ЯКІСТЬ ПРОГРАМНОГО ЗАБЕЗПЕЧЕННЯ

UDC 004.055 + 004.5

Guchenko I.V.
National Aviation University

USABILITY MANAGEMENT IN THE CONTEXT OF SOFTWARE ARCHITECTURE

The article is devoted to the issue of relationship between usability and software architecture. Architectural solutions that have influence on overall software usability through concrete properties and attributesare analyzed using the concept of usability patterns. The last one is appliedto the earlier developed author's usability management method considering the usability model based on the latest standards. Conclusion is made that usability improvement in the context of the method of software usability management should be started from the design stage of the software lifecycle. Design solutions which have positive effect on particular usability property are defined.

Стаття присвячена питанню зв'язку зручності використання програмного забезпечення та його архітектури. Архітектурні рішення, що мають вплив на загальну зручність використання програмного забезпечення через конкретні характеристики та властивості, проаналізовані на основі концепції шаблонів зручності використання. Концепція застосована до раніше розробленого автором методу управління зручністю використання програмного забезпечення, враховуючи модель, засновану на останніх стандартах в даній області. Зроблено висновок, що поліпшення зручності використання в контексті згаданого методу слід починати з етапу проектування програмного забезпечення. Визначено архітектурні рішення, які позитивно впливають на конкретні властивості зручності використання програмного забезпечення.

Статья посвящена вопросу связиудобства использования программного обеспечения и его архитектуры. Архитектурные решения, имеющие влияние на общее удобство использования программного обеспечения через конкретные характеристики исвойства, проанализированына основе концепции шаблонов удобства использования. Концепция применена к ранее разработанному автором методу управления удобством использования программного обеспечения, учитывая модель, основанную на последних стандартах в данной области. Сделан вывод, что улучшение удобства использования в контексте упомянутого метода следует начинать с этапа проектирования программного обеспечения. Определены архитектурные решения, позитивно влияющиенаконкретные свойстваудобства использования программного обеспечения.

Keywords: *softwareusability, usability model, usability patterns, usability management, usability improvement, software architecture, software design, architectural solutions.*

Introduction

Achieving better usability through software architecture is not a new goal. In 1980 sandearly 1990s there was an assumption that usability is a property of presentation of information. Thus, separating presentation from application made it easier to modify presentation after achieving user feedback. Such assumption was wrong for developing usable systems. In later 1990s getting the correct functionality as well as presentation for good usability became the new emphasis. Nevertheless, evenin that case system usability can be greatly compromised if the underlying architecture does not support human concerns beyond modifiability. Still nowadays, many software products suffer from usability issues that cannot be repaired without major changes to the software architecture. A large amount of maintenance costs are spent on dealing with usability problems [1], which are usually detected only during testing and deployment rather than during design and implementation. These high costs prevent developers from meeting all the usability requirements, resulting in systems with less than optimal usability. Explicit evaluation of usability during architectural design may reduce the risk of building a system that fails to meet its usability requirements. Also high cost of adaptive maintenance can be prevented. From this perspective it is important to establish architectural solutions that have influence on overall software usability through concrete properties and attributes.

Literature analysis

In existing scientific works relationship between usability and software architecture is connected with the concept of a usability pattern. *Usability pattern* is a technique or mechanism that can be applied to the design of the architecture of a software system in order to address a need

identified by a usability property at the requirements stage [2].

The collection of twenty usability patterns has been defined in [3]. The important aspects of the patterns are derived from the representing usability as three-layered model. The highest level – ISO 9126 subcharacteristics of usability. The next level contains a number of usage indicators which are indicators of the usability level that can actually be observed in practice when users are at work. Each of this indicators contributes to the abstract subcharacteristics of the higher level. The lower level is the level of means which are used in heuristics for improving one or more of the usage indicators. It is said that usability pattern should state the impact on the user indicators. The structure of a pattern is the following: problem, usability principle, context, forces, solution, rationale, example, known uses and related patterns. The patterns are task related and categorized according to the kind of usage problems they address: visibility, affordance, natural mapping, constraints, conceptual models, feedback, safety, flexibility.

Folmer and Bosch [1] also used a top down approach from the usability definition to usability patterns. The usability framework consists of attributes, properties and patterns. There is not one-to-one mapping between the usability patterns and the usability properties that they affect. The research is on the ground of four most commonly used by different authors usability attributes: learnability, efficiency, reliability and satisfaction. The corresponding properties are: providing feedback, error management, consistency, guidance, minimize cognitive load, natural mapping and accessibility. The patterns collection is different from the Welie's because the authors only considered fifteen patterns which should be applied during the design of a system's software architecture, rather than during the detailed design stage.

In [4] the relationship between the usability and software architecture has been investigated through the definition of a 26 scenarios which are in some way equivalent to properties and patterns in [2]. *Usability scenario*is defined as description of an interaction that some stakeholder has with the system under consideration from a usability point of view. An architectural pattern for each of the general usability scenarios has graphical representation and verbal components' description.

Grounding

Previous author's works are devoted to the development of the method and the tool of software product usability management [5].It supports usability management based on the automated evaluation of users' feedback.The principalfeatureofthismethodisthatnotonlyusability evaluation, butalso usability management is considered in theprocessofsoftware creating. It is achievedbytheautomatedconstructionof variant of providing a givenusability level during next iteration.The optimalwayof such providing is basedon mathematical models of software product usabilityevaluation and assurance,which are focused on usage of customers' feedback.

There are important questions about the stages if software lifecycle, where the recommendations if usability properties improvement should be implemented, and about impact of such recommendations on workproducts. Usability properties are relatedtosoftware architecture and can be considered within the concept if usability patterns when applying the proposed method.

Described above usability patterns receaches are out of date in the sense of used usability definitions and subcharacteristics as they are grounded on the old standards.

The aim of the present article is to apply usability patterns concept to the author's usability management method considering the latest information about usability, particularly from ISO / IEC 25010:2011 (updated ISO / IEC 9126-1:2001) [6].

Case study

In many studies attempts to determine the usability are made, but often they are in consistent [1]. Therefore, we will use the definition given in the standards ISO 9241-11 [7] and ISO / IEC 25010:2011 (updated ISO / IEC 9126-1:2001) [6]:

Usability – degree to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use.

The method of software usability management is based on the iterative evaluation of the current usability level during software creation and on the formation of optimal variant of achieving the established usability level, which is set by the developer at the beginning. Iterative usability estimation,while using the method in iterative development methodology,should be understood as being performed at each iteration, i.e., the completed cycle of development that leads to product release or version. For non-iterative development methodologiesiterative usability evaluation means its occurrence (repetition) in the management process.

The solution of the usability management problem according to the process approach [8] contains the following steps:

1. Construction of the usability hierarchical structure by experts. Includes development of metrics by top-down structural method [9] and contains the following levels:

a) top level – usability subcharacteristics. Choosing of the subcharacteristics is performed on the basis of the existing usability requirements using industry standards, own base of historical data about usability of the earlier created software products and on the ground of information about users' expectations. Priorities and interconnections between attributes and requirements are establishing. Also allowable ranges for numeric attribute values should be set with the help of managers and / or customer;

b) middle level – usability properties. Decomposition of usability subcharacteristics in calculated properties is performed;

c) lower level – usability measures. Decomposition of usability properties in measures is performed.Measures can be directly estimated in numerical form by users while using software product.

2. Calculation of usability properties' valueson the basis of metrics' values derived from users' estimates.

3. Construction of the mathematical model for usability evaluation, which allows, according to the hierarchical model, to reduce the individual values of usability properties derived from users' ratings and experts' rankings into a single numerical value. If the obtainedusability level is equal or more than specified, the report is formed, otherwise it is necessary to go to p. 4.

4. Construction of the mathematical modelfor usability assurance. The mathematical model of usability evaluation is supplementedby function of labor ofusability properties changing, thus the model of optimal assurance of established usability level is obtained.

5. Formation of the optimal variantof providing a given usability level. The result is represented as a set of properties that need improvement (including the change value for each indicator). To determine the effect of changing parameters on the software product usabilityit is proposed to establish the existence and the form of relation between pairs of properties under consideration.

6. Implementation of the obtained variant of properties' changes and control of achieving the established usability level during the next iteration, if necessary – correction of the models.

Implementation of changes for improving usability can be started from the design stage using usability patterns. It is important to define the usability model. In the method above the hierarchical structure was chosen.To clarify this model the latest information about usability subcharacteristicswas used.

In ISO / IEC 25010:2011 [6], which belongs to a series if standards SQuaRE(ISO / IEC 25000 - ISO / IEC 25099), usability is considered in two models: directly – i the product quality model; indirectly – i quality i use model. According to the first model usability has six subcharacteristics: appropriateness recognisability, learnability, operability, user error protection, user interfaces esthetics and accessibility. They form the basis for the specification of usability requirements and its evaluation. Sets of software properties correspond to subcharacteristics. List if properties was developed using QUIM model [11]. These properties match measures [8]. With regard to the measures that are calculated for each usability property, the corresponding list is presented in [8]. Measures are calculated using formulas for simple calculations on the ground of users' feedback (ratings).

Using Folmer and Bosh approach [1, 2] the usability framework was developed. It consists of subcharacteristics, properties and patterns. There is not one-to-one mapping between the usability patterns and the usability properties that they affect. There are twenty usability properties in the author's usability model [8]. List of the patterns and their relations with the usability properties is grounded on the Folmer and Welie works. Graphical representation of the framefork is on the fig. Explanations are given below.

There is not necessary only one method to implement the solution presented in usability pattern. Patterns don't specify implementation details in terms of classes and objects. The main fields in describing patterns are *problem*, *usability context*, *rationale* and *solution* (or architectural implications). Solutions presented in usability pattern can be realized with different architectural and design patterns. For example, Undo may be implemented by Memento design pattern and Multiple views – by MVC architectural pattern etc. It is important to remember that pattern optimizes several usability properties while other properties become worse.

Time behavior, Attractiveness and Likeability

have no analogical usability properties in related works [1-4], but in [3] the rationale for each pattern is created considering such usability aspects as Performance speed and Satisfaction.

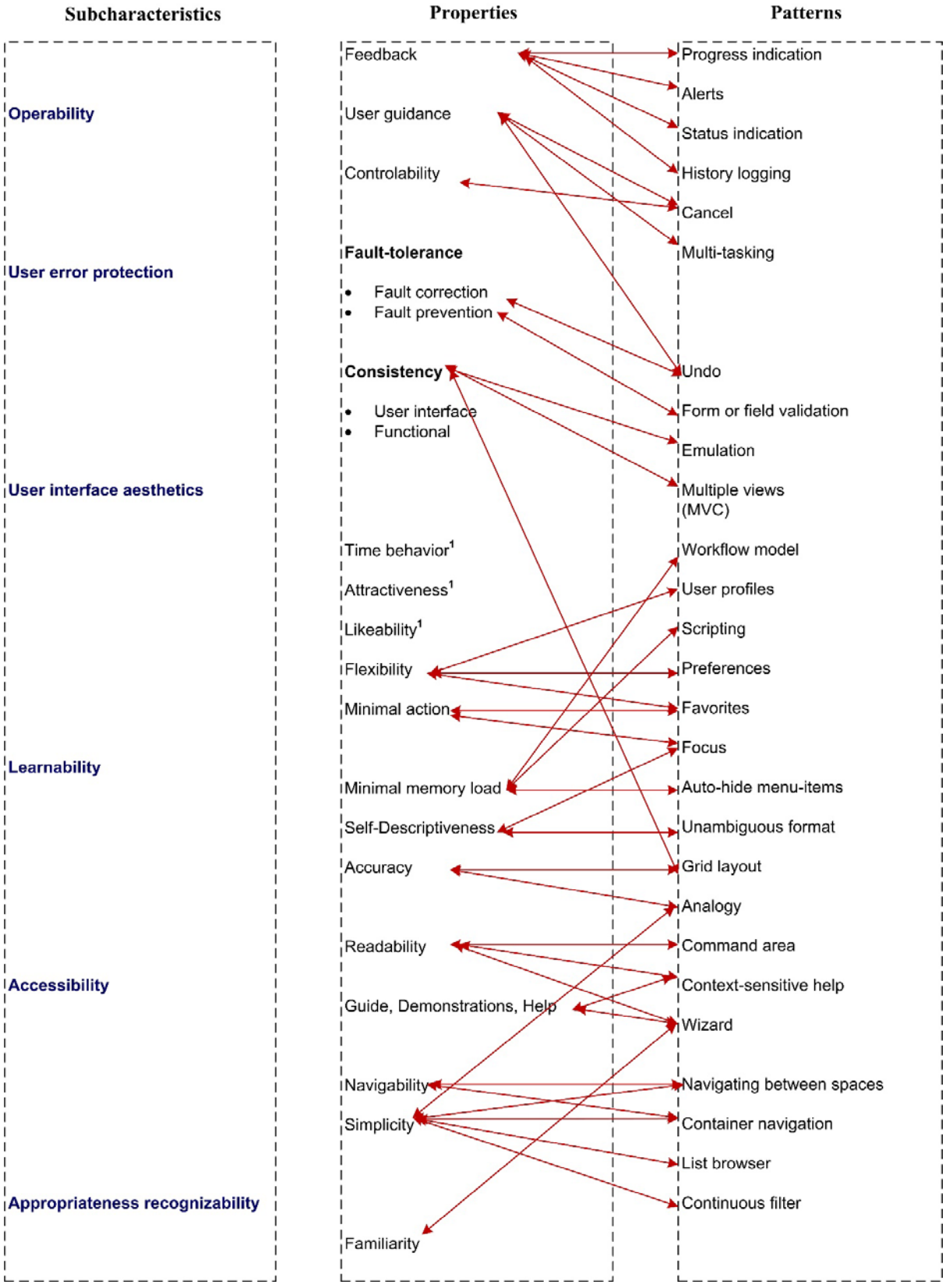

Fig. Connection between usability and software architecture[1]

[1] Connections between properties Time behavior, Attractiveness, Likeability and usability patterns are explained within the body of the article. Relations between usability subcharacteristics and properties are presented in the Annex A.

After patterns [3] analysis it was defined that Time behavior property is affected positively by the following usability patterns:

– *Grid layout*: arranging all objects in a grid using the minimal number of rows and columns, making the cells as large as possible. As a result, the time needed to read the information and task completition time are reduced;

– *Preferences*: providing choices (for example, in a form of dialog box) for the user which will become the default. Tweaking the application for the particular purposes increases possible performance;

– *Focus* (object the user is working on): determines the context of the available functionality. Windows containing relevant functionality are activated when the focus changes. This reduces time of function execution because less actions are needed;

– *Navigating between spaces*: grouping of elements in separate labelled spaces and allowing the user to select only one space in a time. Reduces time for searching an element;

– *Analogy*: using real world metaphors;

– *Favorites*: searching time is reduced by using favorites menu;

– *List browser*: allows the user to navigate directly from one item to the next and back. User does not need to go back to the index and reduces task time;

– *Continuous filter:* component allows user filter in real time only the items that are of his interest. User gets immediate result corresponding the search term.

Attractiveness and Likeability also are supported by the patterns listed above. Additional usability patterns which increases these properties are: Progress and Status indication, Context-sensitive help, Unumbiguous format (allows user to enter data in the correct syntax) and Command area.

Relations between software usability and architecture show that usability improvement in the context of the method of software usability management should be started from the design stage of the software lifecycle. Architecture updating obviously affects the work products of the following stages. Also it can lead to necessity of requirements redefining, thus introduction of changes to the initial stage – requirementsanalysis. In this case, the cost of work performed to achieve a given usability level will be the greatest.

Conclusions

Designing usable software products is difficult and developers need effective methods. Earlier author's works were devoted to the creating of the method of sodtware usability management during development. Current research shows that usability properties are related to software architecture and can be consideredwithin the concept of usability patterns when applying the proposed method.Existing usability patterns researches are out of date in the sense of used usability definitions and subcharacteristics as they are grounded on the old standards thus old usability models are used. In the article usability patterns concept is applied to the author's usability management method considering the latest information about usability, particularly from ISO / IEC 25010:2011 (updated ISO / IEC 9126-1:2001). As a result the method is clarified in the sense of ways of changes' implementationforimprovingusabilityat the design stage. Design solutions for each particular usability property are defined. The future work will be devoted to the analysis of concrete design and architectural patterns which have a positive effect on the usability.

References

1. Folmer E. Experiences with Software Architecture Analysis of Usability / E. Folmer, J. Bosch // International Journal of Information Technology and Web Engineering. – 2008. – Vol. 3(4). – P. 1 – 29.

2. Folmer E. UsabilityPatternsinSoftwareArchitecture / E. Folmer, J. Bosch // HCII'2003: proceedings. – 2003. – P. 93 – 97.

3. Welie M. Interaction Patterns in User Interfaces / M. Welie, H. Traetteberg // PloP'2000: proceedings. – 2000. – P. 113 – 138.

4. Bass L. Achieving Usability Through Software Architecture / Bass L., John B., Kates J. – Pittsburg, PA: Carnegie Mellon SEI, 2001. – 103 p.

5. Гученко І.В. Метод і засіб управління зручністю використання програмних продуктів: дис. ... кандидата технічних наук: 01.05.03 / Гученко Інна Володимирівна. – К., 2012. – 124 с.

6. Systems and software engineering, Systems and software Quality Requirements and Evaluation (SQuaRE), System and software quality models: ISO/IEC 25010:2011. – Geneva: International Organization for Standardization /International Electrotechnical Commission, 2011. – 34p.

7. Ergonomic Requirements for Office Work with Visual Display Terminals (VDTs), Part 11: Guidance on Usability: ISO 9241-11. – Geneva: International Organization for Standardization, 1998. – 22p.

8. Руководство к своду знаний по управлению проектами (руководство PMBOK); пер. сангл. – [4-еизд] – Pennsylvania: Project Management Institute, 2010. – 463 с.

9. IEEE Standard for a Software Quality Metrics Metodology: IEEE Std. 1061 – 1998. – N.Y.: The Institute of Electrical and Electronics Engineers, 1998. – 38 p.

10. Padda Harkirat K. QUIM: A Model for Usability/Quality in use Measurement / Harkirat K. Padda. – Colne: Lambert Academic Publishing, 2010. – 124 p.

Information about author:

Guchenko Inna Volodymyrivna – PhD, Associated Professor of the Software Engineering Department of the National Aviation University. Scientific interests: software engineering.

E-mail: Inna.Guchenko@livenau.net