Uniwersytet Ekonomiczny w Katowicach

Studia Ekonomiczne. Zeszyty Naukowe
Uniwersytetu Ekonomicznego w Katowicach
ISSN 2083-8611 Nr 234 · 2015

Paweł Weichbroth

Politechnika Gdańska
Wydział Zarządzania i Ekonomii
Katedra Zastosowań Informatyki w Zarządzaniu
pawel.weichbroth@zie.pg.gda.pl

Marcin Sikorski

Politechnika Gdańska
Wydział Zarządzania i Ekonomii
Katedra Zastosowań Informatyki w Zarządzaniu
marcin.sikorski@zie.pg.gda.pl

USER INTERFACE PROTOTYPING. TECHNIQUES, METHODS AND TOOLS

Summary: Currently, for interactive applications their usability and ergonomics of the user interface (UI) are critical factors for achieving users' acceptance. Thus, from the perspective of the end-user, success of the entire software development project depends on the quality of the interaction between user and system. Therefore, it is necessary to involve project stakeholders, providing them with the necessary communication tools for the exchange and codification of knowledge. The need to use this kind of tools is especially visible in the software life cycle model, known as prototyping. Based on the literature review and previously conducted research, the authors review, analyse and evaluate available techniques, methods and tools, which support the process of UI prototyping. In this paper, the authors also present a novel model of the UI prototyping process. The main assumption of this model is interactive development of the system prototype and its user interface components, and iterative evaluation by end-users. It is a user-oriented approach in order to ensure that not only expected system functionality will be delivered, but also optimal usability and ergonomics.

Keywords: user interface, prototyping.

Introduction

Beyond any doubt, nowadays computer software is in common use in almost every area of human activity and in many professional domains. The actual usage of the software very much depends on the quality (usability and ergonomics) of its *user interface* (UI), which provides user's input usually by keyboard, mouse or by touch screen. Because users' experience, knowledge and skills to learn how to use the system are never the same, many contextual factors must be included in order to plan and develop appropriate usability characteristics.

Therefore, it is not a straightforward task to design a good UI, which should fulfil eight domain-independent properties such as: clear, concise, familiar, responsive, consistent, attractive, efficient and forgiving [Fadeyev, 2009]. Otherwise, poorly designed UI can lead to difficult interaction, user frustration and poor work performance. Usability flaws are difficult to repair, especially those detected in the final stages of the project or after the system has been already put into operation. In some cases, due to intensive users' complaints an organization can look for alternative software solutions, purchase and deploy a new product, when observable and significant changes in UI cannot be made in an existing system.

The significance of prototyping in the user interface development process should be highlighted for the following reasons:

- UI prototype is an operational model of developed system, which shows not only interaction between a user and a system, but also exposes some subset of its functionality;
- UI prototype is a solid foundation for discussion between project stakeholders (designers and users), giving opportunities to learn more about product features and design [Sikorski, 2012, p. 96];
- UI prototype usability testing enables to thoroughly validate user formal requirements and informal expectations in regard to developed system.

In this paper, in the next section, we discuss related research and several main approaches relevant to UI prototyping. In the second part of this paper we present a novel approach to model a process of user interface prototyping, preceded by some basic definitions, important to fully understand presented model and its specific characteristics. Lastly, we present our plans for further research relevant to studying this model in IT projects practice.

1. Related research

1.1. Rapid prototyping

User interface prototyping takes its origin from a rapid prototyping approach, which for some time has been used in industrial projects wherever there was lack of computational models for predicting a specific behaviour of particular technical components [Chua et al., 2010]. The power of rapid prototyping has been recently strengthened by broad availability of low-cost 3D-printers, which are widely used in hardware manufacturing for a fast manufacturing of prototype mechanical parts.

Unfortunately, in human-computer interaction domain there exist no reliable methods for automatic generation of user interfaces. For this reason rapid prototyping approach – at least in a strict technical meaning – does not apply to developing user interfaces. As a result, user interface prototyping, described in the next section remains mainly as a procedural approach, in contrary to rapid prototyping, not much supported by relevant manufacturing infrastructure.

1.2. User interface prototyping

User interface prototyping is a testing and evaluation approach, which is a crucial component of User-Centred Design (UCD) methodology, widely used in IT projects since the 1990s [Sharp, 2005]. Involving users in evaluating prototypes is an important part of all iterative approaches for IT projects management, and of *agile methodologies* [Schwaber, 2004] in particular. Originating from User-Centred Design, prototyping has also become a popular method for user-based validating design concepts in service design and development [Stickdorn, Schneider, 2010].

In contemporary IT projects user interface prototyping offers following benefits:

- identification and preliminary validation of user requirements already in early stages of the project;
- improved customer/user' attitude by showing that "*something is already running*", what develops user's "*sense of ownership*" for approved solutions;
- when prototyping is frequent, usually the greater frequency of contact with the customer, the less usability flaws and fewer corrections needed at the end of work.

User interface prototyping is generally considered as an excellent approach for facilitating communication between the designers and other stakeholders [Dix et al., 2004; Snyder, 2003]. Prototyping not only helps to visualize design concepts in an interactive way, but also supports expressing new requirements and expectations towards a prospective system. Usability testing of user interface prototypes validates user requirements and functionality of the prototype under evaluation.

Before the actual prototype testing takes place in the project, following two techniques are necessary for identifying required functionality of the prospective system:

1. **Drawing design techniques** – useful for expressing design concepts in the form of UML or BPMN diagrams or as a free-hand sketches and drawings like *Rich Picture* [Gawin, Marcinkowski, 2013].
2. **Context of use analysis** – a structured description of user characteristics, task and organizational environment [Dix et al., 2004].

Prototyping in User-Centred Design (UCD) approach usually is based on evaluation of:

1. **Low-fidelity (paper) prototypes**, aimed to communicate general concept of the prospective system:
   - *storyboards* show in a hand-drawn cartoon-like style the general sequence of screens the user follows a network of specific operations [Snyder, 2003];
   - *paper prototypes* present the general layout of screen elements; they are usually "*flat*" as two-dimensional prototypes laid on the surface of the table; they can be used for on-hand optimizing location of user interface elements and for and gathering first evaluations from prospective users or customers [Snyder, 2003];
   - *service prototypes* (see also section 1.3. below) enable general visualizing how the designed service will work in its spatial environment, who will be involved, which mobile applications will be used and what sort of User Experience will be created at subsequent stages of the service process [Stickdorn, Schneider, 2010].
2. **Interactive prototypes**, which simulate operating an actual system on a computer screen; such prototypes are prepared with a specific software prototyping tools and they may be easily used for usability testing and for gathering new requirements from prospective users [Warfel, 2009]. The list of software tools available for creating interactive prototypes has been presented for instance by [Szekely, 1994] or by [Walker et. al., 2002]. Currently available user interface prototyping tools vary form general ones (like Microsoft PowerPoint or HTML to dedicated professional applications like Axure or GUI Design Studio).

Examples of low-fidelity and interactive user interface prototypes for a fast-food ordering by a touch-screen device are shown in Fig. 1 and Fig. 2, respectively.

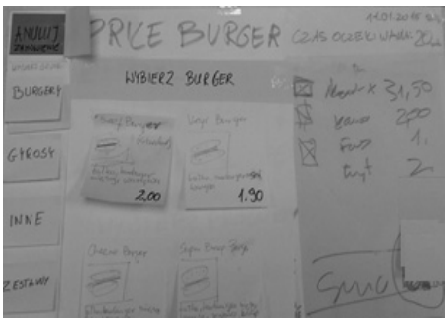

Fig. 1. A low-fidelity prototype of an touch-screen kiosk

Fig. 2. An interactive prototype of an touch-screen kiosk

Both low-fidelity and interactive prototypes are an important part of contemporary IT projects management methodology [Sikorski, 2014]. Early and frequent prototyping of user interface is not only the fundamental mean of usability assurance but it also facilitates communication with a prospective users/customers, especially in agile project management.

In notably customer-oriented e-business solutions, interaction is driven by discovered knowledge from particular datasets in such applications like: web usage mining [Owoc, Weichbroth, 2014], market basket analysis [Kubiak, Weichbroth, 2010] and social network analysis [Vervest et al., 2005].

1.3. Service prototyping (Design Thinking)

As mentioned in the previous section, service prototypes are usually quick-and-dirty prototypes made of paper and other stationery materials, intended for visualizing how the designed service will work, and who will be involved into subsequent stages of a specific service process [Stickdorn, Schneider, 2010].

Service prototypes are usually built as 3D structures placed of a flat surface (of a table, for instance). Moreover, Lego characters often simulate behaviour of customers, service vendors or other stakeholders in specific scenarios, which reflect actual situations which may happen during service operation. In service prototyping the use of mobile apps (being a part of the service process and operated by the user/consumer) can be also included, adding new insights into service design and evaluation.

Service prototypes are often evaluated by prospective customers, and subsequently redesigned while prospective customers are active members of the design team across the whole timespan of a specific project. Very intensive user involvement is a characteristic feature of the Design Thinking approach [Stickdorn, Schneider, 2010], which expands the role of the user/customer from being merely a tester (or an adjunct usability consultant) to being an active co-designer – who is not professionally trained, but brings his/her valuable experience and expectations expressed towards a particular interactive system or service under development.

Service prototyping is a natural extension of early techniques used for service process modelling, such as flowcharts, service blueprints or interaction diagrams [Sikorski, 2012]. In contrary to early service modelling techniques, service prototyping is a form of collaborative design, where service designers and prospective customers work together during the whole design process. Similarly however, as in the case of user interface prototyping, service prototyping facilitates communication within the design team and brings new insights into current design problems.

In case when customers expect some "intelligence" from service prototype we can explicitly place a list of recommendation items, which represent other complementary or substitutive services [Kubiak, Weichbroth, 2010].

1.4. User involvement in prototyping

In the above sections we presented an evolutionary development covering various forms of prototyping related to developing interactive systems in IT projects. Subsequent evolvement of end-user participation in prototyping have been briefly presented. Detailed discussion how the use of prototyping changes the role of the user in contemporary IT projects was presented by [Sikorski, 2013], who revealed the changing role of end-users in cooperation with IT design teams in recent years. As a result of this study where several design perspectives have been discussed, with *Design Thinking* (aka *Service Design*), was found as the most mature, regarding the use of prototyping for developing interactive IT products or services.

All abovementioned prototyping perspectives can be applied at different stages of the same IT project. They are offering all complementary types of outcomes but for achieving efficient use of prototyping usability expertise in needed in addition to good communication skills with project stakeholders. Moreover, the general deficiency of all presented prototyping approaches is that they need to be adapted to the context of use of a prospective system and to specific project management context.

As a result, there is an ongoing need for developing a user interface prototyping approach, which would be context-independent and which would smoothly integrate with current software engineering and IT project management practices.

2. A novel process for user interface prototyping

Firstly, let us define some basic concepts, terms and methods to clarify, methodize and systematize our approach, later illustrated and discussed in details below.

2.1. Definitions

User requirements specification is a documentation, which defines computer system (application) functionality, completely independent of any solution-oriented bias or software vendor. It must be understandable by the project stakeholders (users, developers, designers and project managers).

Data flow diagram (DFD) reveal relationships among and between the four major system components: external entities, processes, data stores and data flows between these [Renolen, 2000, p. 25], giving the functional perspective focuses on processes rather than on objects and physical entities.

An *entity* is the source or destination of data, while the former provides data from data stores (tables, views, files) and the latter receives and loads data into data stores or create in-memory datasets.

A *process* is a defined sequence of transforming data (e.g. cleaning, distributing, merging, summarizing, updating), performing computations, making decisions (logic flow) or directing data flows based on set of rules (written as: *if* <condition> *then* <consequence>). In other words, the process receives input, executes some algorithm (rules) and returns output.

Data store represents the storage (files, tables) of persistent data required and (or) produced by the process for later retrieval by the same process or another one.

Data flow represents the directed path of data, acts as an interface between DFD components and can include a single or multiple sources (destinations).

User interface schema, often defined also as *user interface diagram*, is a visual presentation of components (e.g. forms, controls, labels), arranged, selected and pinpointed in precise and persistent localization to describe its appearance.

User interface *standards* contains common, problem-solving and component-oriented guidelines for designers and users community.

A *use scenario* (US) is a sequence of actions, conducted to accomplish intended goals. In other words, the use scenario is a one possible combination of the *use case* (UC) and might be presented in the form of simple narrative description, consecutively attached to elements of data flow diagram. DFDs should reflect a complete set of use scenario combinations and ought to accommodate all possible intra-relationships between entities and data stores inside one distinct process and inter-relationships between dependent processes, external data stores and entities.

2.2. The process of user interface prototyping

The process of designing user interface consists of five iterative steps [Schneiderman et al., 2009], however they are not strictly conceptualized in a fixed sequence. Let us now briefly recapitulate each step.

1. In first step, we define, design and evaluate relevant data flow diagrams (DFD). Next, we specify and assign use scenarios to them, in cooperation with end-users.
2. The result of the second step is the user interface (UI) *structure* patterns visualized as diagrams, which defines its primary components and occurring relationships between them.
3. In the third step, we define UI *standards* in three different categories such as: navigation, structure patterns, reports and documentation [Schumaker, 2001].
4. The artefact of the fourth step exhibits UI *prototype* – a visual combination of elements, previously defined and designed.
5. Finally, the evaluation of UI ergonomics and usability is conducted, and when assessment artefact (shown in table 1):
   - allows to approve UI, then developers team can implement and plug in system's functionality to relevant UI components, or
   - does not allow to approve UI, then its supplementary modifications shall be done to refine given prototype, validating users requirements, and when necessary, redefine artefacts (products) of first three steps.

Figure 3 graphically depicts an idea, which lays beneath our model. There is always a *moderator*, a person who plays a central role in the successful development of user interface. It is a moderator who sets the context, drives the discussion in the right direction and engages the participants in an interactive dialogue.

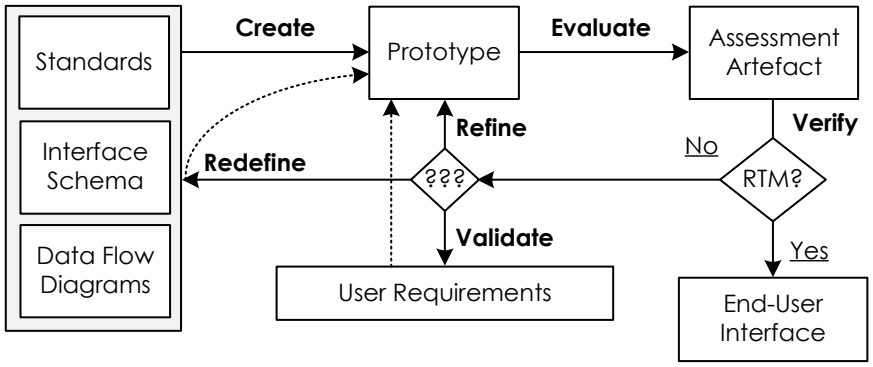

Fig. 3. The process of user interface prototyping

On the left side, we mark three different groups of artefacts (data flow diagrams, interface schema and standards), respectively as the results of first three steps (from bottom to the top). As a set, they serve as an input to create a first prototype of a user interface, which in the next step is evaluated by project

stakeholders, using some *assessment artefact* or other predefined technique. After that, we have to answer a question: *does proposed prototype is ready to manufacture*? Based on the analysis of assessment artefact, only if there is a full agreement on user interface criteria fulfilment, then it can be released to developers team, as a final (ready to manufacture, RTM) version. Otherwise, user interface remains prototype and we have to answer to three major questions: (1) *what should we exactly refine in proposed prototype*? (2) *which artefacts do we have to redefine*? and (3) *do we have to validate users requirements with our customer*? Dotted line represents interactions occurring, when relevant modifications take place in particular artefacts.

Besides, it is also important to engage coherent and definite evaluation methodology. In our model, we adapt related methodology elaborated by Sikorski [2010, p. 77-78], who outlined such in six subsequent steps:

1. Select objects to evaluate.
2. Define a set of evaluation criteria.
3. Define relevant scale for each score.
4. Evaluate each criterion.
5. Calculate generic score measure.
6. Analysis and interpret obtained results.

On the right side of the figure 1, we placed an *assessment artefact*, which should be seen as some abstraction – a result of project stakeholders UI evaluation. There are no constraints and obstacles to use any relevant method or technique, unless it gives outcome leading to further UI development and improvement. One of such technique, in author's opinion, is an *evaluation matrix*, which consists of eight rows and two columns. Each row represents one UI property given by [Fadeyev, 2009], whereas first and second column represent a score and a comment respectively, given by a user to individual property (Table 1).

Table 1. Evaluation matrix template

Property	Score	Comments
Clear	<1, 9> | *or* {yes | no}	text
Concise	<1, 9> | *or* {yes | no}	text
…	…	…
Forgiving	<1, 9> | *or* {yes | no}	text

We mark that a score can be represented as a number from the point scale (from 1 to 9), as well as dichotomous verbal variable (*yes* or *no*). However, in common practice the latter approach might be insufficient [Sikorski, 2010,

p. 77]. Obviously, there is always a users' focus group which members participate in prototype evaluation and in return a set of evaluation matrix is an outcome. At the moment, we affirm only general eight properties, not assigning sub-properties to any of them. To sum up obtained results, we can use an *overall scoring matrix* (Table 2) to show average and standard deviation of given scores, which ultimately illustrates users impression and opinion on particular user interface prototype.

Table 2. Overall scoring matrix (OSM) template

Property	Average	Standard deviation
Clear	$\bar{y}_1$	∂_1
Concise	$\bar{y}_2$	∂_2
…	…	…
Forgiving	$\bar{y}_8$	∂_8

Advantages of above technique are straightforward. Firstly, it is evidently easy to prepare, document and perform even for inexperienced moderator, who does not have expertise in such projects. Secondly, interpretation of those two simple statistics is not difficult (requires only a common sense), representing *average* and *variability of users scores*. Certainly, each subsequent prototype should move toward higher average and smaller standard deviation, when comparing to preceding one. Finally, comments appear as *design hints*, during discussion and should not be omitted or ignored. An OSM collection shows an acceptance tendency in time, which might be an answer to this upstanding question: *are we going in the right direction with our user interface*?

Without question, there are some evident constraints of presented technique. Prototype evaluation is conducted separately from formal users requirements, because each of eight properties is a synthetic synonym, subjectively perceived by each individual user, which in tacit perception might have a different meaning. As a consequence, we should instruct users and unbiasedly moderate discussions. On the other hand, a focus group should have sufficient number of representative users (participants) in order to obtain meaningful outcome. On the top of our concerns and doubts, after developing, evaluating and refining a few user interface prototypes, we see an approaching crunch: what or who can *guarantee avoiding reiteration and redundancy*? Returning to the starting point and regularly repeating the same solutions are the consequences of misconception and misunderstanding. Nevertheless, we argue that yet, it is worth to supplement presented technique with a procedural tool to track and provide control over changes to user interface, comparable with change management tools commonly used in software engineering.

Aforementioned process of user interface prototyping implies some interior dependency: *to implement system functionality, a user interface ought to be fully accepted by a project stakeholders*. In our opinion, this statement has its origin and substantiation in a strong influence of data flow paths throughout the whole interface, especially visible and objectively confirmable in all data-associated components.

3. Discussion

Johnson and Wilson [1994] emphasized that developing particular technology, which serves the needs of people, "requires a revised conception of computer system design", where the most important issue is to bring "the needs of people and the context of usage sharply into focus in the design process". In our model, in each iteration we reexamine user requirements and – what is the most important – we ask users to evaluate given prototype using presented matrix (Table 1). Next, we investigate scores due to perform indicated changes, and if necessary redefine standards, interface structure and DFDs. We fully agree with aforementioned authors opinion that "in developing systems that are intended to be used by people in the varied contexts of their work, private, social and leisure activities, the focus of design must be on the suitability of the designed artifact to support and complement human activity". Obviously, user interface is a part of the system providing access to its functionality, so deductively thinking, the need to collaborate with end-users in its development is justified due to their later acceptance of the entire system.

We argue that our model correspond to novel agile process for evaluation and redesign, where its authors [Garnik et al., 2014] distinguish problem-driven iterative stages, so called *creative sprints*. A creative sprint is "a single coherent episode of evaluation and redesign activities focused on a (group of) usability problem(s), as pre-identified by users", whereas in our approach each discussion panel aims at enhancing and refining given prototype, simultaneously validating user requirements and if required redefining developed artefacts. Eventually, final version of user interface is a cumulated effect achieved in a process, where coherent and relevant evaluation methodology must be engaged.

Microsoft is well-recognized IT industry company with revenue in 2013 more than $86 billion [Microsoft Annual Report, 2014] and 1,5 billion Windows users every day [Microsoft by the Numbers, 2015]. On the Microsoft Developer Network (MSDN) webpage [Microsoft MSDN, 2015] devoted implementing and prototyping user interface, we can read that "smart teams can eliminate

some mistakes before they ship by using UI prototyping and combined with usability studies, prototypes keep teams headed in the right direction". There is no further explanation what exactly means a "smart team", how many members it counts, whether there is some leader/ moderator and how to organize such teams. Nevertheless, we have to keep in mind that it's important to be clear and concise about "*why you're building and what you're building*". In our model, user requirements validation provides answers to those *why* and *what*. Furthermore, let us recall three basic categories of reasons for creating prototypes:

- *proof of concept*: a prototype can be used to illustrate that designed idea is functional, expressing its qualities in a visual and interactive way, also to prove its merit or value and motivate team members to think about the problem from different perspective;
- *design exploration*: if designing interactive software, we usually create mock-up and demonstrates how something will be used and how we can interact with it; in a few cases, particular mock-ups are tied to a usability study, where parts of the prototype can be evaluated in a structured way[1] or just serve as a mean to roughly express to a developer how something should look or work.
- *technical exploration*: developers may take into account different implementation approaches, coding techniques and finally programming languages, within each available technology platform has its own biases and preferences; in this case a prototype represents an exploration into which technology will support certain UI/UX features.

There are various technologies, tools and applications, which can be used for developing and testing prototypes, each of which has its advantages and disadvantages. We always consider type of design work that is being prototyped and the goals of the prototyping effort. For example, Microsoft Visual Basic and C# are one of the most comprehensive and rapid technologies for developing Windows-style UI prototypes, while SketchFlow, a part of Blend for Visual Studio 2013, revolutionizes the speed and efficiency with which we can demonstrate a vision for an application, as well as sophisticated user interfaces. At this point, it is worth mentioning Adobe Director and Adobe Flash, as popular UI prototyping tools among designers, most useful for non-standard GUI designs, multimedia and rich animation driven interfaces. Beginner user can start with FrontPage or other HTML editor, which allows for fast creation of simple prototypes, just using bitmaps files that simply illustrate a sequence of user interaction.

[1] For example, an evaluation questionnaire, with structured questions, includes some type of multiple-choice, where a list of answers or a rating scale, are provided to the respondent, who has to choose the most relevant answer.

Conclusions

There are a few emerging issues, raised during designing and specifying presented model, which we will try to solve in our future research:

- Gathering information from users about what they want and trying to develop document templates with their requested changes in UI, understandable for developers.
- Formulating UI changes in three different aspects (DFD, interface schema and standards), when an involved user does not use specific-domain vocabulary and formulates non-realistic requests.
- Designing for the all kind of user (from beginner to expert), while software testers are highly experienced.
- User understanding of a set of eight properties, being involved in a prototype testing, is frequently quite different.
- Finding appropriate UI testing platform.

Nevertheless, we are aware that our model needs to be further discussed with experts from software industry in order to obtain impartial feedback. We plan to refine this model and support it with techniques and tools applied respectively in the field of knowledge acquisition and management. This idea has received initial support during discussions and evaluation panels conducted between software vendor representative (teaming: designers, developers, testers, project managers) and our research group from Gdansk University of Technology.

Acknowledgements

The authors would like to thank Paweł Sokólski, Jacek Pisz, Rafał Niekurzak, Jacek Sokólczak, Jan Haase and Agata Sawicka for providing the user interface prototypes presented in this paper.

References

Chua C.K., Leong K.F., Lim C.S. (2010), *Rapid Prototyping. Principles and Applications*. World Scientific, New Jersey-London- Singapore-Hong Kong.

Dix A., Finlay J., Abowd G., Beale R. (2004), *Human-Computer Interaction*. Prentice Hall.

Fadeyev D. (2009), *8 Characteristics Of Successful User Interfaces*, http://usabilitypost.com/2009/04/15/8-characteristics-of-successful-user-interfaces/ (accessed: 22.06.2015).

Garnik, I., Sikorski, M., Cockton, G. (2014), *Creative Sprints: an Unplanned Broad Agile Evaluation and Redesign Process* [in:] *Proceedings of the 8th Nordic Conference on Human-Computer Interaction: Fun, Fast, Foundational*, ACM, pp. 1125-1130.

Gawin B., Marcinkowski B. (2013), *Symulacja procesów biznesowych. Standardy BPMS i BPMN w praktyce*, Helion, Gliwice.

Johnson P., Wilson S. (1994), *Task-Based Design and Prototyping*, ACM SIGCHI Basic Research Symposium, Boston, ACM.

Kubiak B.F., Weichbroth P. (2010), *Cross- and Up-Selling Techniques in E-commerce Activities* [in:] B.F. Kubiak, A. Korowicki (eds.), *eCommerce, ePayments and New Entrepreneurship*, "Journal of Internet Banking and Commerce", Vol. 15, No. 3, Array Development, Ottawa, pp. 217-225.

Microsoft 2014 Annual Report (2014), http://www.microsoft.com/investor/reports/ar14/index.html (accessed: 1.09.2015).

Microsoft by the Numbers (2015), http://news.microsoft.com/bythenumbers/index.HTML (accessed: 1.09.2015).

Microsoft MSDN (2015), https://msdn.microsoft.com/en-us/library/windows/desktop/ff728826%28v=vs.85%29.aspx (accessed: 1.09.2015).

Owoc M.L., Weichbroth P. (2014), *Validation Model for Discovered Web User Navigation Patterns* [in:] E. Mercier-Laurent, D. Boulanger (eds.), *Artificial Intelligence for Knowledge Management*, Springer, Berlin Heidelberg, pp. 38-52.

Renolen A. (2000), *Modelling the Real World: Conceptual Modelling in Spatiotemporal Information System Design*, "Transactions in GIS", Vol. 4 (1), pp. 23-42.

Schneiderman B., Plaisant C., Cohen M., Jacobs S. (2009), *Designing the User Interface: Strategies for Effective Human-Computer Interaction*, 5th Edition, Addison-Wesley, Reading, MA.

Schumaker D. (2001), *User Interface Standards*. https://msdn.microsoft.com/en-us/library/office/aa217660%28v=office.11%29.aspx (accessed: 1.09.2015)

Schwaber K. (2004), *Agile Project Management with SCRUM*, Microsoft Press.

Sharp H., Rogers Y., Preece J. (2005), *Interaction Design. Beyond Human-Computer Interaction*. Wiley, Hoboken, NJ.

Sikorski M. (2010), *Interakcja człowiek-komputer*, Wydawnictwo PJWSTK, Warszawa.

Sikorski M. (2012), *Usługi on-line. Jakość, interakcje, satysfakcja klienta*, Wydawnictwo PJWSTK, Warszawa.

Sikorski M. (2012), *User-System Interaction Design in IT Projects*. Gdansk University of Technology, Gdansk.

Sikorski M. (2013), *Evolution of End-User Participation in IT Projects* [in:] M. Pańkowska (ed.), *Frameworks of IT Prosumption for Business Systems Development*, IGI Global Hershley, New York, pp. 48-63.

Sikorski M. (2014), *Interactive Prototypes in Teaching User-Centred Design and Business Process Modelling* [in:] M. Zięba, A. Ziółkowski (eds.), *IT Tools in Business Education*, VIA University College, Horsens, pp. 49-60.

Snyder C. (2003), *Paper Prototyping*. Morgan Kaufmann, Burlington, MA.

Stickdorn M., Schneider J. (2010), *This is Service Design Thinking*, BIS, Amsterdam.

Szekely P. (1995), *User Interface Prototyping: Tools and Techniques*, *ICSE '94 Proceedings of the Workshop on Software Engineering and Human-Computer Interaction*, Springer-Verlag, London, pp. 76-92.

Vervest P., Van Heck E., Preiss K., Pau L.-F. (2005), *Smart business networks*, Springer, Berlin.

Walker M., Takayama L., Landay J.A. (2002), *High-Fidelity or Low-Fidelity, Paper or Computer? Choosing Attributes When Testing Web Prototypes*, *Proceedings of the Human Factors and Ergonomics Society 46th Annual Meeting. September 29th October 4, Baltimore (USA)*, HFES, Santa Monica, pp. 661-665.

Warfel T.Z. (2009), *Prototyping. A Practitioner's Guide*, Rosenfeld Media, New York.

PROTOTYPOWANIE INTERFEJSU UŻYTKOWNIKA: TECHNIKI, METODY I NARZĘDZIA

Streszczenie: Obecnie użyteczność i ergonomia interfejsu użytkownika to krytyczne czynniki akceptacji interaktywnych aplikacji przez ich użytkowników. Zatem z perspektywy użytkownika sukces całego przedsięwzięcia jest uzależniony od jakości zachodzących interakcji pomiędzy nim a systemem. Tym samym niezbędne jest zaangażowanie wszystkich interesariuszy projektu poprzez udostępnienie im niezbędnych narzędzi służących wymianie i kodyfikacji wiedzy. Potrzeba wykorzystania tego typu narzędzi jest szczególnie widoczna w cyklu rozwoju oprogramowania, określanym mianem prototypowania. W oparciu o literaturę i przeprowadzone badania autorzy dokonali przeglądu, analizy i oceny dostępnych technik, metod i narzędzi, aktywnie wspierających proces prototypowania interfejsu użytkownika. Ponadto autorzy zaproponowali również nowy model prototypowania interfejsu użytkownika. Głównym założeniem leżącym u jego podstaw jest interakcyjny rozwój prototypu systemu i komponentów interfejsu użytkownika, gdzie sukcesywnie w kolejnych iteracjach poddawane są one ocenie przez użytkowników końcowych. Zaproponowane podejście można określić jako ukierunkowane na użytkownika, którego celem jest dostarczenie zarówno oczekiwanej funkcjonalności, jak i optymalnej użyteczności i ergonomii w dostępie do niej.

Słowa kluczowe: interfejs użytkownika, prototypowanie.