

Eliciting Required Characteristics for Usable Requirements Engineering Approaches

Kristina Winbladh, Hadar Ziv, and Debra J. Richardson
Department of Informatics
Donald Bren School of Information and Computer Sciences
University of California, Irvine
{awinblad,ziv,djr}@ics.uci.edu

ABSTRACT

It has been reported that many software companies do not use existing requirements engineering approaches. This indicates that there is room and opportunity for improving the usability of existing requirements engineering approaches. This paper describes a market study intended to elicit a set of characteristics that could improve the usability of requirements engineering approaches. The survey is aimed toward software stakeholders such as developers, designers, customers, and managers at various software companies. The survey results are used to define a set of desirable characteristics for usable requirements engineering approaches and to suggest a set of guidelines that could help achieve the desirable characteristics.

Categories and Subject Descriptors

D.2.1 [Software Engineering]: Requirements/Specifications

Keywords

Requirements approaches, survey, usability

1. INTRODUCTION

The goal of Requirements Engineering (RE) is to aid the elicitation, specification, analysis, and validation of stakeholder needs and wishes for a software system. RE is recognized as an important and difficult undertaking, and as such, the software community has produced a wide variety of RE approaches including [3, 4, 5, 6, 7, 8, 9, 10, 11, 12] among others. Despite the availability of RE approaches, a recent study shows that many software development companies do not use “textbook requirements engineering processes” [1]. There are naturally many contributing factors to why existing RE approaches are neglected, e.g., stakeholder preferences, habits, and application domains. This paper explores one contributing factor in more detail, namely usability of RE approaches. Although RE values eliciting stakeholder

needs prior to design and development, the usability needs of RE stakeholders – software developers, designers, managers, customers, etc. – were not fully explored and defined prior to the implementation of many RE approaches. The ultimate goal of this research is to elicit and specify RE stakeholder needs, to determine whether usable RE approaches exist, and if they do not, to design such a RE approach.

This paper reports on a survey of usability characteristics for RE approaches aimed toward RE stakeholders at several software companies. The survey results are used to define a set of desirable characteristics for usable RE approaches and a set of guidelines that could help achieve the desirable characteristics.

2. SURVEY DESIGN

The survey is aimed toward users of RE approaches within various companies with different cultures, application domains, and sizes. Random sampling was achieved through open solicitation for participation through company email lists. The survey was designed using the Goal-Question-Metric (GQM) framework [2]. The goal of the survey is to explore and determine characteristics that improve usability of a RE approach. The overall goal is decomposed into two sets of questions, “Rate & Rank” and “Free text”; and standard statistics are used as metrics to evaluate the results. The “Rate & Rank” questions are used to get feedback on a set of characteristics that we perceive particularly important for usable RE approaches. The “Free text” questions address the incompleteness of our initial set of usability characteristics by asking open-ended questions to gain knowledge about other important usability characteristics. Detailed descriptions of survey design and demographics of participants are available in [14]. The characteristics provided by us are listed below and are divided into three categories: clarity, testability, and manipulability.

1. Clarity: Understandable requirements.
2. Clarity: Unambiguous requirements.
3. Testability: Using requirements for test case creation.
4. Testability: Using requirements for test data selection.
5. Testability: Using requirements for test coverage criteria definition.
6. Testability: Using requirements for test oracle creation.
7. Testability: High quality test artifacts (test cases, test data, test coverage, test oracles).
8. Testability: Easy to use the requirements for various testing activities.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SAC'09 March 8–12, 2009, Honolulu, Hawaii, U.S.A.
Copyright 2009 ACM 978-1-60558-166-8/09/03 ...\$5.00.

9. Manipulability: Easy to initially create the requirements.
10. Manipulability: Easy to maintain the requirements throughout software development.

The participants are first asked to rate the importance of each characteristic and then to rank the characteristic relative to other characteristics within the same category. The combination of rating and ranking provides a better measurement than either one alone. The use of only rating typically generates over-optimistic results because every characteristic seems important and the result is that nothing is more important than anything else. The use of only ranking is problematic because we cannot assume even spacing between each pair of ranked items. When combining the two approaches we can get a better idea of the actual perceived importance of the characteristics. We evaluate the answers to the “Rate & Rank” questions by calculating the average of the ratings and the frequency count of the relative rankings.

The first question in the “Free text” section asks the participants to select three activities, among a list of activities, and/or enter their own activities, that are the most important for a RE approach to support. The second question of the “Free text” section asks the participants to select three characteristics among a list of characteristics, and/or enter their own characteristics, that are the most important for a RE approach to exhibit. Figure 5 and Figure 6 list the choices available for each question. The metric used for both questions is a count of which alternatives are most frequently selected. The remainder of the “Free text” section is a set of questions for which the participants provide free text answers (see list below). The metric for these questions is qualitative analysis of the answers.

1. What requirements approaches/notations do you use?
2. What are some weaknesses of the requirements approaches/notations you use?
3. What characteristics or activities do you wish your requirements approaches/notations would support?
4. What characteristics of a requirements approach/notation do you consider the most important and why?
5. What techniques do you use to validate your requirements, and what are the pros and cons of those?
6. Do you use requirements in testing activities? If so, what techniques do you use and what are the pros and cons of your current process?

We compare the results from rating and ranking and combine these with insights from the “Free text” section to establish a set of desirable characteristics for usable RE approaches as well as propose practical guidelines that could help achieve the characteristics.

2.1 Limitations

The characteristics in the “Rate & Rank” questions are by no means a complete set of characteristics with regard to usability of RE approaches, rather they are a set of characteristics that we think are important but often neglected by existing approaches. The free text questions are intended to get a more complete set of characteristics, beyond the ones covered in the rating questions, by asking the participants to evaluate their current practices as well as describe their needs and wishes. As with any survey, the results of our study are impacted by interpretation and time issues. We

addressed these concerns by explaining the terms used in the questions and by posting the survey online so that the participants can choose when to participate.

3. RESULTS & GUIDELINES

3.1 Participation

The survey was completed by 25 anonymous participants that accessed the survey through an open call for participation. The participants are from several different software companies of different sizes and application domains. The participants consider themselves experienced with software development activities such as requirements engineering, software architecture and design, testing, and planning. More details regarding demographics are available in [14].

3.2 Rate & Rank Results

When comparing the ratings of understandability and unambiguity (see Table 1) to their relative ranking (see Figure 1), we see a slight difference in the two results. The rating shows that the two categories are equally important, but when forced to choose, participants chose understandability to be more important almost twice as often.

Although understandability and unambiguity are both important, we believe that it is often difficult to achieve both. In general unambiguity can be improved by more formality. More formality, however, generally decreases understandability for many software project stakeholders. One of the survey participants described the situation as follows:

The problem is that you can easily have multiple parties read something and say they understand it, yet have different understanding and expectations. This creates problems because you have an artifact that everyone understands, but differently, yet it does not stand out as ambiguous because everyone understands it.

Characteristic	Average rating	Standard dev.
Understandability	4.56	0.58
Unambiguity	4.40	0.82

Table 1: Average rating of understandability and unambiguity.

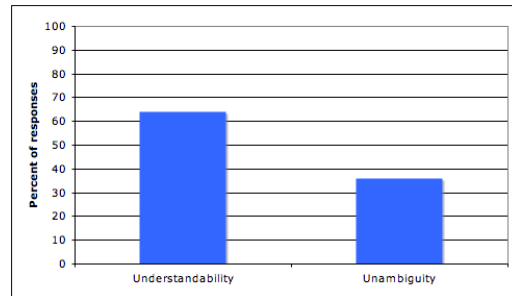


Figure 1: Understandability vs. Unambiguity

On the other hand, many participants pointed out that, although not an optimal solution, unambiguity can be identified and dealt with in later stages of development whereas

understandability is absolutely essential up front. The responses to this question offer a new insight that there are two types of understandability: perceived understandability and true understandability. Perceived understandability is the ease with which stakeholders interpret the requirements, and true understandability is whether stakeholders actually understand the requirements properly. These insights lead to the following desirable characteristics:

Desirable characteristic 1 *Unambiguity.* *It is typically more difficult to achieve unambiguity than perceived understandability, but the advantage is that with reduced ambiguity comes increased true understandability.*

Desirable characteristic 2 *Perceived understandability.* *It is important to recognize that understandability is perceived more important than unambiguity and a RE approach is unlikely to be used unless it offers a high level of perceived understandability.*

Although the two guidelines might seem contradictory, we believe this to be an area open for improvement rather than a necessary correlation. Natural language processing, for example, could yield high marks with regard to both perceived understandability and unambiguity. There are however other challenges with natural language processing making it an unlikely candidate for usable RE approaches.

Table 2 shows the average rating of the importance of a RE approach supporting four different testing activities. Figure 2 shows the response distribution for first to fourth place ranking per testing activity and Figure 3 shows the overall ranking results. The results from rating and ranking correlate in that support for test case creation is perceived the most important, then support for test data selection, then support for test coverage criteria, and test oracle creation is perceived least important. Most survey participants find it highly important that a RE approach supports test case creation and test data selection.

Characteristic	Average rating	Standard dev.
Test case creation	4.68	0.48
Test data selection	3.82	0.96
Test coverage criteria	3.55	1.14
Test oracle creation	3.45	1.37

Table 2: Average rating of the importance of RE support for four different testing activities.

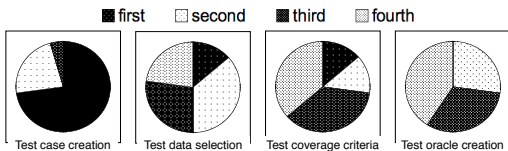


Figure 2: Response distribution per testing activity.

In most companies software testing is mainly a human dependent and human intensive activity. As such, survey participants pointed out that rather than supporting particular testing activities, it is more important that the requirements are clear so that a human tester can use them for testing; this reinforces Desirable Characteristics 1 and

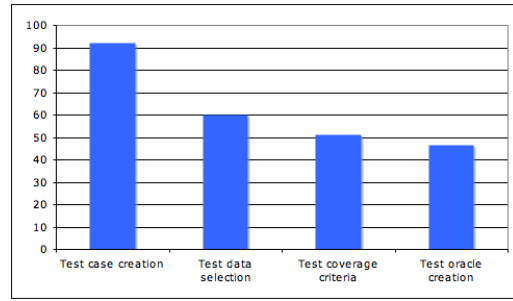


Figure 3: Overall ranking of testability activities.

Characteristic	Average rating	Standard dev.
Initial Ease	3.45	1.10
Maintenance Ease	4.41	0.59

Table 3: Average rating of importance of initial ease to create and maintain requirements.

2. One survey participant pointed out that a more important problem, than generating tests from the requirements, is generating tests at all. Software testing typically does not receive enough resources to be done sufficiently.

The responses regarding testability reinforce our notion that requirements-based testing is quite important. Direct support in the form of automation for at least one or two of the testing activities is advantageous as it provides an additional direct benefit of using the requirements approach as well as it addresses the need for testing efficiency since testing is often under-budgeted. These insights lead to the following desirable characteristic:

Desirable characteristic 3 *Testability.* *Automated requirements-based testing provides support that the software developed meets stakeholder needs and also makes use of existing testing resources effectively and efficiently.*

When comparing the results of importance of initial ease and maintenance ease, both the rating (see Table 3) and ranking (see Figure 4) results show that it is highly important to support maintenance of requirements and that it is more important to support maintainability than initial ease.

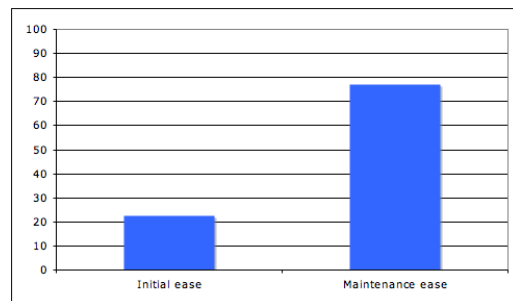


Figure 4: Initial Ease vs. Maintenance Ease

These results are certainly not surprising. Although maintenance is only important once requirements exist, maintenance is often more costly than initial effort and therefore requires better support. One survey participant gave the following motivation (others expressed similar motivations):

Reality: requirements are never completely specified up front, and things change midstream. It is more important to be able to easily update them during a project, because it's almost a certainty that this will be happening often.

Desirable characteristic 4 Maintainability. *Requirements will change throughout the software lifecycle and updating their specification while keeping it correct and consistent should require least possible effort.*

3.3 Free text Results

The free text questions show that approximately 90% of the survey participants use informal textual requirements. Below is a typical response from one of the participants:

The requirements are text in various formats such as documents, spreadsheets, emails, meeting minutes. Sometimes they are worked out during software development (*shudder*). Yeah, this is the stone age, no doubt about it.

When asked about weaknesses of current approaches, most survey participants mentioned lack of formality or guidance when creating requirements; too much ambiguity; that clarity for one group of users seems to be at the expense of another group (developers, designers, customers); and that it is difficult to maintain the requirements as they change.

When asked which characteristics they wished their RE approaches exhibit, the majority of survey participants answered that it has to be quick and easy to create and maintain the requirements. Several also mentioned that the requirements should be clear and unambiguous.

Figure 5 shows activities the survey participants wish their RE approach would support. The four most popular alternatives are: support for maintenance, support for transition to design, automatic test generation, and support for informal conversations with customers. Figure 6 shows the results for the question of what characteristics the survey participants wish their RE approach would exhibit. The most frequently selected alternative is operational descriptions. This alternative is followed by traceability between requirements and other software engineering artifacts, domain descriptions, and links between requirements.

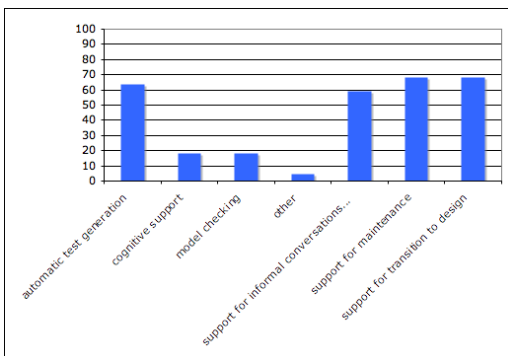


Figure 5: Activities the survey participants wish their RE approach would support.

Most of the answers from the “Free text” questions reinforced the desirable characteristics defined above, but a few brought up new desirable characteristics:

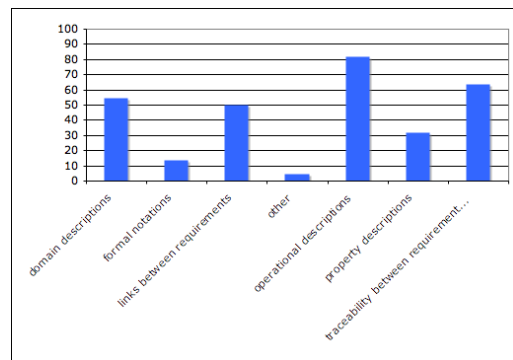


Figure 6: Characteristics the survey participants wish their RE approach would exhibit.

Desirable characteristic 5 Designability. *It is perceived important that a RE approach provides appropriate support for transition to design.*

Desirable characteristic 6 Traceability. *It is perceived important that a RE approach provides traceability support to other software engineering artifacts, e.g., design and tests.*

The desirable characteristics identified above provide high-level non-functional requirements for RE approaches, that should be further refined into more concrete guidelines that could then be used to achieve them. We provide one possible set of such guidelines. Figure 7 shows our guidelines and the desirable characteristics they address. The following paragraphs describe the guidelines in more detail.

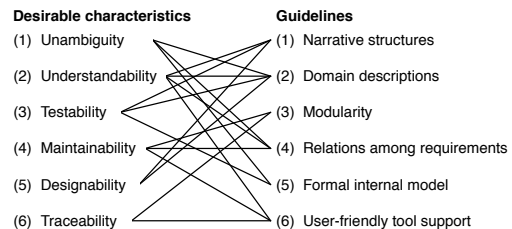


Figure 7: Relations between desirable characteristics and guidelines.

Guideline 1 Strive for narrative structures. Requirements with narrative structures, e.g. use cases and stories, describe operational requirements and can capture informal conversations with customers (highly ranked in Figures 5, 6). Narrative requirements are typically easy to understand, could be the starting point for test scenarios, and message sequence charts (Desirable characteristics 2, 3, 5).

Guideline 2 Strive for domain descriptions. Domain descriptions ranked high in Figure 6. Consistently referencing defined terms in a domain model can decrease ambiguity. Clearly defined terms can also increase understandability. A domain description aids testability as abstract test data can be selected from the model. A domain model can also be a useful starting point for module design. (Desirable characteristics 1, 2, 3, 5).

Guideline 3 *Strive for modularity.* Keeping narrative and domain models separate and decomposing these into related functionalities and concepts, is a useful strategy to provide higher levels of maintainability and traceability (Desirable characteristics 4, 6).

Guideline 4 *Strive for relations among requirements.* Providing syntactic and semantic relations between requirements was highly ranked in Figure 6. Relations between domain and narrative structures can improve unambiguity and understanding of the requirements. Relations can provide guidance for change impact analysis during maintenance. Desirable characteristics 1, 2, 4. Syntactic and semantic relations could also be the basis for a useful RE tool.

Guideline 5 *Strive for a formal internal model.* Providing a formal description of the requirements could increase the level of unambiguity, could be used as a basis for automated test support (Desirable characteristics 1, 3), and was expressed as a need in the “Free text” answers .

Guideline 6 *Strive for tool support.* Tool support could make use of the guidelines above to provide functionality including searching, refactoring, and showing relations. The tool should provide an easy-to-use interface that shields the user from the complexity of internal formal representations. (Desirable characteristics 2, 4, 6).

4. LESSONS LEARNED & FUTURE WORK

Although none of the desirable characteristics in this paper conflict with other lists of requirements properties, the major novelties in this paper are that our characteristics suggest a different set of priorities than previous work and that these priorities are derived from a survey of actual software practitioners. One of the most well-known published lists of requirements properties is the IEEE standard 830 [13], which states that requirements in a software requirements specification should be correct, unambiguous, complete, consistent, ranked for importance and/or stability, verifiable, modifiable, and traceable. Although these properties are important and sometimes overlapping with our desirable characteristics, they are properties of a requirements document and not of the approach used to produce the document and most of the properties in the IEEE standard 830 do not concern the usability of a RE approach. The survey results in this paper clearly show that usability of a RE approach is a major concern for practitioners and should be a focus in RE research. We are unaware of any existing RE approaches that in fact address each of our desirable characteristics or that have been designed with regard to usability.

We intend to address the different requirements by creating a new RE approach, iMuse – Integrated Model-based Use-case Storytelling Environment – whose requirements will be based on the guidelines described above. Particularly iMuse will be used to create narrative requirements in the form of stories, use an ontology to describe the domain with which the system under development interacts and that can be referenced from the stories, and provide an internal and formal representation of stories and domain that can be used for generation of test scenarios. These features will be manifested in a RE environment for creating and manipulating stories through an easy-to-use interface that shields the user from the complexity of internal representations.

We will set up several usability studies with students and industry professionals in order to evaluate how well the prototypes of iMuse fulfill our usability characteristics. These studies can also be used by other RE researchers when evaluating the usability of other RE approaches. During our development of a more usable RE approach we will continue to explore other usability requirements. In particular we will research what characteristics would make a RE approach support the transition from requirements to design. We hope that other researchers in RE will find the usability characteristics presented in this paper useful as a guideline in ongoing and future research efforts.

5. ACKNOWLEDGMENTS

Our deepest gratitude to the participating companies.

6. REFERENCES

- [1] J. Aranda, S. M. Easterbrook, , and G. Wilson. Requirements in the wild: How small companies do it. In *RE'07*, 2007.
- [2] V. R. Basili. Software development: a paradigm for the future. In *COMPSAC'89*, Sep 1989.
- [3] R. Boddu, L. Guo, S. Mukhopadhyay, and B. Cukic. RETNA: From requirements to testing in a natural way. In *RE'04*, 2004.
- [4] A. Borgida, S. Greenspan, and J. Mylopoulos. Knowledge representation as the basis for requirements specifications. *Computer*, 18(4):82–91, Apr 1985.
- [5] J. Castro, M. Kolp, and J. Mylopoulos. Toward requirements-driven information systems engineering: The tropos project. *Information Systems*, 27(6):365–389, 2002.
- [6] A. Cockburn. *Writing effective use cases*. Addison-Wesley Boston, 2001.
- [7] A. Dardenne, A. van Lamsweerde, and S. Fickas. Goal-directed requirements acquisition. *Science of Computer Programming*, 20:3–50, 1993.
- [8] A. Eberlein and J. do Prado Leite. Agile requirements definition: A view from requirements engineering. In *TCRE'02*, Sep 2002.
- [9] D. Firesmith. Specifying good requirements. *Journal of Object Technology*, 2(4):77–87, 2003. http://www/jot.fm/issues/issue_2003_07/column7.
- [10] M. Glinz. An integrated formal model of scenarios based on statecharts. In *European Software Engineering Conference*, 1995.
- [11] M. Jackson. *Problem Frames - Analyzing and structuring software development problems*. Addison-Wesley, 2001.
- [12] C. Potts, K. Takahashi, and A. Antón. Inquiry-based requirements analysis. *IEEE Software*, Mar 1994.
- [13] The Institute of Electrical and Electronics Engineers. IEEE recommended practice for software requirements specifications. IEEE std 830–1998.
- [14] K. Winbladh, H. Ziv, and D. J. Richardson. Surveying the usability of requirements approaches using a 3-dimensional framework. Technical Report UCI-ISR-08-3, UC, Irvine, August 2008.