, and the fastest user's average time is 1220ms. In the second step, where the target was a larger icon, the slowest user's average time was 1796,83ms, and the fastest user's average time was 1133,76ms:

	max	min	average
before	2829,00	1220,00	1734,92
after	1796,83	1133,76	1421,87
difference	**1032,17**	**86,24**	**313,05**

As conclusion, our expectation was right. By increasing the icon's size both the fastest and slowest users' average time was decreased. The global average time is also decreased significantly.

References

[1] ISO/IEC 9126-1: Software engineering - Product quality - Part 1: Quality model ISO/IEC, 2001

[2] ISO/IEC TR 9126-4 Software engineering - Product Quality Part 4: Quality in use metrics ISO/IEC, 2004

[3] Hadházi-Borsos Balázs, Bevezetés a szoftver-ergonómiába Debreceni Egyetem, 2009

[4] Dr. Izsó Lajos - Dr. Antalovits Miklós, Bevezetés az információ-ergonómiába Budapest Műszaki Egyetem, 2000

[5] MacKenzie, I. S., Fitts' law as a performance model in human-computer interaction. Doctoral dissertation. University of Toronto: Toronto, Ontario, Canada., 1991

[6] Sanders, M.S., McCormick, E. J., Human Factors in Engineering and Design McGraw-Hill, New York, 1993

[7] Shneiderman, B., Designing the User Interface Reading, MA: Addison-Wesley, 1987, 1992