Embedded Self Organizing Systems (Vol 8. No 1. 2021) (pp.16-21)

Embedded Selforganizing Systems

Issue Topic: "Intelligent Approaches for Smart Cities"

Classification for Quality Assessment of the User Interface and its Application in the Development of Web-applications

Nikolai Gervas
Novosibirsk State Technical University
Department of Computer Science
nik.gervas@mail.ru

Evgeny Romanov
Novosibirsk State Technical University
Department of Computer Science
romanov@corp.nstu.ru

Wolfram Hardt
Technische Universität Chemnitz
Department of Computer Science
wolfram.hardt@informatik.tu-chemnitz.de

Abstract[1]—The article considers a classification for validation and quality assessment of the user interface (UI) from the point of view of the main aspects of design and its application in the development of web-applications. The problem with inaccurately crafted user interface requirements is relevant and as a result, developers often have to redesign the interface and architecture of the application. The article analyzes the role and place of UI in the architecture of client-server applications, analyzes aspects of UI design, on the basis of which the classification is formed. The classification is used to analyze UI design oversights of the developed web-applications for BPMS "Fireproof Corporation" company. Based on the results of UI validation, a set of typical UI design oversights has been added.

Keywords—user interface (UI); quality assessment of the UI; UI design aspects; business process management system (BPMS); client-server application (CSA).

I. INTRODUCTION

There is a problem with inaccurately crafted user interface requirements according to business analytics data. As a result, developers often have to redesign the interface and architecture of the application. To make this process manageable, it is logical to propose a classification of common UI design oversights, on the basis of which the user interface is validated, assessed the quality of the UI and to anticipate design oversights at an early stage. The classification is based on the main *aspects* [1, 6] of the user interface design, it also determines the *frequency of occurrence* of this oversight, the *impact* of the oversight on the work with the system and the *design phase* at which this oversight occurred.

Using the developed classification, a set of typical UI design oversights has been formed. For UI validation, a specific aspect should be selected to validate the application's user interface against a set of common user interface oversights. In the future, the set of typical UI design oversights should be expanded, which will allow better checking the application interface for oversights.

II. THE ROLE OF UI IN THE ARCHITECTURE OF CLIENT-SERVER APPLICATIONS

The system of client-server applications can be described as a distributed system, consisting of three main elements of the architectural pattern MVC (MVP), but interpreted much more broadly, as the main categories, which can include a code component, design artifact and any other entity related to the CSA [1, 2]:

- **model** - everything related to the presentation of data and operations on them in the CSA: business entities and connections between them, the server database, business objects, local user data, the model of the subject area of the CSA itself, means of access, updates and synchronization. This also includes atomic one-time operations on business entities;
- **behavior** - a description of the software system in the form of elementary interactions - precedents and their detailed interpretation - scenarios;
- **view** - components and processes of display and interaction with the user or the external environment. User interaction code (display, input / output) should not be located in the components responsible for the presentation and processing of data in the program.

The *model* is a set of business entities (BS) and relationships between them. There are two types of business entities: basic (such as account, mail, address, profile, artifact (file)) and domain business entities. Relationships between business entities fall into two categories - structural (associations) and behavioral (dependencies). Typically, a business entity is accompanied by a state diagram describing its life cycle in the system. States are integrated into business entities.

There is the greatest uncertainty regarding the *behavior* component, caused by the diversity of points of view on this component and its implementations in different models and patterns. Associated with it are terms such as *use case, controller, scenario, view model*, which are related to both architectural solutions and systems analytics. Behavior should be understood as the *implementation of a business process in the form of a set of interacting scenarios (use cases)*. The description of the behavior includes an abstract component - a *business scenario* in the form of a business process implementation and its concrete implementation in the form of *view scenarios*. The business scenario exists solely as a component of systems analytics, while view scenarios are implemented in controllers, view models, and other architectural components.

In relation to *view* layer, there is a unity of opinion. However, view is not always clearly separated from behavior both at the level of systems analytics and in architecture. It is necessary to formulate the key properties of the view:

- the view cannot contain any component of the behavior (scenario), for example, make a control (button) visible, call an API function, etc.;
- the behavior, in turn, does not depend on the concrete implementation of the view in the UI and the model of the user interface (a single form "cockpit", a chain of dialogs - "scenario driven by the UI"). View scenarios are trajectories in the UI model, there can be no one-to-one correspondence.

Interaction of the view component and its scenario is carried out through two interfaces - *actions on the view model and events of the view model*. At any event or user operation on the UI element, they are translated into the functions of the event interface, the implementation of the interface functions of actions on the model is translated into actions on the UI elements.

Based on the above description, the place of the UI can be determined as follows (Figure 1):

- the business scenario is implemented as a *trajectory* (dotted lines on Figure 1) in the user interface of the application (the sequence of opening forms and actions with controls);
- the same component can participate in the implementation of several scenarios - there is no one-to-one correspondence between UI elements and business scenarios;
- the components of the user interface in the form of architectural classes do not contain elements of behavior (scenario), are separated from the components that determine the behavior, and interact with them through the interfaces of events and actions - the MVP pattern;
- the view scenario is implemented by a separate component (controller, representative), has its own state (behavior model) and uses business entities (business objects) of the model in accordance with their states.

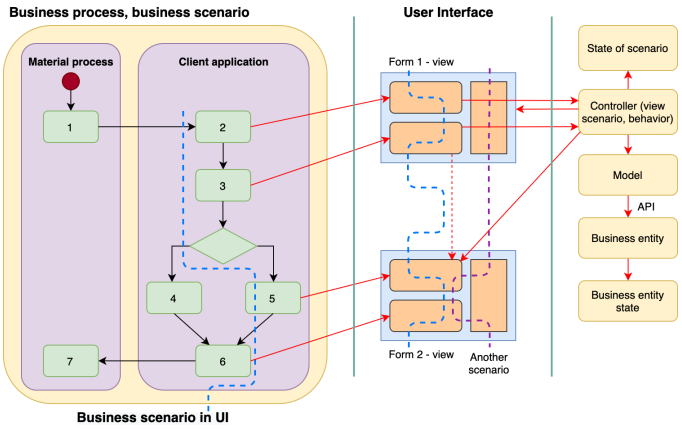


Figure 1: The role of UI in the client-server architecture

III. Analysis of UI design aspects

When assessing the quality of the UI, it is necessary to understand how much it helps or hinders the user in achieving goals while working with the system [3,6]. There are a number of aspects from which UI should be considered.

A. *Performance*

The performance of the UI or the speed of working with it is a comprehensive assessment of the entire process of the user working with the system through the UI and includes the following stages:

- purpose (deliberation);
- sequence of actions (deliberation);
- execution;
- perception;
- evaluation of the result.

A technique known as GOMS (Goals, Operators, Methods, and Selection Rules) is used to assess UI performance. The ways in which actions are performed also affect performance (arranged in order of increasing):

- menu;
- hotkeys for advanced users;
- pictograms, their main drawback is the difficulty of choosing a combination;
- an item in the picture with the action assigned to the pictogram;
- direct manipulation.

In psychology is used the term *focus of attention* - concentration of attention on a certain object, its behavior and control. Restoring the focus of attention requires certain time and psychological costs. In UI, focus refers to objects on the screen that the user sees and can manipulate. If, in the process of work, attention switches to another object, then in

order to continue working with the first object, it is necessary not only to appear on the screen, but also to restore the context of the performed action when restoring the focus of attention, which includes:

- the action to be performed;
- the step the user left off;
- entered parameters;
- the current input focus is the cursor position.

Duration of physical activities. Any action can be either *quick or precise* [4,5]. The time to reach the target is inversely proportional to the size of the target and the distance to the target. The same is true when manipulating any objects in the UI. Accordingly, the shorter the manipulation movements, the higher the productivity. Recommendations for using UI elements:

- context menu (minimum distance);
- dialog box in place of the control;
- screen border as a pseudo button. When the cursor is "sticky", a large UI element appears at the edge of the screen, which allows for fast positioning that does not require precision.

The duration of the system reaction. If the operations performed by the system lead to tangible delays, then a correct estimate of their execution time is required, during which no user intervention is required. Recommendations:

- before starting a long process, all data must be received immediately, it is unacceptable to request additional data after the start of the operation;
- setting a timeout for pop-up windows with confirmation of the operation, after the expiration of the time interval, a positive response is accepted by default;
- implementation of a real progress indicator. A typical mistake is to ignore the final operations when the indicator with the value "0 seconds left" is displayed for a long time.

B. *Human errors*

Humans tend to make mistakes. The user-friendly UI suggests that the system is not a mentor, but offers options for fixing it. Errors can be caused by various reasons:

- knowledge gaps in the subject area;
- typos;
- motor errors associated with inaccurate mouse manipulation;
- decreased attention, skipping / ignoring warnings.

Error prevention measures:

- training of users in the process of work;
- reduced requirements for attentiveness;
- increasing the legibility and visibility of indicators;
- reducing the system's sensitivity to errors.

C. *Memorization and screen space allocation*

All information that the user receives from the system passes through the graphical interface. And here it is very important how it will be structured for presentation and in what form it will be rendered. Formally, these questions are not related to functionality, but they are extremely important [7].

Memorization. In reality, a person is able to effectively manipulate only objects that are directly in the field of view. The volume of short-term memory is limited to 7 ± 2 units of unassociated data, i.e. data, the memorization of which did not arise figurative associations. When designing a graphical interface, it is necessary to minimize the need for such memorization. In the structure of the graphical interface, objects are divided according to their level of accessibility and the need for the user to use short-term memory:

- directly visible;
- directly accessible through visible associated elements - bookmarks, icons;
- selectable through visible associated elements - drop-down lists, pop-up windows;
- located in a chain of calls known to the user, for example, opening a file in a dialog box via the File menu;
- located in a chain of calls unknown to the user, for example, a configuration method or parameter unknown to the user.

The last point has to do with the *learning* factor. The rest - to the factor of *performance* and to a certain focus of attention in it.

Screen space allocation. For many applications, the graphics screen space is a critical resource that must be skillfully allocated between the displayed data. One of the hard-to-solve problems is the *elimination of unnecessary information*. Superfluous information can be understood as unnecessary, obsolete or no longer used information found during the search. Usually in the process of working with the program, the number of active elements - open windows, bookmarks, etc. is constantly increasing, and they have to be closed or removed manually. There is usually no means of collecting such *interactive garbage* - UI elements (windows, bookmarks, icons) created in the course of work, but no longer used.

- In the development system, open files are displayed as bookmarks with the following rules:
- the number of visible bookmarks is within ten, the size of the bookmark depends on the length of the name;
- bookmarks are displayed in the order of opening files, repeated access does not change their order in the list;
- to get a hidden bookmark, necessary to scroll through the list in the bookmarks bar or open the full list with a single click.

When constantly working with a large project, necessary to periodically close unnecessary windows. Therefore, it

would be logical to have tools that track and close long-unused or once-used windows.

D. Subjective perception

This aspect relates to the actual appearance of the interface, graphic design and the peculiarities of its subjective perception by the user [8]. Aspect includes:

- socio-psychological perception - fashion, "beauty";
- technical design - a combination of aesthetics with manufacturability;
- psychological feeling of comfort at work.

Socio-psychological aspects. The graphic design of applications is highly fashionable, it can be focused on certain social groups that make up a significant part of the users, it often has to reflect the requirements of the brands that it promotes, etc.

Principles of technical interface design. If the socio-psychological factor uses the interface as an additional factor in attracting attention, then the technical design requires from it exactly the opposite [9] - it should be as invisible as possible, transparent and subordinate to the functional:

- the interface is not an end in itself, it is invisible;
- the interface is functional and informative;
- an interface is a subject of long-term use;
- the interface is technologically advanced, it provides productive work, minimizes errors;
- the interface is harmonious - all elements are commensurate, proportionate, made in the same style.

Subjective feeling of comfort. Not always the quality of the graphical interface can be expressed by the measured parameters. The subjective feeling of the friendliness of the interface is formed, among other things, from a number of psychological factors [10, 11]:

- subjective feeling of speed of work: filling pauses with background actions, breaking actions into smaller ones;
- a sense of control over the system;
- self-expression - the ability to personalize the program;
- a reasonable system of identification and protection: choosing a login instead of entering - cookies, a drop-down list, entering a password in plain text, remembering a password for a limited period. Usually, protection systems bring significant discomfort, because actions to prevent potential harm are not perceived as necessary, complex passwords must be memorized or stored separately, etc.

Satisfaction with the use of the system. Several methods can measure the user satisfaction when working with the system. For this, the structure-oriented evaluation model (SURE model) defines several sub-goals [12,13]. The sub-goals for user satisfaction can be:

- acceptance of user interface;
- design of the visualization (graphical user interface);
- clearness of terminology;
- learning environment.

IV. Classification of UI design oversights

Based on the analyzed aspects of UI design, a classification is proposed for assessing the quality and validation of the user interface. This allows to analyze and categorize user interface oversights, as well as get recommendations for fixing.

The classification is made up of 4 main criteria:

- UI Aspects - a sign of classification "by type"
  - performance;
  - error protection;
  - graphic design and perception, ergonomics;
  - search and visualization;
  - training, help;
  - user needs, level of requirements;
  - screen space as a resource, interactive garbage;
  - UI responsiveness;
  - mental model, subject area;
  - navigation, trajectory, connectivity;
  - context, focus of attention;
  - information content (hierarchy, identification);
  - UI model.
- Frequency of occurrence
  - typical;
  - periodic;
  - separate project.
- Impact
  - disadvantage;
  - difficulty;
  - impossibility of work.
- Design phase
  - business analytics;
  - system analytics (requirements);
  - architectural design;
  - UI design;
  - implementation (code, lack of design patterns).

Interface validation allows to more accurately determine the type of design oversights (based on aspects), the frequency of their occurrence, what consequences are entailed by ignoring the oversight and the design stage of the software product at which the oversight occurred. This classification is not a full-fledged methodology that allows to unambiguously correct oversights in architecture and design at the level of system and business analytics. Using

the classification, it is possible to more accurately determine the nature of the oversight and in the future to foresee its appearance at an early stage of the design.

V. Analysis and Classification of UI Design Oversights

To analyze and classify UI design oversights, it is necessary to validate the user interface of web-applications for BPMS "Fireproof Corporation" company using the developed classification of UI design oversights. Based on the results of validation, it is necessary to form a set of typical UI design oversights with recommendations for correction. Below is the result of the analysis and classification of design oversights based on the experience of developing web-applications.

A. UI oversight: data presentation hierarchy

Figure 2 shows an incorrect option for displaying detailed information about an application - it is not possible to view the "history" of an application, its parent and child applications. Figure 3 shows a revised version of the dialog box with the ability to view the "history" of the order.

Oversight classifications:

- Type: informativeness;
- Impact: difficulty in work;
- Oversight: lack of hierarchy of data presentation;
- Development stage: development project from a business analyst;
- Solution: automatic disclosure of the "history" of the application and the ability to hide the "history".

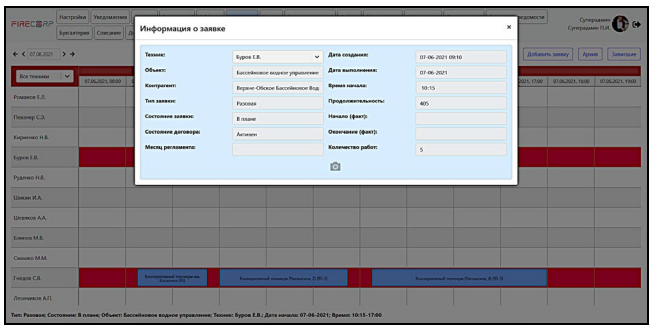


Figure 2: Incorrect design of the dialog box for viewing detailed information about the application

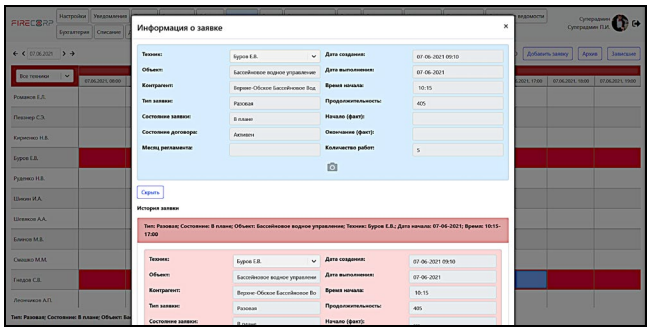


Figure 3: Corrected design of the dialog box for viewing detailed information about the application

B. UI oversight: no graphical editing when interacting with the calendar-scheduler

Figure 4 on the left shows an incorrect design option for a calendar-scheduler from a business analyst document - there is no data on the time and duration of service (1), the presence of shifts (2), the list of unassigned requests (3), there is no possibility of assigning and editing via drag and drop (4). It is also an oversight to move an order across the scrolling screen and display narrow application elements with the impossibility of manipulation. Figure 4 on the right shows a revised version of the calendar-scheduler, which contains a list of unallocated applications, the possibility of assignment by moving the application element for a certain time and "stretching" the application to the required duration using the drag and drop mechanism. The items being moved are close to the target area of movement, the size of the ticket item is large enough for interaction.

Oversight classifications:

- Type: informativeness, performance;
- Frequency of occurrence: specific development (BPMS);
- Impact: difficulty in work;
- Oversight: lack of data on the time and duration of service (1), the presence of shifts (2), the list of unassigned requests (3), assignment and editing via drag and drop (4);
- Development stage: development project from a business analyst;
- Solution: list of unallocated applications, destination (time, duration) - drag and drop.

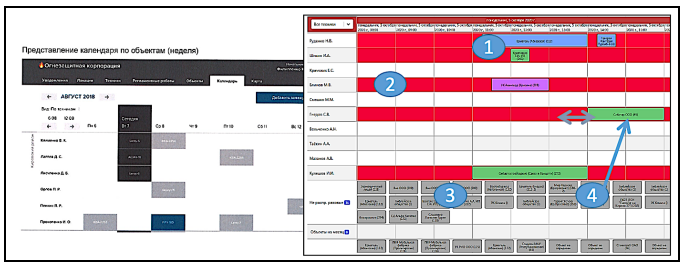


Figure 4: Interaction with the calendar-scheduler

C. UI oversight: screen space efficiency

Figure 5 in the background shows an incorrect design for the object panel from a business analyst document. An oversight is the display of the panel in a tabular form with a limited set of visible data. Figure 5 in the foreground shows a revised version of the object panel - all the necessary data is displayed in one window without a "scroll", filters are implemented by the main business entities related to the object (orders, contracts, contractors, technicians).

Oversight classifications:

- Type: informativeness (1), navigation (2), search (3);
- Impact: difficulty in work;
- Oversight: lack of hierarchy of data presentation, a list of objects in a short form is not needed;

- Development stage: development project from a business analyst;
- Solution: implementation of a search (1), a filter of applications (3), displaying the main characteristics of the entity in question in a row, instead of a table display.

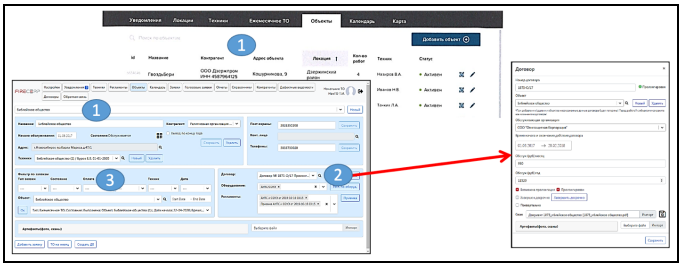


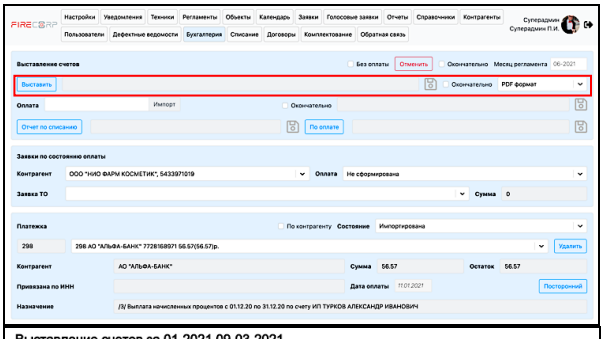
Figure 5: Interaction with the object control panel

D. *UI oversight: no preview of bulk changes*

Figure 6 shows the accounting panel, with the block responsible for the billing process highlighted. This process is a massive change and cannot be incorrect. It is necessary to have a mechanism to switch between preliminary and final billing with an appropriate progress report.

Oversight classifications:

- Type: error protection;
- Impact: difficulty in work, subsequent errors;
- Oversight: the final execution of irreversible changes without the ability to preview;
- Development stage: development project from a business analyst, implementation;
- Solution: before performing an operation with a mass data change, the user must receive a corresponding report, on the basis of which he confirms the changes.



Выставление счетов за 01-2021 09-03-2021

№ п/п	Объект(ы)	Контрагент	ИНН	Сумма	Заявка	Месяц ТО
1	Библейское общество (1)	Религиозная организация	5403123910	960р.	ЕТО	1
2	Марк-2 (магазин "Дефилетто") (2)	ООО Компания «МАРК-2»	5405214666	1030р.	ЕТО	1
3	Бассейновое водное управление (3)	Верхне-Обское Бассейновое	5404161860	1900р.	ЕТО	1
4	ЗПК Западно-Сибирская промышленная	ООО "ЗПК" (Западносибирская	5405381963	2360р.	ЕТО	1
5	Навигатор (14)	ООО "Навигатор"	5404215459	2600р.	ЕТО	1
6	РЭН фирма (15)	ООО Фирма «РЭН»	5403156602	500р.	ЕТО	1
7	Поднебесное ООО (16)	ООО "Поднебесное"	5405345972	12000р.	ЕТО	1
8	Запсибкола (17)	ООО "ПО Запсибкола"	5403148591	6350р.	ЕТО	1
9	Попов М.Т. ИП (18)	ИП Попов Михаил Тимофеевич	550200004801	1200р.	ЕТО	1
10	Вернисаж ООО (19)	ООО "Вернисаж"	5407003156	700р.	ЕТО	1
11	Сибирские ресурсы ООО (20)	ОАО «Сибирские ресурсы»	5406102933	900р.	ЕТО	1
12	Контур ТД ООО (21)	ООО Торговый дом «КОНТУР»	5401278845	3060р.	ЕТО	1
13	СТиЛ Компания ООО (22)	ООО Компания «СТиЛ»	5404455605	1260р.	ЕТО	1
14	ЖЭУ №4 (23)	ООО "ЖЭУ №4"	5405018132	1148р.	ЕТО	1
15	Лазурит ТД ООО (24)	ООО Торговый дом «Лазурит»	3917032714	7680р.	ЕТО	1

Figure 6: Billing process

VI. Results and Conclusion

The developed classification is not a full-fledged methodology that allows to unambiguously correct errors in architecture, design at the level of system and business analytics. But when assessing the quality of the user interface, the methodology is not needed as a tool. The validation of the application interface is performed in accordance with the selected aspects and the typical design oversights corresponding to the aspects. The article discusses several typical design oversights based on experience in developing web applications. For a wider use of the classification, it is necessary to replenish the set of typical UI design mistakes.

References

[1] E.L.Romanov, “Software engineering”: study guide / E.L.Romanov. - Novosibirsk : NSTU publishing, 2017. - 407 p. - (NSTU textbooks). - ISBN 978-5-7782-3455-0.

[2] E.L.Romanov, “Client-server application framework based on an object-oriented network model” / E.L.Romanov, G.V.Troshina, S.A.Menzhulin // IOP Conference Series: Materials Science and Engineering. - 2020. - Vol. 919, № 5 : Advances in Material Science and Technology : intern. sci. conf. CAMSTech-2020, Krasnoyarsk. - Art. 052014 (9 p.) - DOI: 10.1088/1757-899X/919/5/052014.

[3] E.L.Romanov, “Software engineering. Beyond the intuitive interface”. Accessed: April 19. 2021. Available online at https://dispace.edu.nstu.ru/didesk/course/show/5164/3

[4] E.L.Romanov, “Software engineering. Unified Process - Work Tool or Unattainable Ideal”. Accessed: May 3. 2021. Available online at https://dispace.edu.nstu.ru/didesk/course/show/5164/3

[5] N.V.Gervas, “Analysis of user interface design methods” [Electronic resource] / N.V.Gervas ; [sci. ed. E.L.Romanov] // International symposium on computer science, computer engineering and educational technologies (ISCSET-2020), Mongolia, Ulaanbaatar, 21–23 Oct. 2020. – Chemnitz : TUDpress, 2020. – P. 57-60. – (IBS Scientific Workshop Proceedings ; bd.10). - Mode of access: http://www.mier.mn/iscest/Proceedings.html. - Title from screen - ISBN 978-3-95908-223-5.

[6] V.V.Golovach, “User interface design”, 2000, vol.1.2 pp. 89-104.

[7] S.Krug, “Don't make me think. Web usability and common sense”, Eksmo, 2014, pp.45-67.

[8] T.P.Brusentsova, “User interface design”, manual for students of specialty 1-47 01 02 "Design of electronic and web publications", 2019, pp. 103 - ISBN 978-985-530-799-1.

[9] E.Volchenkov, “User interface standartization”. Accessed: May 9. 2021. Available online at https://www.osp.ru/os/2002/04/181312

[10] A.Anatolyev, “Design of human-machine interfaces”. Accessed: June 2. 2021. Available online at http://www.4stud.info/user-interfaces/

[11] Qubstudio, Design Agency, “How to Design Products with Goal-Centered Design”. Accessed: June 4. 2021. Available online at https://qubstudio.com/blog/how-to-design-useful-products-with-goal-centered-design/

[12] E.Norbert, J.Tonndorf-Martini, A.Heller, L.Gaitzsch, U.Tudevdagva, W.Hardt. "Adaptive learning system in automotive software engineering." In 2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), pp. 1-5. IEEE, 2019.

[13] U.Tudevdagva, “Structure-oriented evaluation: An Evaluation Approach for Complex Processes and Systems”. SPRINGER, 2020.