A Survey of Practices and Problems Associated with Incomplete User Requirements Documents

Ban Al-Ani,

Lecturer, Dept. of Software Engineering, Faculty of Information Technology, University of Technology Sydney, alani@it.uts.edu.au

Abstract

Incompleteness is one of the inherent problems of system requirements represented through informal notations (natural language, diagrams, pictures...etc) and captured in the User Requirement Document (URD). These early requirements need further analysis and refinement to produce a structured document that can then be formally specified.

A survey was conducted to investigate the problems encountered and the common practices applied during requirement analysis. The main purpose of the qualitative survey was to use practitioners' responses to develop an approach to the problem of incompleteness within the context of existing literature. This paper presents the main findings of the interviews conducted with eight individuals employed by six different companies in Australia and in the U.S.A. during a six-month period.

Keywords: *User requirement document (URD), incompleteness, natural language notation, decomposition, prioritisation.*

1. Introduction

Incompleteness is one of the inherent problems of system requirements represented in natural language [15, 7, 16, 11]. Leveson [15] states that this problem is critical because psychologists found practitioners tend to ignore information that is not represented (in the requirements document) during problem solving. She concludes the following:

1. Problem solvers given a short fundamental representation of the problem did "better" than problem solvers given a lengthy yet incomplete representation.
2. A lengthy but incomplete representation impairs developers because they tended to rely on it as a comprehensive and truthful representation.

This paper presents the finding of a qualitative survey conducted through a series of interviews with practitioners in the field of requirement development. The intention was to use this opportunity to gain first hand information of issues relating to the problem of incompleteness.

The following section outlines survey motivation and objectives in greater detail. It also presents a description of topics discussed during the interviews. This is followed by an outline of survey participants and relevant findings. Finally this paper presents the conclusion reached as a result of analysing survey findings.

2. Survey Outline

The survey was conducted as part of the research into the problem of incompleteness of the URD [2]. The main objective of the qualitative survey was to give context to reviewed literature. Yin (1994) states "some survey questions (such as those seeking categorical rather than numerical responses) rely on qualitative, and not quantitative, evidence".

2.1. Choice of Research Strategy and Objectives

Yin (1994) states a survey strategy is preferred when the researcher wants to discover: who, what, how many, where, how much. The survey objectives are listed below:

- **Who** is involved? Investigate the skills of the people involved in the early Requirement Engineering (RE) process. To achieve this objective it was also necessary to find out **who** is traditionally involved in requirement development not just in overseas (where several major surveys were conducted e.g. 5, 8, 17) but also in Australia.
- **What** is done to detect incompleteness and **what** is done after incompleteness is detected? In addition to understanding of **what** the practitioner's current perception of incomplete requirements is. **What** practices are applied to detect and overcome the problem of incomplete requirements? **What** are the problems a requirements engineer faces when addressing incomplete requirements?
- **How many** revisions are usually performed throughout system development and whether requirements are traditionally frozen?
- **How much** time is spent correcting faults.

The survey was motivated by three factors, namely:

1. Previous surveys do not focus on the problem of incomplete requirements [5, 8].

2. A single-purpose survey would help develop the *industry-as-laboratory* approach described by Potts [22] and develop an approach based on practitioners' viewpoints thereby grounding subsequent research in applied practices.
3. A survey would help bridge the gap that exists between research findings and applied practices [29, 22].

The questions posed during the survey interviews were developed based on the issues stated above.

2.2. Survey Questions

Survey interview questions (presented in appendix A) were adapted from Lubars et al [17], which was considered suitable for the following reasons:

- It deals with requirement development and not issues related to software or system development in general (unlike the Curtis et al, 1988 survey).
- It can be adapted to achieve the objectives of this research's survey. The questions can be modified to establish specific practices and problems related to incomplete requirements.
- It was used extensively for a period of over two years and thus its effectiveness was proven. A review of Lubar's et al [17] survey findings demonstrated the effectiveness of the questionnaire at establishing the generic problems encountered and requirement engineering practices [22].
- The questions asked (and made available) are short and to the point. This was important because participants are typically not willing to spend an excessive amount of time answering questions.

The main topics explored in the survey are as follows:

- **General workplace information**

  The questions in this section of the questionnaire are based on Lubar et al's [17] conviction that it can help put participants at ease and in a frame of mind to answer the questions. In addition, general workplace information help establish participant's background and field of expertise.
- **Requirement representation**

  This can identify the preferred notations (formal, semi-formal, and/or informal) used to document requirements, their relative merits and the media used to document requirements (i.e. visual, audio capture).

  The results can be addressed within the context of other related surveys [5, 8]. It would also give context to the adopted practices discussed during the interview.
- **Requirement development and verification**

  Participants were asked what methods and tools are used to verify the complete capture of requirements. This helped identify the indications that led the analyst to investigate the completeness of documented requirements. Finally, the answers to these questions were used to understand the difficulties encountered when attempting to detect incomplete requirements.
- **Requirements Refinement**

  The participants were asked to describe the steps taken to refine an incomplete requirements document and the effectiveness of the tools used (if any). The aim was to give insight into the problems that practitioners face when attempting to refine incomplete requirements based on their perception of the problem.

The answers were documented during the interview, then compiled and analysed.

3. Survey Participants

Eight individuals employed by six different companies were interviewed in Australia and the U.S.A. during a six-month period. Survey interviews lasted for an average of one and half hours. Each of the individuals had more than seven years experience in system development. Their job description (when the interview was conducted) varied from analyst/programmer to project manager/leader.

Each participant discussed an average of two different complex projects, where a team of no less than 5 developers worked for an average of a year to develop the system. The selected projects were diverse and ranged from flight visualisers (built from scratch) to a multi-user system (utilised by a ministry of education) that had to be upgraded.

4. Summary Of Survey Data

A summary of the main insights derived from the survey is presented in the following sections (summary is presented in appendix B).

4.1. Main Practices Applied During Requirement Analysis

The following is a description of the prominent practices that are relevant to this research:

1. **Decomposition**

   Bulky requirement documents are decomposed into smaller manageable requirement blocks. Requirement decomposition is the basic principle of military standards used by many of those interviewed, because they found that it increases requirements readability and traceability. Nelson [19] states that Military Standards (MIL-STD) are written for processes, materials or functions. In many cases, they are written in a generalised form to apply to as many possible situations that the author can think of.
2. **Prioritisation**

   It is common practice to ascribe priority to requirement statements. Several projects, most notably market driven ones, assigned priorities to

requirements because certain features are more important than others and some features or properties can be ignored if deadlines dictated. This is in line with Lubars et al's [7] who also stated that requirement prioritisation is a common practice. Prioritisation is also a recommended practice by several studies [18, 26, 6].

4.2. Main Problems Encountered During Requirement Analysis

Participants collectively conceded that aiming for completeness in requirements is problematic, for the reasons stated below:

- **A stakeholder cannot fully define what is needed until there is something tangible to use and critique**

  It is very difficult for stakeholders to completely identify requirements of large complex systems. Studies by Patel [20], Anton and Potts [3], Hsai et al [12] and Rowen [25], support this finding. They found that the difficulty of *uncovering* requirements is one of the challenges facing practitioners. They agree that while an agreement on terminology can be achieved, the system can remain imperceptible. This could be because not every component of a system has a physical embodiment like a machine tool. Consequently, even well written relatively complete requirement documents must depend on the mental images a description evokes. Rowen [25] found that this could be due to the overall complexity of large systems that places a burden on all written documentation, which are useful only as an unambiguous vehicle of communication.

- **Stakeholders can be unaware of certain aspects of the system being described**

  This can occur because stakeholders are familiar with the domain and take it for granted. It is also possible that stakeholders do not reveal certain aspects because of power relations in organizations. Scharer [27] also found this true in her study of the requirements engineering process. She states that it can become highly political and mentally taxing. This is also supported by ethnographical studies, which revealed that the power structure of an organization (amongst other reasons) might lead to the incomplete capture of requirements. Consequently, there is a move to improve how requirement engineers deal with social and organisational factors. Viller and Sommerville [31] have found that the ethnographer often acts as a "proxy" for the field site, responding to question and suggestions for the actors in the domain.

- **A balance must be achieved between professional responsibility and developing what is being paid for**

  The knowledge that additional fees must be paid to upgrade a system once it has been handed over to stakeholders makes it beneficial (financially) to minimise additions to the requirements document. Consequently, the contents of the requirements document can prematurely stabilise and its evolution can be restrained. Williams and Kennedy's [32] study found that while the requirements document size fluctuated during the first five months of development, it gradually became more stable after that time.

  This has also been recognised in Potts' [24] study. He states: once the contract is signed, practitioners find introducing new requirements means that the developers will either have to re-negotiate the contract or incorporate the new (approved) requirements into the existing document structure. William and Kennedy [32] state that these additions might be time consuming and costly. Sommerville and Sawyer [30] also observe that it is expensive to make changes to requirements after they have been agreed upon.

  Potts [21] voiced concern about prematurely *freezing* requirements should the document be perceived as comprehensive and taken a lot of effort to write (and read).

- **Although complete requirements were stated as being desired, it was also a common policy to exclude details when defining requirements**

  This meant that the development team had more creative freedom and requirements interpretation became more flexible. Absolute completeness was not desired or expected by any of the practitioners who were interviewed. This is in line with Gyezly [10] conviction that attempting to achieve absolute completeness generates its own problems because well-defined requirements reduce flexibility and creativity during development. He also states that the effort needed to develop relatively complete requirements is both time-consuming and costly.

4.3. Participants Perception of Incomplete Requirements

The foremost question asked during the interview was the following:

If you were given a requirements document and told it was incomplete, what would you think was missing?

Three different informal working definitions were established after analysing the participant's attempts

to define requirement incompleteness. They are listed below.

- *A set of requirements is incomplete if a desired part of the system is not explicitly stated and documented.*

  Participants found it difficult to define what constituted a set of requirements, which demonstrated that their perception and consequent definition lacked precision and clarity.

- *A requirement document is incomplete when requirements have been incompletely implemented (in one or more phases of development).*

  While this was one of the given definitions, it is a misinterpretation because incomplete requirement implementation can occur even when the requirements document is complete. For example, an incomplete implementation of requirements captured in the URD can lead to an incomplete system design (although the URD itself is complete). Attempting to achieve a higher level of requirements completeness within the URD will not safeguard against incomplete requirements implementation occurring. Here, incompleteness occurs when one or more *parts* of a system feature (but not all) are excluded from the URD. In contrast, definition 1 refers to the exclusion of a whole system feature description from the URD.

- *A requirement statement is incomplete when the service or constraint it attempts to describe is insufficiently captured.*

  The overall assumption was that captured information was sufficient (and therefore acceptably complete) when requirements needed no further elaborations because all the desired details had been captured. The participants were unable to define the meaning of sufficiency or define what constituted as an elaboration.

5. Conclusions and Suggestions for Future Work

Table 1 presents a summary of survey findings and associated conclusions. This summary can provide insight to the areas of concern, thus achieving the objectives of the survey.

Survey findings cannot be considered statistically conclusive, based on the limited number of participants [8]. Further work is required to provide a means to estimate the cost of incompleteness, its effects and the most effective method applied by practitioner to detect and overcome this problem. Expanding the survey to include a larger number of participants can give a more comprehensive depiction of the problem being investigated.

Table 1: Summary of survey findings and conclusions. *Survey findings and derived conclusions focus on the problem of incomplete requirements documented in natural language.*

Findings	Conclusions
Different working definitions of incomplete requirements were presented. However, the definitions are not clear or precise.	There exists a need for a rigorous (ideally formal) definition of incomplete requirements.
More than one type of incompleteness could exist in a requirements document. This finding is supported by research conducted by Jaffe [13] and Alagar and Kourkopolous [1].	To define the problem the following should be addressed: • Acknowledge that more than one type of requirements incompleteness could occur within a single requirements document. • Attempt to define all categories of incompleteness that could occur within a requirements document.
Participants stated the most common notations used to document requirements are not formal notations (i.e. informal and semi-formal). This finding is supported by studies in this field [23, 9, 12].	A definition of incompleteness can be based on the premise that requirements are documented through informal notation.
One practice is to decompose requirements into smaller more manageable portions. This finding complies with those presented in existing literature [14, 4, 19].	An attempt to define incompletely documented requirements can be based on the premise that requirements are decomposable.

6. Bibliography

[1] Alagar, V., Kourkopoulos, D., (1994). "(In)completeness in Specifications", Information and Software Technology, Vol. 36, No. 6, pp. 331-342.

[2] Al-Ani, B., (2001). "A Framework for the Detection and Evaluation of Partially Complete Requirement Document", Submitted Doctor of Philosophy Dissertation in Computer Systems Engineering, Faculty of Engineering University of Technology Sydney.

[3] Anton, A., Potts, C., (1998). "The Use of Goals for Surfacing Requirements for Evolving Systems", International Conference on Software Engineering (ICSE'98), 19-25 April, pp. 157-166.

[4] Arora, V., Greer, J., Tremblay, (1992). "A Framework for Capturing Design Rationale Using Granularity Hierarchies", International workshop on CASE, Montreal, Canada, 6-10 July, pp.246-251.

[5] Curtis, B., Krasner, H., Iscoa, N., (1988). "A Field Study of the Software Design Process for Large Systems", Communications of the ACM, Vol. 31, No 11, Nov. 1988, 1268- 1287.

[6] Dorfman, M., Thayer, R., (1990). "Standards, Guidelines, and Examples on System and Software Engineering", edited by Dorfman et al, IEEE Computer Society Press, California, Jan. 1, Chap. 1-3.

[7] Edwards, M. L., Flanzer, M., Terry, M., Landa, J., (1995). "RECAP: A Requirements Elicitation, Capture, and Analysis Process Tool for Large Complex Systems", IEEE CS Press, Loss Almitos, California, Aug., pp. 278-281.

[8] El Emam, K., Madhavaji, N., (1995). "A Field Study of Requirements Engineering Practices in Information Systems Development", IEEE CS Press, Loss Almitos, California, July, pp. 68-80.

[9] Fantechi, A., Gnesi, A., Carenini, M., Vanoochi, M., Morschini, P., (1994). "Assisting Requirement Formalisation by means of Natural Language Translation", Formal Methods in System Design,4th ed., Kluwar Academic Publishers, Boston, pp. 243-263.

[10] Gyeszly, S., (1991). "Lack of a Comprehensive Requirements' Definition Results in Never-Ending Costly Product Development", International Journal of Project Management, V. 9, N. 3, pp. 179-182.

[11] Heninger, L. K., (1979) "Specifying Software Requirements for Complex Systems: New Techniques and their Applications", Proceedings of SRS (Specifications for Reliable Software). IEEE Computer Society, New York, pp. 1-14.

[12] Hsia, P., Davis, A., Kung, D., (1993). "Status Report: Requirements Engineering", IEEE Software, November, pp. 75-79.

[13] Jaffe, M., (1988). Completeness, Robustness and Safety in Real-Time Software Requirements Specifications: A Positivist Looks at Requirements Engineering, Doctor of Philosophy Dissertation in Information and Computer Science, University of California, Irvine

[14] Kirkman, D., (1998). "Requirements Decomposition and Traceability", Requirements Engineering Journal, Springer-Verlag London Limited, 3:107-114.

[15] Leveson, N., (1998). "Intent Specifications: An Approach to Building Human-Centred Specifications", Proceedings of the 3rd International Conference on Requirements Engineering, April 6-10, Colorado Springs, Colorado, IEEE CS Press, Los Alamitos, California, pp.204-213.

[16] Lindersttrom, D., R., (1993) "Five Ways to Destroy a Development Project", IEEE Software, pp. 55-58.

[17] Lubars, M., Meredith, G., Potts, C., Ritcher, C., (1992). "Object-Oriented Analysis for Evolving Systems", Proceedings on the International Conference, ACM, Vol 14, pp. 173-185.

[18] Moisaidis, F., (2000). "Prioritising Scenario Evolution", Proceedings IEEE International Conference of Requirements Engineering, June.

[19] Nelson, D., (1990). "System Engineering and Requirement Allocation", IEEE CS Press, June pp 60-68.

[20] Patel, N., (1999). "The Spiral of Change Model for Coping with changing and Ongoing Requirements", Requirements Engineering Journal, Springer-Verlag London Limited, 4:77-84.

[21] Potts, C., (1991). "Seven (Plus or Minus Two) Challenges for Requirements Research", IEEE Software, Sept., pp. 256- 259.

[22] Potts, C., (1993) "Software Engineering Research Revisited", IEEE Software, Sept., 256-259.

[23] Potts, C., (1997). "Requirements Models in Context", Proceedings of the 3rd IEEE International Symposium on Requirement Engineering, IEEE Computer Society Press, 6-10 Jan, pp. 102-104.

[24] Potts, C., (1999). "From Requirements Engineering to Design for Usefulness", Mini-Tuturial, IEEE International Symposium on Requirement Engineering June 7-11, University of Limerick, Limerick, Ireland, pp. 189.

[25] Rowen, R., (1990). "Software Project Management Under Incomplete and Ambiguous Specification", IEEE Transactions on Engineering Management, Vol. 37,No. 1, Feb., pp. 10-20.

[26] Ryan, K., Karlsson, J., (1997). "Prioritizing Software Requirements in an Industrial Setting", Communication of the ACM, pp. 564-565.

[27] Scharer, L., (1993). "Pinpointing Requirements", Proceedings of IEEE International Symposium on Requirements Engineering, CA, USA, Jan, pp. 17-22.

[28] Sibley, E., Wexelblat, R., Michael, J., Tanner, M., and Littman, D., (1992). The Role of Policy in Requirements Definition, IEEE CS Press, January, pp. 227- 280.

[29] Siddiqi, J., Shekran, C. (1996)," Requirements Engineering The Emerging Wisdom", IEEE Software, March, V. 13, No. 2, pp. 15-19.

[30] Sommerville, I., Sawyer, P., (1997). Requirements Engineering: A Good Practice Guide, John-Wiley and Son, New York.

[31] Viller, S., Sommerville, I., (1999). "Social Analysis in the Requirements Engineering Process: From

Ethnography to Method", IEEE International Symposium on Requirement Engineering June 7-11, University of Limerick, Limmerick, Ireland, pp. 6-13.

[32] Williams, D., Kennedy, M., (1999). "A Viewpoints Conceptual Framework for Improving the Organisational Information Requirements Process: A System Dynamic Perspective", http://www.scism.sbu.ac.uk/~willidw/viewpoints.html

[33] Yin, R., (1994). Case Study Research: Design and Methods, 2nd edition, Sage Publications.

Proceedings

6th Australian Workshop on Requirements Engineering

AWRE 2001

ISBN 0 7334 1859 7

Additional copies may be ordered from
The Centre for Advanced Software Engineering Research
UNSW
Sydney, 2052, Australia.

Table of Contents

6th Australian Workshop on Requirements Engineering

Foreword

Welcome to the Sixth Australian Workshop on Requirements Engineering (AWRE'2001).

This workshop began in 1996 at Monash University and has been held every year since then. This year we see the venue changed and the sponsorship for the conference move from Macquarie University to The University of New South Wales. The aim of the workshop has remained constant – to provide a forum to bring requirements engineering researchers and practitioners together to exchange ideas and experiences.

Of the nineteen papers submitted to the workshop, fourteen were accepted for publication in the proceedings. All submitted papers were reviewed by three members of the program committee. In addition to papers sessions, we have a keynote address and two panel sessions devoted to requirements management tools and empirical research in requirements engineering.

We hope that participants find all of the workshop contents interesting and inspiring, and that it helps identify new ideas that will have an impact on your research in the future.

Finally, our thanks go to the members of the program committee for their work in ensuring a good selection of papers, to John Shepherd for technical assistance with the web site, and Liz Woordward for thorough administrative support during the planning and execution of the workshop. Special thanks also to Didar Zowghi who carries the workshop memory that allowed us to follow on from previous successful meetings.

Aybüke Aurum

Program Chair
School of Information Systems, Technology and Management

Ross Jeffery

General Chair
School of Computer Science and Engineering