Study of ergonomic criteria for evaluating the software user interface

Study of ergonomic criteria for evaluating the software user interface

D V Yatsenyak[1], V P Oleksiuk[1,2] and N R Balyk[1]

[1] Ternopil Volodymyr Hnatiuk National Pedagogical University, 2 M. Kryvonosa Str., Ternopil, Ukraine
[2] Institute for Digitalisation of Education of the NAES of Ukraine, 9 M. Berlynskoho Str., Kyiv, 04060, Ukraine

E-mail: yatsenyak_dv@fizmat.tnpu.edu.ua, oleksyuk@fizmat.tnpu.edu.ua, nadbal@fizmat.tnpu.edu.ua

Abstract. This article summarizes research on the development and evaluation of a set of software usability measurements. Today they are called "ergonomic criteria". The authors summarize the research work carried out on the development and evaluation of a set of software usability measurements. The paper contains a description of each such criterion. All of them together provide a quality user interface. The authors highlighted the significant advantages of a well-designed and well-thought-out interface. The paper also contains an analysis of each of these criteria. The authors of the article discuss ergonomic problems of software development. As a result, some characteristics of the software are identified. They reduce the ergonomics of the user interface. The paper also discusses ergonomic issues and the things developers need to get rid of to make the app enjoyable. The success of solving this problem in software development largely depends on how functional, clear and user-friendly the interface is. The authors give some examples of bad user interfaces. They are accompanied by an explanation of the problems and solutions to these shortcomings. The systems, designed with practicality in mind, are ergonomic and work exactly as users expect, allowing users to focus on their own tasks rather than the specifics of interacting with the system.

1. Introduction

In the 21st century, in the age of digital transformation and automation of human activity, no one can do without any technical device. This allows you to expand or supplement natural capabilities such as complex computing, transmitting and receiving information, improving accuracy, moving in space, making decisions, and more.

The **aim** of this work is to identify the main set of measurements of ease of use, commonly known as "ergonomic criteria" based on a review of already written research papers. The research concerns the process and results of one branch of international standardization of human-system interaction. The article discusses the points where the results of standardization could be more successful.

Software developers solve many problems when designing a user interface (UI). As a result, users can use information systems efficiently and economically. Typically, developers consider the features of the software separately from its user interface. But some researchers believe that the user interface is a complement to system functions. But users do not differentiate between functionality and software interface. For UI this is the same program. For them, if the

software interface is good, then the program itself is good and convenient. This means that this understanding of UI is too limited. In fact, the user interface contains all aspects of design that affect user-system interaction.

Innovations in technology and automation in control systems have increased the need to improve the user interface and all processes for its creation [1]. The benefits of Internet technology create a unique opportunity to quickly and efficiently provide employees with data and access to it. There have long been technologies that can significantly improve the UI. However, they alone do not produce ergonomic interfaces.

For example, the graphical user interface itself is no more ergonomic than the text interface. Experience has shown that it may be less useful if designed incorrectly [2].

Because the user considers the interface as a key factor in the functionality of the program, a poorly designed interface severely limits the functionality of the system as a whole [3]. Companies that do not seek to develop ergonomic UI for their products and get all the benefits provided by modern technology, weaken their competitive position. Timely and professional development of the interface leads to increased software efficiency [4]. And this reduces the duration of user training, reduces the cost of redesigning the system after its implementation, promotes the fullest use of the functionality embedded in the software, etc.

Software designed with practicality in mind is ergonomic and works exactly as users expect. As a result, users focus on their own tasks rather than the specifics of interacting with the software. Ergonomic software products are easier to learn, they are more effective, they also minimize the number of human errors and increase user satisfaction. But this does not happen by itself. The effective user interface is the result of the developer's understanding of the need to pay attention not only to the data but to the user, his tasks, and activities.

Here are some of the most significant benefits of a good user interface

- reduction of losses of productivity of users at system implementation;
- faster restoration of the lost productivity;
- reducing the number of human errors;
- improving the morale of users;
- reducing the cost of system support;
- reduction of costs for UI redesign.

2. Generalization of evaluation criteria for the user interface

Each hardware or software product is a set of different functions. This provides users with various controls. Therefore, it is necessary to have standards for evaluating the technology or products of the user interface. Obviously, UI directly depends on the task that solves the software, data input and output. However, all this data can be presented to the user in different forms. The success of solving the problem in software development depends on how functional, clear and user-friendly the interface is.

If the interface forces its users to make mistakes, then this interface is bad, at least it is worse than the interface that helps to avoid such mistakes. The UI design process is most influenced by the designer's own perception of clarity, convenience, beauty. Therefore, it is very important to assess the quality of UI. Carrying out such assessments in the early stages of the design process avoids numerous errors, miscalculations, deviations of the software by the end-user.

There are many ways to assess the quality of UI. The general model of the software evaluation process includes the stages of determining the quality requirements. In other words, the interface of a software product can be evaluated based on various standards, quality indicators and interface attributes. Although the evaluation of the quality of the user interface process is quite subjective and difficult to formalize, it is safe to say that a good interface should ensure the efficiency and productivity of the user's work.

Utility, usability, desirability are three things that should underlie the design of any software product [5, 6]. Taking into account the needs and expectations of users from the design of the interface, we can identify certain recommendations that should be followed and based on them to edit the user interface.

- *Display the real world in the user interface.* The developer should try to reflect the language and concepts that users use in the real world, based on what their target audience is. Providing logically structured information and engaging users' expectations from their actual experience will significantly reduce cognitive load and facilitate the use of the software product.
- *Standards and consistency.* Developers of the user interface must ensure that conditional standards for both graphical elements and terminology are maintained. For example, an icon that represents one category or concept should not represent another concept if it is used multiple times in a program or site.
- *Efficiency and flexibility.* As the frequency of use increases, there is a need for fewer interactions that provide faster navigation and ease of use. This can be achieved with abbreviations, function keys, hidden commands, macros. A good solution would be to allow users to customize or adapt the interface to their needs so that frequent actions can be performed with more user-friendly tools.
- *Recognition.* Human attention is limited, and we are only able to store a few items in our short-term memory at a time. Because of the short-term memory limitations, designers need to provide an interface so that users can simply use recognition instead of remembering the information they remember in fragments. Recognizing something is always easier than remembering it because recognition involves the perception of signals that help us penetrate our vast memory and allow relevant information to appear.
- *Documentation and reference.* In an ideal software product, users navigate the system without resorting to documentation. However, regardless of the type of solution, documentation is still required. When users need help, it's important to find it easily. According to the task, the documentation should be designed in such a way that it guides users through the necessary steps to solve the problem they face.
- *Error prevention.* When designing, developers aim to minimize potential errors. Ordinary users should not be encouraged to identify and correct problems that may go beyond their level of knowledge, for this purpose the work of testers is provided. Eliminating or flagging actions that can lead to errors are two possible ways to prevent errors.
- *Freedom of action of the user.* A good solution is to create a digital space where the user can take steps backward, including undoing and redoing previous steps.
- *System status.* It is important to inform users about the current state of the software product, whether it is "download", "search", "recovery" or other variations. Easy to understand and clearly visible status should be displayed on the screen for a reasonable period of time, in particular, while the system is in the appropriate state.
- *Minimalism.* The point of this tip is to minimize clutter in the interface. All unnecessary information competes for limited resources of the user's attention, which can prevent the search for relevant information in the user's memory. Therefore, the display should be limited to the necessary components for current tasks, while providing clearly visible and unambiguous means of navigating to other content.

Based on these recommendations, the quality UI can be based on a number of criteria. These criteria vary from author to author [2, 7, 8]. We summarize the list of these criteria and present the following characteristics of software quality assessment in the table 1.

Scientists [2, 9] offer a sequence of stages of designing an ergonomic user interface, such as

Table 1. Criteria for evaluating the user interface.

Group of criteria	**Criterion**	**Description**
Functionality	Suitability	Compliance of the program with the declared set of functions and the ability to perform relevant tasks
	Accuracy	Software attributes that require the correctness and relevance of results or effects
	Interoperability	The ability of software to interact with specific systems, the ability to integrate it with other applications and services
	Concordance	Compliance of software with standards, agreements and laws
	Securicity	The ability of a program to prevent unauthorized access, accidental or intentional alteration of its data
Reliability	Meturity	Frequency of software failures caused by errors in its design and development
	Fault tolerance	Ability to maintain a certain level of performance in cases of software errors or violations of a particular interface
	Recoverability	Ability to restore functionality and data that has been damaged due to failure of the application or service
Usability	Understandability	Measured by the user's efforts to understand the logical concept and applicability of the software
	Learnability	It is measured by the user's efforts to learn how to use an application or service
	Operability	It is measured by the user's effort to use and operational control of the software
Efficiency	Time behavior	Determined by the response time and processing of functions by the application
	Resource behavior	Determined by the amount of OS resources used and the duration of such use
Portability	Adaptability	Ease of adapting the software to different operating conditions other than those provided for this application
	Installability	The effort required to implement the software in a specific environment
	Replaceability	Simplicity and complexity of using a software application instead of another software tool

- specification of design objective;
- analysis of tasks and function of the interface;
- analysis of requirements and preferences of the operator;
- selection of rules of interface design;
- development of the structural scheme;
- rapid prototyping;
- development of experimental evaluation of the prototype interface;
- analyses of tests results and develop recommendations of finalize;
- creating of functional software;
- a comprehensive experimental evaluation of the interface.

3. Problems of developing ergonomic user interfaces

Today, ergonomics of software as a science is still at the stage of initial development. But active research in this area has already produced many useful conclusions regarding the factors affecting the performance of software use. As experience shows, many interfaces are unsatisfactory [7, 8, 10]. This is because they are too narrow-profile and require a fixed combination of user skills. A good interface should serve different users. One of the main problems of ergonomic UIs creating is the construction of a user‘s activity model. This framework can help to receipt of estimates of error-freeness, resource consumption indicators and variant analysis of alternative methods of data processing [11].

In general, we can distinguish between human-oriented and computer-oriented approaches to the development of user interfaces. However, any of these approaches must be the result of an analysis of the user's tasks. From ergonomic principles [12] it is possible to understand why modern software cannot be used productively enough in the context of various and unstructured tasks. Considering the typical modern software products, we have identified some shortcomings, such as

- *No integration.* To get the desired results, end users often have to combine the work of several applications. Each application has its own data format requirements. This usually requires manual user intervention. For example, using a text editor, output data from one program to a specific input format for another. Another problem is that once data has been imported, its structure is often lost. Each program and the operating system itself has its own requirements for data formats and modes of their import. Some programs have several such modes that require the user not only the relevant knowledge, but also an understanding of the current application of the desired mode. This is a high cognitive load and therefore a source of error.
- *Loss of context.* During the task, the user finds himself in a situation where the required data is not immediately available. They must be obtained in some way outside the context of the current task. Many systems switch only when the expected task enters a quiet state. Perhaps instead of suspending such a task, it should be closed. This means that the context of the unfinished task is lost and the user must explicitly return it during recovery.
- *Mysteriousness.* The exact forms or formats of data that are required for commands or functions of the program, often have to look for in huge and difficult to interpret manuals.
- *Swiss Army Knife Syndrome.* Many features of the software package, such as e-mail, finding a letter (searching), archiving it (storing it), composing a letter (editing), and so on. Such functions are in fact exhausted versions of much more general functions that happen to be applied to one particular context. This can lead to a huge duplication of functionality and increased inconsistency, causing constant danger of confusion in the mode and deepening the overall complexity of the user interface.

It may seem that the shortcomings were deliberately designed for the purpose of preventing users from making effective use of technology. However, developers never deliberately create interfaces for everyday use with flaws. Design flaws are identified by real-world testing, which detects uses that have not been properly taken into account at the design stage. Disadvantages are the sins of omission (not taking into account something), not the sins of committing (intentionally making it difficult to use something). Exceptions may be software products designed for scientific purposes to study the impact of these deficiencies on users.

The existence of such shortcomings imposes a large cognitive load on the user. This distracts him from the real task that needs to be done. As a result, numerous user errors become inevitable. This is especially true for routine tasks, in which computers easily outperform people. Peoples start making mistakes due to a drop in attention during repetitive tasks. But a significant portion of the effort in using computers as an intelligent tool is a task without intelligent content. Such as data conversion formats. The current situation forces users to constantly spend much of their attention on the low-level aspects of communication. Although computers are much better than humans at tracking the exact state of a task, the user is forced to store the details of the context in their own memory, not the computer.

Mistakes in most cases are just "stupid" mistakes that cause irritation and loss of time but do not cause much harm. From time to time, however, a well-understood user error is costly, causing incorrect results, or the loss of a large amount of work, or even the irreversible loss of information, which causes great suffering [13]. Even without such troubles, the total time users spend guessing, puzzling, or trying to figure out something rather trivial is impressive. However, these "inevitable" mistakes are inevitable only because the simplest principles of ergonomics are not applied to the design of the user interface. An important prerequisite for a good interface is that the system appears to the user as an integrated system with a single interface, rather than a set of different interfaces, one for each individual function of the system, even if each individual interface is well-designed. This, in turn, requires that a single conceptual framework underpin the entire system, a framework in which a variety of programs can be integrated naturally.

4. Examples of disadvantages in the ergonomics of user interfaces

Good UI design makes user interaction with the application or site simple, intuitive, efficient, and smooth. A bad user interface can be annoying or directly prevent users from continuing to work. The shortcomings in the examples below may seem comical or downright annoying, but they are still found in UI design.

Use of intuitive notation.

Changing the language is the first thing a user wants to do if they open a site or software they can't read (see figure 1). It also makes sense to make this option as simple and optimal as possible to choose before performing any other actions such as login, registration, password change. The most successful is the idea of using intuitive notation. A globe icon or language icon will greatly facilitate the search for the user. However, caution should be exercised when using country flags. This is because common languages or branched dialects are easy to confuse.

Choosing the right tool to work

The basis of interface design is choosing the right tool to work with. This should usually mean respect for common symbols and the use of familiar controls. For example, for clarification, flags are used when it is possible to mark several options or radio buttons (see figure 2). In this case you should use the plus and minus buttons, which allow user to adjust the number of the desired item to the his/her choice .

Ambiguity

At first, it is worth noting that users are accustomed to marking checkboxes themselves, rather than removing already selected ones. However, in this case, the problem of the interface is the ambiguity of the interpretation of concepts. To remove an ingredient in order, do you need

Figure 1. The problem of language choice.

Figure 2. Inappropriate controls.

to uncheck or leave it? This confusion misleads the user and causes several misunderstandings and issues (see figure 3). That is, uncheck no salad to remove the salad. In this case, it is better to change the text related to each of the checkboxes to a clearer and simpler one.

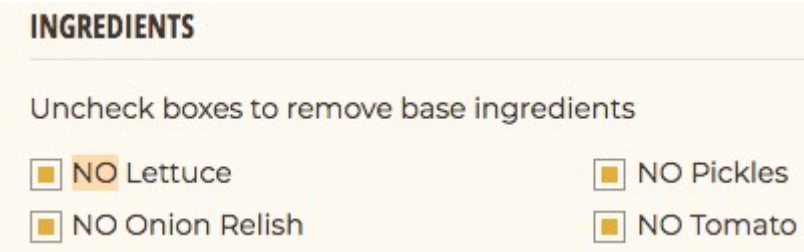

Figure 3. Ambiguous interpretation action of the checkbox.

Excessive choice of options

Drop-down lists to choose from numerous items are not a good solution. It is much easier

for a user, for example, to enter his year of birth than to flip through a long list over the years. Regarding the choice shown in figure 4, developers should consider whether to use drop-down lists at all or to change the approach to filling out the form. In the best case, suggest the user enter data from the keyboard yourself, and only then, according to the entered combination of words or symbols to give examples of filling options.

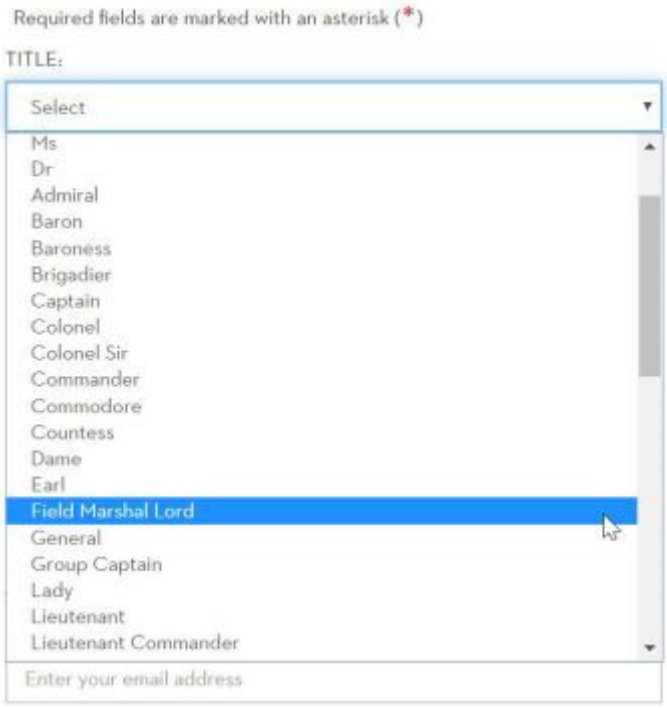

Figure 4. The disadvantage of a long list.

Thinning of the contents

To find out the entire history of the company, the user needs to go to five different web pages, although it would be much more ergonomic to form the whole concept of the company in one page (see figure 5). This thinning of the content in different sections forces the user to "wander" on the interface and as a result to spend extra time. The developers of this site should expand the content of one page "About" at the expense of the content of nested child pages so that the user visiting this site received complete information about the company without making unnecessary transitions on the pages.

5. Conclusions

The problem of designing user interfaces is currently relevant. Therefore, to facilitate the work with the software product, developers need to think about and create an ergonomic user interface that would accordingly perform all the tasks assigned to it. This paper argues the importance of ergonomic interface, the benefits of a good UI, such as reducing the overall cost of system support or redesign, a significant reduction in the number of human errors during use.

Given the user needs, expectations, and expectations of the interface design, the paper identifies strong recommendations that should be considered and included in the editing of the user interface. These include well-understood documentation and usage tips that unscrupulous UI developers can ignore or the user's freedom of action based on digital space with possible return steps. It is also recommended to resort to minimalism, which includes the definition of

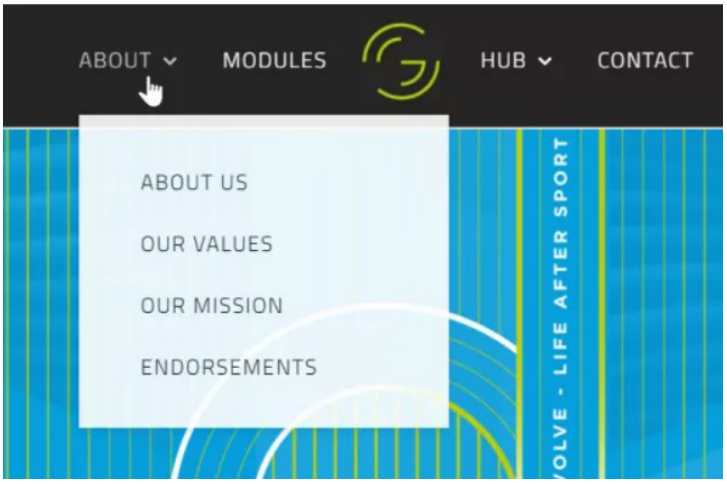

Figure 5. Disadvantage of a multi-page website.

the primary functions of the interface and the elimination of cluttered clusters of components that are not necessary for current tasks. These recommendations and advice are the basis for a number of criteria.

The article contains the characteristics of software quality assessment, namely summarizes the criteria for ergonomics of the user interface based on a review of already written scientific papers. All sixteen criteria are divided into five general groups: Functionality, Reliability, Usability, Efficiency, Portability, which give an understanding of what this or that characteristic refers to.

Considering the published scientific works, the sequence of stages of designing an ergonomic user interface is noted by scientists. One of these steps is the analysis of test results and the development of recommendations for refinement, which can already assess the problems that may arise during the development of ergonomic user interfaces. Therefore, the article lists some typical shortcomings in modern software products that have not been removed. In total, 4 shortcomings were noted: Swiss Knife Syndrome, Mystery, Loss of Context, Lack of Integration, which are reasonably recommended to eliminate when refining the UI.

Ergonomic criteria should take into account possible errors of user users. Therefore, ergonomic applications should be interactive, intuitive, interruptible, "indulgent" to mistakes. For a practical picture of the problems of the non-ergonomic user interface, the work contains illustrated examples of shortcomings that are still encountered by users and significantly affect usability. Ordinary people who use these software products face problems and misunderstandings. Instead, the authors drew attention to ways to correct existing errors and changes that would improve user interaction with the software interface. This seemingly understandable problem with the default interface language mismatch is easily eliminated by replacing words with intuitive notations that successfully implement the basic functions of the user interface. An efficient interface is the result of the developer's awareness of the need to pay considerable attention not only to the data that the user will work with, but also to the actual details of the interface.

ORCID iDs

D V Yatsenyak `https://orcid.org/0000-0002-8427-5532`
V P Oleksiuk `https://orcid.org/0000-0003-2206-8447`
N R Balyk `https://orcid.org/0000-0002-3121-7005`

References

[1] Pasichnyk V, Kunanets N, Veretennikova N, Rzheuskyi A and Nazaruk M 2019 Simulation of the social communication system in projects of smart cities *Proceedings of the International Scientific and Technical Conference on Computer Sciences and Information Technologies* vol 3 pp 94–98 URL https://doi.org/10.1109/STC-CSIT.2019.8929825

[2] Bastien J and Scapin D 1995 *International Journal of Human-Computer Interaction* **7** 105–121

[3] Yamaoka T 2013 Evaluating user interface design using hierarchical requirements extraction method (rem) *Universal Access in Human-Computer Interaction. Design Methods, Tools, and Interaction Techniques for eInclusion* ed Stephanidis C A M (Berlin, Heidelberg) pp 137–142 URL https://doi.org/978-3-642-39188-0_15

[4] Fernández J and Marías J 2021 Heuristic-based usability evaluation support: A systematic literature review and comparative study *Proceedings of the XXI International Conference on Human Computer Interaction* pp 1–9 URL https://doi.org/10.1145/3471391.3471395

[5] Clig:guidlines analytics URL https://clig.dev/#further-reading

[6] Human interface guidelines URL https://developer.apple.com/design/human-interface-guidelines/

[7] Penha M, Correia W, Soares M, Campos F and Barros M 2013 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **8012 LNCS** 379–388

[8] Fernandes F and Paschoarelli L 2014 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **8518 LNCS** 104–115

[9] Khan M 2019 *International Journal of Innovative Technology and Exploring Engineering* **9** 1406–1410 URL https://doi.org/10.35940/ijitee.A3997.119119

[10] Semerikov S, Striuk A, Striuk L, Striuk M and Shalatska H 2020 Sustainability in software engineering education: A case of general professional competencies *Proceedings of the International Conference on Sustainable Futures: Environmental, Technological, Social and Economic Matters* vol 166 URL https://doi.org/10.1051/e3sconf/202016610036

[11] Lavrov E, Pasko N, Siryk O, Burov O and Natalia M 2020 Mathematical models for reducing functional networks to ensure the reliability and cybersecurity of ergatic control systems *Proceedings of 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering, TCSET 2020* p 179 – 184 URL https://www.doi.org/10.1109/TCSET49122.2020.235418

[12] Honorio dos Santos W, Gamez L and Mancini F 2017 *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* **10279 LNCS** 26–38

[13] Oliinyk B and Oleksiuk V 2019 Automation in software testing, can we automate anything we want? *Proceedings of the 2nd Student Workshop on Computer Science Software Engineering* vol 2546 ed Kiv A, Semerikov S, Soloviev V and Striuk A (CEUR-WS.org) pp 224–234 URL http://ceur-ws.org/Vol-2546/paper16.pdf