

The Factors of Software Systems that Contribute to Requirements Elicitation

Allison Scogin
Mississippi State University
Department of Computer Science and Engineering
PO Box 9637 Mississippi State, MS 39762

Abstract

Requirements elicitation describes the activities needed to determine the requirements of a software system. It is essential to the success of a software development project. Requirements elicitation techniques are used by requirements engineers to discover the information needed to build systems that will satisfy the needs of stakeholders. This paper presents different types of software systems and the factors that would affect the selection of attributes needed to determine appropriate elicitation techniques. These attributes could then be used to determine elicitation techniques that would be successful for different project types.

Keywords: Requirements Engineering, RE Process, Elicitation, Software Systems

1. Introduction

Software systems are often late and over budget. According to reports from the Standish Group, more than half of all products created by the software industry fail to satisfy the needs of stakeholders [4, 16]. Stakeholders are people or organizations that will be affected by the system and can include customers, developers, and users of the software system [11]. If the requirements of a system can be correctly determined early in the development process, then the project has a greater chance of being successful. A requirement is a description of what should be implemented in a software system [7]. Requirements engineering is a process that identifies, documents, and validates the requirements for the software system [11].

Requirements elicitation is the process of discovering the requirements of a software project [5]. The requirements are discovered using requirements elicitation techniques. Elicitation techniques are methods used by requirements engineers to determine the needs of stakeholders [4]. Elicitation techniques are used throughout the RE process to guarantee that the requirements are complete, consistent and correct. Selecting a technique for elicitation is often difficult. There are many techniques available, but not all of these techniques are suitable for every type of software project.

This paper explores four very different types of software systems and five general factors that influence these systems. The software systems are COTS, safety critical, security critical, and web services. The factors are the relationship between the customer and the developer, the skill set of the developer, the necessary domain knowledge, the time constraints of the system, and the cost constraints of the system. These factors can affect the requirements elicitation differently for each type of software project. The factors can drive a requirements engineer to pick specific attributes for the elicitation process. The technique decided to be most successful can be determined based on these attributes.

The next section briefly describes the requirements engineering process. Section 3 explores the four software systems and the contributing factors. Section 4 includes the conclusion and further research. Section 5 concludes with acknowledgments.

2. The RE Process

Requirements engineering is a process that identifies, documents, and validates the requirements for a software system [11]. The requirements engineering process consists of four main areas: elicitation, analysis, specification, and validation [7]. The elicitation of requirements is the activity which should occur first in the RE process [11]. A requirement is a description of what should be implemented in a software system [7]. If the

requirements of a system can be correctly determined early in the development process, then a project has a greater chance of being successful. If the requirements are wrong, software systems become late and unreliable. A software project is deemed unsuccessful if the stakeholders are not satisfied with the end product. Therefore, it is crucial to the success of a software system that the requirements be correct.

Requirements elicitation is the process of discovering the requirements of a software project [5]. The requirements are discovered using requirements elicitation techniques. Elicitation techniques are methods used by requirements engineers to determine the needs of stakeholders [4]. Elicitation techniques are used throughout the RE process to guarantee that the requirements are complete, consistent and correct. Selecting a technique for elicitation is often difficult. This is especially true when one considers the variety of software systems currently being developed.

3. Types of Software Projects and Contributing Factors

Four very different types of software systems were considered: COTS, safety critical, security critical, and web services. Five general factors that influence these systems were also considered: the relationship between the customer and the developer, the skill set of the developer, the level of domain knowledge required, the time constraints of the project, and the cost constraints of the project. These factors can affect the elicitation of requirements differently for each type of software project. The types of software systems will be considered one at a time. The factor deemed most important will be discussed first followed by the remaining factors in decreasing importance.

3.1 COTS

Commercial-off-the-Shelf (COTS) software describes generic software components that contain fixed functionality [19]. The most important contributing factor for a COTS system would be the time constraints on product development. Because COTS software promises a faster time-to-market, the software products are able to reach customers and users quickly [12, 19]. Time-to-market matters when several competitors are working on similar products simultaneously. Cost constraints should then be considered. By using COTS, a developer can be saved from “reinventing the wheel.” This means that by taking advantage of COTS, the overall development cost can be reduced [19]. Developer skills are relatively unimportant; developers only need to have enough knowledge to write the “glue code” to piece the software components together. Knowledge of the software project domain is not crucial as a contributing factor because COTS is meant to be generic and not specific to a particular domain. Finally, customer relationships are considered least important for the following reason: the software already contains a fixed functionality. The developer and the customer do not have to frequently discuss the details of the software development.

3.2 Safety Critical

Madan defines a safety critical system as “a system that must exhibit, with very high assurance, some specific qualities such as safety, reliability, confidentiality, integrity, availability, trustworthiness and correctness” [9]. The software systems that control medical devices, nuclear power plants and aircraft are all considered safety critical systems. It is obvious from these examples, that failure would be disastrous [13]. Safety critical systems should always perform as desired and should never fail [9]. Knowledge of the domain is therefore considered the most important contributing factor. The software developer must fully understand the environment that the safety critical system will operate in for successful system development. Developer skill sets are the next crucial factor. Developers must be highly trained and experienced in developing safety critical systems. Next, a tremendous amount of time is needed to verify that the system is safe. Also, ensuring a high level of assurance against failure makes safety critical systems very expensive to develop. The factor considered least important is the relationship between the customer and the developer. The developer must learn exactly what the system must do so they do not develop a product that is dangerous to users. Although, the customer/developer relationship is deemed the least important factor, it is still crucial to the goal of a successful safety critical system.

3.3 Security Critical

A security critical system is concerned with the confidentiality, integrity, and availability of assets [15]. Security critical systems are governed by security policies. The Trusted Computer Systems Evaluation Criteria (TCSEC) Glossary defines a security policy as "...the set of laws, rules, and practices that regulate how an organization manages, protects, and distributes sensitive information" [17]. Hence, complete knowledge of the domain is the most important contributing factor. Next, developer skill sets are critical to the success of a security critical software system. Like safety critical systems, developers must be highly trained and experienced in the design and development of software systems. Cost constraints are considered to be the third most important factor. Stakeholders must determine both the cost of the assets and the cost of a potential security breach. The stakeholders must then determine how much money they are willing to spend to protect the assets. Time constraints are the fourth factor. A substantial amount of time is needed to verify that the system is as secure as possible. Finally, the relationship between the customer and developer is considered least important, although it is critical for the success of the system. For example, without proper communication, the developer could possibly design a software system unable to work with the current operating system.

3.4 Web Services

According to Leavitt, "web services are a framework of software technologies designed to support interoperable machine-to-machine interaction over a network" [8]. Developing a web service can be very costly. An entire infrastructure must be designed and implemented with underlying hardware and software. There are software development costs, data center operation costs, backup system costs, and licensing for the use of operating systems [1]. It is for these reasons that cost constraints are considered the most important contributing factor. Time constraints are considered the second most critical factor. Because the initial set-up can be lengthy, time-to-market can suffer. If the developers are concerned with getting their project on the market quickly, they should consider the time needed to get established with web services. Next, the developer must have extensive skill sets and be very knowledgeable about the domain. There is much complexity in developing the infrastructure and fully understanding the standards [1]. Customers of web services also need to communicate with developers to ensure that their implementations are compatible with the web service systems of other vendors. If the implementations are not compatible, then additional work will have to be done which can further increase the time-to-market [8].

4. Conclusions and Future Work

Elicitation techniques are used throughout the RE process to guarantee that requirements are complete, consistent and correct. Selecting a technique for elicitation is often difficult. There are many techniques available, but not all of these techniques are suitable for every type of software project [11]. This paper described four different types of software projects and five general factors that influenced these systems. The software systems were COTS, safety critical, security critical, and web services. The factors were the relationship between the customer and the developer, the skill set of the developer, the necessary domain knowledge, the time constraints of the project, and the cost constraints of the project.

Table 4.1 summarizes the results from Section 3. The factors are ordered from 1 to 5, with 1 representing

Table 4.1 Software Systems vs. Factors Affecting Elicitation

Software Systems	Factors Affecting Elicitation				
	Domain knowledge	Customer/Developer Relationship	Developer Skill Sets	Time	Cost
COTS	4	5	3	1	2
Safety Critical	1	5	2	3	4
Security Critical	1	5	2	4	3
Web Services	4	5	3	2	1

what the author considered most important and 5 representing what the author considered least important. The factors can drive a requirements engineer to pick specific attributes for the elicitation process. The technique decided

to be most successful can be determined based on these attributes. There are many available techniques for requirements elicitation (e.g., surveys, interviews, prototyping, etc.) [4] and the technique should be chosen based on the software project's attributes. This paper does not explore the possible attributes or techniques but leaves that as a future research opportunity.

5. Acknowledgements

The author would like to thank Professor Ray Vaughn and the National Science Foundation for funding the author's graduate studies.

References

- [1] A. Bosworth, "Developing Web Services," *Proceedings: 17th International Conference on Data Engineering*, Heidelberg, Germany, Apr. 2001, IEEE Computer Society, pp. 477-481.
- [2] P. T. Devanbu and S. Stubblebine, "Software Engineering for Security: A Roadmap," *Proceedings of the Conference on The Future of Software Engineering*, Limerick, Ireland, May 2000, IEEE Computer Society, pp. 227-239.
- [3] A. M. Hickey and A. M. Davis, "Requirements Elicitation and Elicitation Technique Selection: A Model for Two Knowledge-Intensive Software Development Processes," *Proceedings: 36th Hawaii International Conference on System Sciences*, Jan. 2003, Hawaii, IEEE Computer Society.
- [4] A. M. Hickey and A. M. Davis, "Elicitation Technique Selection: How do Experts do it," *Proceedings: 11th IEEE International Requirements Engineering Conference*, Sept. 2003, Monterey Bay, California, IEEE Computer Society, pp. 169-178.
- [5] A. M. Hickey and A. M. Davis, "A Unified Model of Requirements Elicitation," *Journal of Management Information Systems*, vol. 20, no. 4, 2004, pp. 65-84.
- [6] L. Jiang, A. Eberlein, and B. H. Far, "Combining Requirements Engineering Techniques – Theory and Case Study," *Proceedings: 12th IEEE International Conference and Workshops on the Engineering of Computer-Based Systems*, Greenbelt, Maryland, Apr. 2005, IEEE Computer Society.
- [7] G. Kotonya and I. Sommerville, *Requirements Engineering: Processes and Techniques*, John Wiley and Sons, 1998, pp 3-10.
- [8] N. Leavitt, "Are Web Services Finally Ready to Deliver?," *IEEE Computer*, vol. 37, issue 11, Nov. 2004, pp. 14-18.
- [9] S. Madan, "Techniques to Facilitate Development of Safety Critical Software Systems," *Proceedings: Canadian Conference on Electrical and Computer Engineering Conference*, St. John's, Newfoundland, Canada, May 1997, IEEE, pp. 249-252.
- [10] C. McPhee and A. Eberlein, "Requirements Engineering for Time-to-Market Projects," *Proceedings: 9th Annual IEEE International Conference and Workshops on the Engineering of Computer-Based Systems*, Lund University, Lund, Sweden, Apr. 2002, pp. 17-24.
- [11] B. Nuseibeh and S. Easterbrook, "Requirements Engineering: A Roadmap," *Proceedings of the Conference on The Future of Software Engineering, International Conference on Software Engineering*, Limerick, Ireland, 2000, IEEE Computer Society, pp. 35-46.

- [12] M. Ochs, D. Pfahl, G. Chrobok-Diening, and B. Nothhelfer-Kolb, "A Method for Efficient Measurement-Based COTS Assessment and Selection - Method Description and Evaluation Results," *Proceedings: 7th International Symposium on Software Metrics*, London, England, Apr. 2001, IEEE Computer Society, pp. 285-296.
- [13] D. L. Parnas, A. J. V. Schouwen, and S. P. Kwan, "Evaluation of Safety-Critical Software," *Communications of the ACM*, vol. 33, issue 6, Jun. 1990, pp. 636-648.
- [14] F. Paetsch, A. Eberlein, and R. Maurer, "Requirements Engineering and Agile Software Development," *Proceedings: 12th International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises*, 2003, IEEE Computer Society, pp. 208.
- [15] C. P. Pfleeger and S. L. Pfleeger, *Security in Computing*, 3rd Edition, Prentice Hall, 2003, pp.10.
- [16] The Standish Group, "The Chaos Report, " 1995, and "Chaos: A Recipe for Success, " 1999, http://www.standishgroup.com/sample_research/index.php (current Nov 10, 2005).
- [17] TCSEC: Department of Defense Trusted Computer System Evaluation Criteria. Dept. of Defense Standard, Department of Defense, Dec. 1985.
- [18] W. T. Tsai, X. Wei, Y. Chen, B. Xiao, R. Paul, and H. Huang, "Developing and Assuring Trustworthy Web Services," *Proceedings: 7th International Symposium on Autonomous Decentralized Systems*, Chengdu, Jiuzhaigou, China, Apr. 2005, IEEE Computer Society, pp. 43-50.
- [19] J. Voas, "COTS Software – The Economical Choice?," *IEEE Software*, Mar. 1998, pp. 16-19.
- [20] J. Voas, "The Challenges of Using COTS Software in Component-Based Development", *IEEE Computer*, Jun 1998, pp. 44-45.