A Dialectic Approach for User Requirements Engineering for Multi-User Systems

Research Abstract of my Ph.D. Proposal

Andreas Maier

Fraunhofer Institute for Experimental Software Engineering (IESE)
Kaiserslautern, Germany
andreas.maier@iese.fraunhofer.de

1 Problem

Interactions with information systems shall create a positive user experience (UX) based on a perceived fulfillment of the user requirements during these interactions [15]. User requirements reveal the user needs with respect to the achievement of particular goals with the help of a particular system in a particular context. The user needs are the underlying rationales of desires which are expressed by users [18]. However, users tend to form requirements for desired experiences or design solutions rather than for concrete goals or needs. A proper user requirements engineering has to identify these needs by reasoning, which requires an intense engagement with user goals, needs, and desires. This is especially hard in projects with a large number of geographically distributed stakeholders, where the organizational handling is hard and cost intensive [1]. Due to this fact, the number of misunderstandings, overlooked requirements and ambiguities increases and cannot be discovered and resolved in an early development phase. As shown in [18], [19], particular existing requirements elicitation approaches are not appropriate for a large group of stakeholders and leave out important information; others require a lot of effort or time or face a hard selection of adequate stakeholders; some might elicit incorrect and outdated requirements, or are unlikely to reflect the experience of actual users; some are applicable only when a system already exists or are hard to organize. For user requirements elicitation, the current best practice is provided by [9]: design solutions (prototypes) are produced after having understood and specified the context of use and the user requirements and evaluated by end-users or by designers who take the end-users' view. This human-centered design process is performed iteratively to revise and refine the prototypes in each iteration [9]. Further, [9] advances the view that user requirements are better expressed and understood (and therefore are validated) when prototypes are available. In general, prototypes are used for the evolutionary discovery, refinement, and satisfaction of user requirements [11]. They are also used for the better understanding of problems, the dissolving of uncertainties, and the completion and validation of requirements; furthermore, they are supposed to make the user needs more tangible, explore possible designs, and uncover overlooked requirements [18]. With prototype, we refer to any type of prototype, from a mock-up, via a simulation, to a fully functional electronic prototype [9, 18]. However, prototypes have a number of known and severe disadvantages [8]: they often represent what users said, but not what they meant, the set of functions provided by prototypes prevents users from articulating other key requirements, users are not engaged effectively, and the number of features and options provided by the prototype may confuse the users. These disadvantages are predominantly solved by a prototyping process comprising the creation of personas, the creation of a series of scenarios which describe the goals which shall be achieved by using the system under development, and finally the rapid prototyping of potentially required functionality [8]. But this prototyping process has to be run through several times and requires a number of prototypes both at specific phases of development and throughout the whole system development. Remaining general challenges of prototyping are that users have difficulties to express their needs with respect to a system under development until they concretely see an existing solution. In such cases, prototypes have to be built on the developers' best guesses just to learn that the solution which the prototype offers is not what the users need and desire. Furthermore, prototypes can distract users and miss their original purpose by their visual design; in such cases, the visual design is what the users talk about instead of concepts or requirements the prototype represent [18]. In any case, prototypes bear the danger of confusing users and limit their thoughts to the functionality and design provided by the prototypes and preventing requirements engineers from eliciting the real needs and rationales of user statements which is the precondition for validating the user requirements. Although the resolution of uncertainties early in the development process is often the primary reason for creating prototypes [18], it is this usage of prototypes which we consider as being too time and cost intensive and bearing the problems we mention above. Usually, end-users start concrete considerations of their requirements not until prototypes are available [18], which results in a late validation of user requirements. So, the elicitation as well as the validation of user requirements with the help of such prototypes seems not to be the most appropriate practice for projects with a large number of geographically distributed stakeholders. The resulting question is: how can requirements of distributed end-users be validated earlier in the life-cycle?

2 Relevance

Currently, (user) requirements evolve from initial user intentions which are clarified by corresponding statements. These statements are negotiated, i.e., a set of statements has to exist until negotiation can be initiated. When the statements meet particular characteristics, are consistent and feasible, their meaning and rationales are identified, and end-users agree on the statements, they become requirements [10]. In the agreement process with prototype-based approaches, end-users often realize too late that they did not get what they requested. Due to these difficulties in the elicitation and the validation of requirements either reworks become necessary that cause high costs and an extended time to market or the user requirements are met insufficiently, which leads to systems with a bad quality and without a basis for a positive UX. Reaching the state of specified requirements early in the software development process allows for an early validation of the requirements and therefore to an earlier, cost saving implementation of the requirements and finally to a more positive UX. Prototypes do not have to be developed for the sake of requirements evolution and discarded afterwards, but can be evolutionary prototypes for implementing already existing requirements with a proper architecture.

3 Proposed Solution

From our point of view, requirements engineering does not only denote the elicitation and specification of problems in the as-is situation, but also the discussion of possible solutions. For both the elicitation and discussion of user requirements, user statements have to be questioned to identify the rationales of these statements. This way, further discussions are stimulated, users are motivated to participate in the discussions, and discussions are led into a direction which otherwise would not have been considered. This approach leads to user requirements that better represent the expectations and desires of the users as well as their underlying needs, in contrast with prototype-based approaches. The stimulation of discussions of statements including their questioning statements by a large number of end-users leads to a common understanding of the requirements and their effects on the product under development. Also, a larger number of statements are provided and an early clarification of statements is possible. Therefore, an earlier specification and an earlier validation of user requirements compared to prototype-based approaches can take place. Due to a deeper engagement with their needs and the discussion of possible solutions for the system under development, we assume that end-users get a clear mental system image. This helps users validate their requirements; a prototype for negotiations and clarifications is not necessary anymore. Simultaneously, the clearer system image enables the users to anticipate the UX they will gain when they actually use the finally developed system. As an effect, the system is supposed to create a more positive UX and motivate a larger number of potential users to be engaged in using the system than a system developed with a prototype-based approach. Furthermore, the proposed approach reduces the system's time to market.

To stimulate reflections on user statements and to solve the depicted challenge of reasoning with existing requirements elicitation approaches, we propose to use dialectics [4], [14]. Dialectics is a method of reasoning by the creation of the synthesis of opposites, in our case a logical method for the identification of rationales of user statements through the synthesis of opposite user statements. Since a large number of end-users shall participate in the discussions of their statements and large-scale projects usually involve geographically distributed end-users, a web-based approach for the facilitation of conjoint discussions is proposed. Therefore, we plan to implement a web-based discussion platform which uses dialectics and automatically creates questioning statements, which represent the contrary statements to the ones provided by the end-users. Topics for discussions on this platform shall be classified according to UX factors and general user requirements issues, both of which are provided by the platform. Clarification questions shall be identified by implementing a glossary, which is dynamically created based on terms used in the user statements. With the help of linguistic techniques like part-of-speech tagging, stemming, and pattern matching, terms which are not defined yet can be identified by comparing the terms in the user statements with the ones in the glossary. Conflicting requirements can be automatically identified with the help of a lexical ontology: again, linguistic algorithms can be used for looking up antonyms of terms used in user statements and thus for identifying conflicts. The requirements' adherence to [10] is based on part-of-speech tagging and keyword-spotting; therefor, a keyword list is used and matched with the user statements to detect particular characteristics. Undetected characteristics are treated as unmet. Users of the discussion platform are all end-users of the system under development, i.e., end-users participate in discussions on a voluntary basis and are not pre-selected on particular characteristics and forced to participate as in workshops, focus groups, or interview approaches, for example. All in all, the proposed solution comprises (1) a UX quality model which provides the topics for discussions, (2) linguistic algorithms for the automated check on the adherence of user statements to characteristics provided in [10], (3) for the generation of questioning statements, (4) statements for term clarification, (5) and the identification of conflicting requirements, and (6) a web-based user requirements discussion platform based on dialectics and the linguistic algorithms.

4 Novelty of the Approach and Related Work

The problem we want to tackle is so far described by the terms 'requirements evolvement', 'requirements negotiation' or 'requirements discussion', which fail to support the conjoint development of a complete set of correct requirements specifications at an early stage. Although the problem of misleading, conflicting, incorrect, or incomplete requirements is known, existing approaches mainly deal with the requirements negotiation process, which requires an existing set of statements and thus bias and limits the users' thoughts [7]. There are a number of approaches focusing on the distributed requirements negotiation process (e.g., [5, 7, 13, 16]). Other related approaches address the analysis of requirements specifications [17] and the discovery of stakeholder requirements in projects with distributed stakeholders [12]. Further approaches focus on distributed participatory design [6], and forum-based [3] or social media-based requirements elicitation [11]. These approaches help collecting statements and specify requirements, but do not question the statements and their rationales or do not ask for rationales at all, and do not stimulate the elicitation and discussion of tacit requirements beyond already elicited statements. Thus, the proposed approach is supposed to be a novelty in user requirements elicitation and validation for projects with a large number of geographically distributed end-users in terms of the early and semi-automated identification of the rationales of user statements by questioning these statements.

5 Research Method

To get to the proposed solution, we follow the GQM approach [2] and define the following GQM goals:

G1: Analyze the set of user requirements resulting from using the dialectic discussion platform for the purpose of characterization with respect to the adherence to the characteristics of requirements according to [10] from the viewpoint of requirements engineers and user experience engineers in the context of projects with geographically distributed end-users;
G2: Analyze the user requirements engineering process of the dialectic discussion platform for the purpose of comparison with respect to the earliest possible point in time of conduction of user requirements validation from the viewpoint of requirements engineers and user experience engineers in the context of projects with geographically distributed end-users;
G3: Analyze the system developed in consideration of the user requirements specified by applying the proposed user requirements engineering process for the purpose of comparison with respect to the perceived user experience from the viewpoint of end-users in the context of projects with geographically distributed end-users;
G4: Analyze the proposed user requirements engineering process for the purpose of comparison with respect to costs from the viewpoint of the corporation in the context of projects with geographically distributed end-users.

G1 requires the check of each user requirement against its adherence to a large number of requirements characteristics provided by [10]. To get to a pragmatic approach, we will assess if each of these criteria is applicable to and necessary for user requirements. In terms of G3, we assume that the perceived UX is more positive for the end-users who participated in the discussions of the statements leading to the user requirements than for the end-users who did not participate in the discussions and assess the fulfillment of their requirements when they use the system. Although we wish to motivate every single end-user in participating in the discussions of statements via the dialectic discussion platform, we assume a number of end-users will not participate. Following the GQM approach, we are going to refine the GQM goals into questions about measurable issues which contribute to the achievement of these goals and appropriate metrics for answering each question.

To illustrate why prototypes might be inappropriate for the elicitation and validation of user requirements in projects with geographically distributed end-users, we are going to perform a literature research on existing models of UX emergence and combine the findings to a new UX emergence model. We also plan to build the dialectics-based online discussion platform based on a literature research on factors which influence UX and are influenceable by software engineering. For this purpose, we will create a new UX quality model based on the identification of UX factors influenceable by software engineering in existing UX models and enhanced with appropriate metrics also found in literature. The identified factors shall serve as topics within the discussion platform. In addition, the literature research is used to reveal appropriate linguistic methods for the generation of contrastive statements, the generation of statements to clarify aspects, the identification of conflicts, and for checking the user requirements' adherence to [10]. Regarding the appropriate formulization of dialectics, we also begin with a literature research and may need to perform some interviews or case studies. This also holds for single aspects of the dialectics-based online discussion platform. Finally, we are going to evaluate the discussion platform in an experiment where we want to validate the following hypotheses and compare them with a prototype-based requirements engineering approach:

- H1: The user requirements specified via the dialectics-based online discussion platform are correctly specified (i.e., do not need further refinement, clarification, or negotiation) according to the requirements characteristics provided by [10];
- H2: The user requirements can be validated earlier compared to a prototype-based approach;
- H3: The user experience is more positive compared to a prototype-based approach;
- H4: The costs (money, time, resources) the whole requirements elicitation, specification, and validation process causes are significantly lower compared to a prototype-based approach.

H2 is assumed since correct requirements specifications are supposed to be available early, which allows for early requirements validation. Qualitative studies may be necessary to find out which is the most appropriate technique to approximate an absolute correctness and completeness of user requirements. We assume H3, because we suppose the users to be deeper engaged in discussions and because of a larger number of elicited user requirements due to this deeper engagement. The assumption of H4 is based on the supposed achievement of the high-level goal of this approach that a prototype for requirements elicitation, negotiation, and clarification does not have to be developed.

6 Progress in My Research

I have been working on my thesis since beginning 2013. Since then, I have been sharpening my theses and established a border between the scope of my approach and existing approaches, state-of-the-art, and best practice. Currently, I almost finished the literature research, had a deeper look into the UX emergence process and have built a new UX emergence model based on existing literature. The same is true for factors which influence UX and are influenceable by software engineering. Both models are important bases of the discussion platform, which will follow dialectics principles found in literature and use linguistic algorithms. I plan to begin implementing the dialectics-based online discussion platform based on the findings in the literature research soon. I I wish to begin my experiment later in 2014 and finish my thesis end of 2015 / beginning of 2016.

References

1. Bang, J.Y. et al.: CoDesign: a highly extensible collaborative software modeling framework. In: Kramer, J. et al. (eds.) ICSE (2). pp. 243–246 ACM (2010).
2. Basili, V. et al.: The goal question metric approach. In: Marciniak, J. (ed.) Encyclopedia of Software Engineering. Wiley (1994).
3. Castro-Herrera, C. et al.: A Recommender System for Requirements Elicitation in Large-scale Software Projects. Proceedings of the 2009 ACM Symposium on Applied Computing. pp. 1419–1426 ACM, New York, NY, USA (2009).
4. Chesñevar, C.I. et al.: Logical Models of Argument. ACM Comput. Surv. 32, 4, 337–383 (2000).
5. Damian, D. et al.: The role of asynchronous discussions in increasing the effectiveness of remote synchronous requirements negotiations. In *Proceedings of the 28th international conference on Software engineering* (ICSE '06). pp. 917-920. ACM, New York, NY, USA (2006).
6. Danielsson, K. et al.: Distributed Participatory Design. CHI ’08 Extended Abstracts on Human Factors in Computing Systems. pp. 3953–3956 ACM, New York, NY, USA (2008).
7. Grünbacher, P., Seyff, N.: Requirements Negotiation. In: Aurum, A. and Wohlin, C. (eds.) Engineering and Managing Software Requirements SE - 7. pp. 143–162 Springer Berlin Heidelberg (2005).
8. Hakim, J., Spitzer, T.: Effective prototyping for usability. Professional Communication Conference, 2000. Proceedings of 2000 Joint IEEE International and 18th Annual Conference on Computer Documentation (IPCC/SIGDOC 2000). pp. 47–54 (2000).
9. International Organization for Standardization: ISO 9241-210: Ergonomics of human-system interaction - Part 210: Human-centred design for interactive systems. (2010).
10. ISO/IEC/IEEE Systems and software engineering - Life cycle processes - Requirements engineering. IEEE STD 29148:2011. c1–94 (2011).
11. Lim, S.L., Finkelstein, A.: StakeRare: Using Social Networks and Collaborative Filtering for Large-Scale Requirements Elicitation. IEEE Trans. Software Eng. 38, 3, 707–735 (2012).
12. Maiden, N.: Systematic Scenario Walkthroughs with ART-SCENE. In: John Wiley & Sons (ed.) Scenarios, Stories, Use Cases: Through the Systems Development Life-Cycle. pp. 161–178 (2004).
13. Mikulovic, V., Heiss, M.: “How Do I Know What I Have to Do?”: The Role of the Inquiry Culture in Requirements Communication for Distributed Software Development Projects. Proceedings of the 28th International Conference on Software Engineering. pp. 921–925 ACM, New York, NY, USA (2006).
14. Mitroff, I.I., Mason, R.: Dialectical pragmatism. Synthese. 47, 1, 29–42 (1981).
15. Nielsen Norman Group: The Definition of User Experience, http://www.nngroup.com/articles/definition-user-experience/.
16. Potts, C. et al.: Inquiry-based requirements analysis. Software, IEEE. 11, 2, 21–32 (1994).
17. Rupp, C.: Linguistic methods of Requirements Engineering (NLP). Proceedings of the EuroSPI 2000 Conference. pp. 68–80 , Copenhagen, Denmark (2000).
18. Wiegers, K.E., Beatty, J.: Software Requirements. Microsoft Press, Redmond, WA, USA (2013).
19. Zowghi, D., Coulin, C.: Requirements Elicitation: A Survey of Techniques, Approaches, and Tools. In: Aurum, A. and Wohlin, C. (eds.) Engineering and Managing Software Requirements SE - 2. pp. 19–46 Springer Berlin Heidelberg (2005).