5-20-2011

An Empirical Scenario for the Evaluation of Requirements Elicitation Tasks

Aaron Read
University of Nebraska at Omaha, aread@mail.unomaha.edu

Andy Callens
University of Nebraska at Omaha, acallens@mail.unomaha.edu

Cuong Nguyen
University of Nebraska at Omaha, cdnguyen@mail.unomaha.edu

Gert Jan de Vreede
University of Nebraska Omaha, gdevreede@usf.edu

An Empirical Scenario for the Evaluation of Requirements Elicitation Tasks

Aaron Read
Center for Collaboration Science
University of Nebraska Omaha
aread@mail.unomaha.edu

Andy Callens
Center for Collaboration Science
University of Nebraska Omaha
acallens@mail.unomaha.edu

Cuong Nguyen
Center for Collaboration Science
University of Nebraska Omaha
cdnguyen@mail.unomaha.edu

Gert-Jan de Vreede
Center for Collaboration Science
University of Nebraska Omaha
gdvreede@mail.unomaha.edu

ABSTRACT

While requirements elicitation has been established as a crucial phase of the systems development process, empirical research on the topic of requirements elicitation is sparse. In this paper we present a requirements elicitation scenario that can be used by researchers to evaluate different methods of eliciting a set of requirements. This scenario consists of an elicitation tasks, a system features set, and a coding method. The task revolves around the generation of a set of features for an online student textbook exchange. Such a system is likely to be familiar to students. We present a set of criteria to assess the quality of a requirements elicitation scenario and provide support from our experience using the scenario.

Keywords

Requirements elicitation, requirements specification, requirements engineering, systems development, experimentation, experimental task.

INTRODUCTION

Software development remains a challenging process with only one third of projects successfully completed; other projects are canceled, considered late, over budget, or completed with fewer features than planned (Rubenstein, 2007). Poorly defined requirements are considered to be a leading factor in project failure (Hofmann and Lehner, 2001). A key step in the requirements development or requirements engineering process is the elicitation of requirements from stakeholders (Hickey and Davis, 2004). While this step is important, it remains a key stumbling block in the process (Pan, 2005). More than downstream software development processes, it involves working with conflicting goals, prioritizations, and unestablished system boundaries to create a solution (Lamsweerde, 2000).

To improve the requirements elicitation process, much of the research on requirements engineering has focused on the development of elicitation notations and techniques (Cheng and Atlee, 2007). While many of these techniques have been applied and assessed in the field (Boehm, Grunbacher, and Briggs, 2001; Hickey and Davis, 2004) such evidence has not yet resulted in the development and testing of theory that can build generalizable knowledge applicable to the requirements engineering domain. The software engineering community, which includes research covering requirements engineering topics, has recently emphasized the need for more empirical studies and empirical studies of a higher quality (Sjoberg, Dyba, and Jorgensen, 2007; Wohlin, Höst, and Henningsson, 2003). Requirements elicitation research is especially lacking (Sjoberg et al., 2005). This may be due to the fact that it is difficult to conduct experiments in requirements engineering that have both rigor and relevance(Alves, Niu, Alves, and Valença, 2010). A central issue of research on requirements elicitation is the ability of a user to describe a set of desired features of the system, based on tasks or goals that the user needs to accomplish. It is therefore important that an experimental scenario resembles a situation where a subject or group of subjects can function as realistic, potential informants of system requirements. Many real world systems may be too large or require too much expertise for a group of subjects, especially student subjects, to generate a complete set of requirements in a short amount of time. Completeness has been considered a key indicator of the quality of requirements (Pitts and Browne, 2007). Also, for an elicitation experiment to be practical, the system involved should not be so complex that the task of generating the complete set of requirements for it cannot be completed in a short time. In this paper, we first propose several criteria to assess experimental scenarios to empirically study requirements elicitation. We then present an experimental scenario on a student book exchange system which includes a system features set along with the accompanying task, and coding method. We assess this scenario using the proposed evaluation criteria.

BACKGROUND

Requirements elicitation can be understood as a means of building knowledge about a system (Hickey and Davis, 2004). Many methods of requirements elicitation exist, including interviews (Pitts and Browne, 2007), user observation, model driven techniques, and group elicitation techniques such as JAD workshops or brainstorming sessions (Boehm et al., 2001; Bragge and Merisalo-Rantanen, 2009; Duggan, 2003). Because there is a continuing proliferation of methods in requirements elicitation, there is a need for practitioners to have valid evidence of a method's merit (Dyba, Kitchenham, and Jorgensen, 2005). The evidence needs to have both *internal validity*—strong justification that the variables were measured correctly, and strong justification of a causal link between independent and dependent variables, as free from confounding variables as possible, and *external validity*—the ability of the results of the experiment to generalize to other situations, specifically those relevant to requirements engineering (Cook and Stanley, 1963). We use these principles as a basis for the evaluation of experimental scenarios:

Relevance to the Requirements Elicitation Domain (External validity): Some research on the generation of requirements focuses solely on creating a higher number requirements (Maiden and Robertson, 2005). While this has its benefits, many of the popular requirements engineering methods and formats relate to the idea of creating complete sets of requirements (Pitts and Browne, 2007). Goals, for example, a prominent requirements modeling paradigm, allow analysts as well as stakeholders to reason about the completeness of asset of requirements (Lamsweerde, 2000). Since it is most likely that students subjects will be involved in the experimental task, the task should give the students as much as possible a vested interest in the system as well expertise about the necessary tasks to be completed with a system in order to reflect users commonly involved in requirements elicitation.

Internal Validity: A key benefit of experimental research is the ability to isolate independent variables for their effects on desired outcomes (Wohlin et al., 2003). The internal validity of an experiment can still be threatened by poor measurements of the variables or the presence of confounding variables. Many of the threats to external validity are also threats to internal validity. If students do not have the expertise or interest in generating requirements for a system, their lack of ability to generate requirements may obscure the true difference between requirements elicitation methods. A lack of clarity of the elicitation task will also obscure results in a similar fashion. Finally, the ability to assess completeness in a meaningful way is key to internal validity. The results of the experiment will be inconclusive about an elicitation method's ability to generate a complete set of requirements if "requirements completeness" cannot be measured.

The criteria are summarized in Figure 1 below:

External Validity:

- **Tests for completeness**. Can the task be assessed for its ability to result in a complete set of requirements?
- **Interest in the system.** Are students interested enough in the system to be motivated to expand effort in the requirements elicitation task?
- **Expertise.** Do the students have sufficient expertise in the tasks that must be completed with the system?

Internal Validity:

- **Level of complexity of the system**. Can the students generate a complete system in a short amount of time for the given system? Is the system's complexity such that a complete set of requirements can be pre-defined by the researchers?
- **Experience with a similar system**. To what extend do the students have prior user experience with similar or comparable systems? Is there a variation in the amount of experience using the system?
- **Clarify of the task.** Do the students understand the tasks? To what extent where the students certain that they executed the task correctly?

Figure 1. Criteria to assess experimental requirements elicitation tasks.

THE SYSTEM FEATURE SET, TASK, AND ACCOMPANYING ANALYSIS

In this section we present an experimental scenario concerning a book exchange system. It consists of a set of system features, the accompanying requirements elicitation task, and the method for coding features generated against the feature set. The feature set was created as part of an experiment to test different methods of generating user stories in groups (Read, Callens, Nguyen, and De Vreede, Submitted to AMCIS, Submitted)The entire feature set consists of 101 features and space does not allow us to present it in its entirety. We present the feature categories in the Figure 2 below. The full feature set may be obtained by contacting the authors.

Development of the Feature Set

Many of the features were taken from an actual book exchange website where students exchanged textbooks, but it did not provide payment services. The features were then enriched by features inspired from user stories generated by the first several groups of subjects in the study (Read et al., Submitted to AMCIS, Submitted). During the pilot round of coding the data the feature sets were refined through discussion among the researchers about the meaning of categories and features. Such discussions resulted in the removal of duplicate features, the changing of the wording of features and feature categories and subcategories. A key aspect of the feature subcategories (see Figure 2 below) is they actually provide high level descriptions of features. We often ran into the challenge of coding a user story submitted by a student that was not quite the same as one in our feature set but nevertheless related to a feature which accomplished nearly the same goal as a feature in our feature set. Organizing features in this manner allowed us to give some credit to features that were essentially within the scope of the system, but did not literally match any of our features.

1. User account
    a. Account Setup
    b. Book Advertising
    c. Professor
2. Transactions
    a. Shopping cart
    b. Buyer Indicates Intention to Buy
    c. Seller Notification of Buyer's Interest
    d. Transaction Completion Notification
    e. Transaction Reviewing
    f. History
3. Books
    a. Book Content Feedback
    b. Book Condition
    c. Book Price
    d. Book Ad Information
    e. Browsing for Books
    f. Searching for Books
4. Administration
    a. Managing Book Ads
    b. Managing Users
    c. Managing Reviews
5. Out of Scope

Figure 2. The Book Exchange Feature Set Categories.

Development of the task

We feel that characteristics of the requirement elicitation task may vary significantly to accommodate for a variety of requirements methods. However, we found that providing examples of requirements was helpful in assuring that students generated features relevant to the system (and not just desired results of the system such as "I want the system to be able to save me money"). This also helped encourage subjects to generate requirements at the right level of abstraction. We feel that, at a minimum, a task should include some example of how a requirement should be generated. We did not restrict or eliminate contributions that did not meet the specified format. The task description for our experiment is given to students is presented in Figure3:

Every subject in the groups will be given the same overview description of the system to be designed as follows:

The Book Exchange is a website which will be designed to allow students at this university to buy and sell text books at a reasonable price. The website will not provide payment services; it will simply allow sellers to post items for sale, allowing potential buyers to search for their textbook offerings. The website will also have features that facilitate a buyer's search for textbooks. For example, the website will have access to which textbooks are required for a given course.

Next, the subjects receive the following instructions:

- Provide as many user stories as possible. A user stories is a story that provide a feature that the system to be designed should have in your opinion. A recommended form for a user story is:
  "As a <type of user>, I want <some goal> so that <some reason>." (Type of user = buyer, seller, professor, administrator).

  e.g. "As a buyer, I want to be able to see the prices of all the books so that I can decide whether to buy the book or not.
- Continue to brainstorm user stories

Your stories should not be more than two sentences in length. You are NOT being asked to come up with a technical description of the website (i.e., it will use mySQL database for data storage). Instead we are asking you to describe what the website can do from the perspective of the website's users.

Figure 3: Experimental Book Exchange Task Description

Development of the Coding technique

The coding set was developed iteratively, along with the feature set as explained above. Rules and definitions were developed to take set of raw subject input (user stories) and find the number of related features in the Book Exchange. The coding technique is depicted in Figure 4.

1. Decide what information in the feature is relevant
   a. Use all relevant information in the user story to decide which features are covered, including which user the user story is for.
   b. If a subject takes a specific feature concept into another direction (for example, the subject says to use student ID and password to login, but our system says a user name and password is needed), then focus only on how the concept of the feature (user name and password) relates to the book exchange feature set.
2. Decide which features are covered by the information in the user story
   a. If a user story contains more than one feature concept, categorize the user story as applying to all applicable features.
   b. Choose the most directly related features. Do not choose other features that the proposed feature depends on unless they are specifically mentioned.
   c. Assume that a requested feature is performed by the system and not by the user. For example, if a user story makes the request for a count of similar books, do not use the feature browsing for books (and counting them manually).
   d. Each category contains a *high level feature*. For example, the "Browsing for Books" category contains the high level feature "The system allows the user to browse for books." If a feature does not match the other features in the category, but can be considered an elaboration of the high level feature ("The system allows the user to browse for books by media type").
3. Resolve differences in features covered between coders by feature. If there are differences in features covered, then discuss which user story should or should not have applied to the feature and code the feature as covered based on the result of the discussion.

Figure 4: Coding Procedure for the Book Exchange feature set

EVALUATION

To evaluate our feature set and analysis according to the criteria presented in Figure 1, we review its use in a recent experiment both from the viewpoint of the researchers involved and the students involved.

From the Viewpoint of the Researchers

As researchers, we can provide some evidence of the successfulness of the feature set, task, and analysis to measure a complete set of requirements: We feel that the inter-rater reliability achieved-- 88.8% shows that researchers can come to an agreement about what is part of the system. In the experiment, users generated an average of 38 (SD=11.66) out of the 101 features. Students were able to generate a significant portion of the feature set in 35 minutes. The participants in the groups where certain prompting techniques were used kept generating requirements the entire time(Read et al., Submitted to AMCIS, Submitted). Based on our experience we believe that with more time allocated for the elicitation task, the students could have generated the majority of the features.

From the Viewpoint of the Participants

To assess the participant’s viewpoints we asked questions about engagement, perceived level of complexity, their experience with student websites, and the clarity of the task. Here is a brief overview of the responses from a set of 10 subjects:

Interest: All students felt challenged and engaged by the requirements elicitation task. Students felt input mattered in the website’s development since they envisioned such a website being used heavily in the future.

Experience: Having previous experience with similar websites like Amazon and Ebay helped formulate ideas by providing liked and disliked features of these previously used websites. However, some students had no experience previously with such websites.

Complexity: All but two students felt they could generate the requirements in the time allotted. Most students felt confident that they could have generated a complete set of requirements for the website.

Understanding: Most students felt like they were qualified to provide requirements for the website.

CONCLUSION

We have presented an experimental scenario for requirement elicitation that provides a feasible approach to empirically evaluate requirement elicitation methods and that has qualities to increase its internal and external reliability. It allows researchers to take advantage of a convenient population, students, in an experimental setting that is real enough to them to generalize the study results to users in other settings. We also feel that this requirement elicitation task and feature set could generalize to several different formats of requirements (e.g. use cases, class diagrams, and goals), as well as different requirements methods (e.g. interviews). A limitation of our work is that the task may be less useful for students at universities that have access to a real book exchange system. They may not be suitable subjects since they are too expert in the system. Also, more experience with different student subject populations is required to further assess quality of the experimental scenario. Caution should be exercised about using this method to asses requirements elicitation methods in scenarios with very complex systems.

REFERENCES

1. Alves, V., Niu, N., Alves, C., and Valença, G. (2010). Requirements engineering for software product lines: A systematic literature review. *Information and Software Technology*, 52, 8, 806-820.
2. Boehm, B., Grunbacher, P., and Briggs, R. O. (2001). Developing groupware for requirements negotiation: lessons learned. *Software, IEEE,* 18, 3, 46-55.
3. Bragge, J., and Merisalo-Rantanen, H. (2009). Engineering E-Collaboration Processes to Obtain Innovative End-User Feedback on Advanced Web-Based Information Systems. *Journal of the Association for Information Systems*, 10, 3, Article 4.
4. Cheng, B. H. C., and Atlee, J. M. (2007). *Research Directions in Requirements Engineering*. Paper presented at the 2007 Future of Software Engineering.
5. Cook, T. D., and Stanley, J. C. (1963). Experimental and Quasi-Experimental Designs for Research. Boston Houghton Mifflin Co.

6. Duggan, E. W. (2003). Generating systems requirements with facilitated group techniques. *Hum.-Comput. Interact.*, 18, 4, 373-394.
7. Dyba, T., Kitchenham, B. A., and Jorgensen, M. (2005). Evidence-Based Software Engineering for Practitioners. *IEEE Softw.*, 22, 1, 58-65.
8. Hickey, A. M., and Davis, A. M. (2004). A Unified Model of Requirements Elicitation. *Journal of Management Information Systems,* 20, 4, 65-84
9. Hofmann, H. F., and Lehner, F. (2001). Requirements Engineering as a Success Factor in Software Projects. *IEEE Software*, 18, 4, 58-66.
10. Lamsweerde, A. v. (2000). *Requirements engineering in the year 00: a research perspective*. Paper presented at the Proceedings of the 22nd international conference on Software engineering.
11. Maiden, N., and Robertson, S. (2005). *Integrating Creativity into Requirements Processes: Experiences with an Air Traffic Management System*. Paper presented at the Proceedings of the 13th IEEE International Conference on Requirements Engineering.
12. Pan, G. S. C. (2005). Information systems project abandonment: a stakeholder analysis. *International Journal of Information Management*, 25, 2, 173-184.
13. Pitts, M. G., and Browne, G. J. (2007). Improving requirements elicitation: an empirical investigation of procedural prompts. *Information Systems Journal*, 17, 1, 89-110.
14. Read, A., Callens, A., Nguyen, C., and De Vreede, G. J. (Submitted to AMCIS). Generating User Stories in Groups with Prompts.
15. Rubenstein, D. (2007). Standish Group Report: There's Less Development Chaos Today. *Software Development Times*, from http:// www.sdtimes.com/content/article.aspx?ArticleID=30247
16. Sjoberg, D. I. K., Dyba, T., and Jorgensen, M. (2007). *The Future of Empirical Methods in Software Engineering Research*. Paper presented at the 2007 Future of Software Engineering.
17. Sjoberg, D. I. K., Hannay, J. E., Hansen, O., Kampenes, V. B., Karahasanovic, A., Liborg, N.-K., et al. (2005). A Survey of Controlled Experiments in Software Engineering. *IEEE Trans. Softw. Eng.*, 31, 9, 733-753.
18. Wohlin, C., Höst, M., and Henningsson, K. (2003). Empirical Research Methods in Software Engineering. In *Empirical Methods and Studies in Software Engineering* (Vol. 2765, pp. 7-23): Springer Berlin / Heidelberg.