Journal of Software Engineering and Applications, 2014, 7, 218-224
Published Online April 2014 in SciRes. http://www.scirp.org/journal/jsea
http://dx.doi.org/10.4236/jsea.2014.74023

Proposing a Systematic Approach to Verify Software Requirements

Zuhoor Abdullah Salim Al-Khanjari

Department of Computer Science, College of Science, Sultan Qaboos University,
Muscat, Oman
Email: zuhoor@squ.edu.om

Received 28 February 2014; revised 22 March 2014; accepted 29 March 2014

Abstract

Identifying stakeholder's needs, eliciting, categorizing and translating them into specifications is the requirement analysis process. Requirement analysis can be a long and arduous process during which many delicate psychological skills are involved. For any software, it is important to identify all stakeholders, collect their requirements and ensure they understand the implications of the software. The gap between stakeholders' vision of the proposed software and the analysis's depiction of that software is the cause of shortcomings in analysis. If the requirements specified by analysts can be tested against stakeholders' expectations, then this gap might be narrowed, and better solutions might be resulted. This paper discusses the impact of the activities of the analysis phase on the development process and on the software itself. It describes the development of the ReqVerifier tool and proposes a systematic approach on how to test software requirements and verify them against stakeholders' vision in order to develop a good software requirement for a quality software.

Keywords

Requirement Analysis, Requirement Elicitation, Requirement Management, Requirement Testing, CASE

1. Introduction

Software requirements are actually customer's statements of scope. They are defined as a set of guidelines, conditions, capabilities or abilities that a system must provide, conform or be consistent with. They deal with functional and nonfunctional requirements of a system. They should be documentable, actionable, measurable, testable, traceable, related to identified business needs or opportunities and defined to a level of detail sufficient for software design.

In requirement finalization process, the stakeholders play an important role. A stakeholder can be defined as anyone who is directly or indirectly affected by the system being developed or deployed. Stakeholders are broadly classified into two major categories: 1) user, who uses the system and 2) customer, who requests for the development of the system and be responsible for approving it. Requirement analysis plays an important role in the software development process. It serves as a baseline for any software project. It must consider the system in all of its stages. From the small units (e.g. objects or modules) to the integration phase, it fits in a broader and a bigger picture. Requirement analysis is critical to the success of a software project. It depends on the correctness and completeness of the software requirements [1] [2]. Failing to successfully analyze the requirements will be fatal for the software at hand and will lead to problems in delivering the required products on time with the right quality. Realizing a high quality application is a tedious development activity. Many projects fail because the analysis phase was immature and did not fully address the system's specifications. A number of studies conducted on this issue have reported that the main reason of project failure is due to employing bad requirement engineering practices. In [3], it has been reported that 70% of the systems errors are ordinarily due to the improper requirements and remaining 30% are because of the design faults.

The introduction of modern technological applications can enhance and improve the software development process. In the IT community, this was realized through the usage of computer aided software engineering (CASE) tools. CASE tools are being used to facilitate activities during software development process [4]. As a result, productivity can be increased and quality can be improved. CASE tools can help in managing the complex tasks of the requirement analysis process.

This paper introduces the activities of the analysis process and the impact of each activity on the development process and on the software itself. It proposes a systematic approach and describes the development of the ReqVerifier tool to test and verify the analyzed requirements of the software against the stakeholders' needs and expectations.

This paper is structured into five sections. Section 2 introduces the literature review. Section 3 discusses the requirement analysis and its related issues. Section 4 demonstrates the development of the ReqVerify tool. Finally, Section 5 provides the concluding remarks and future prospects.

2. Literature Review

The requirement analysis is an important phase in the software development process. In 1992, Christel and Kang [5] identified the following problems that indicate the challenges of the requirement analysis:

1) **Problems of scope:** The boundary of the system is ill-defined.

2) **Problems of understanding:** The users are not completely sure of what is needed?

3) **Problems of volatility:** The requirements change over time. The rate of change is sometimes referred to as the level of requirement volatility.

In 1997, Alfeche [6] published a guide on good requirement engineering practices. He considered requirement elicitation as a major part of a systems analysis.

In 1998, Kotonya and Sommerville [7] have mentioned that any software development process has to involve some requirement analysis.

In 2001, Pressman [4] specified that the analysts job is to assure that proper function of the system will be achieved, and that the requirement analysis process is intensified and focused specifically on software.

In 2003, Wiegers [8] stated that most IT systems fail to meet expectations because the requirements didn't address the right issues. He showed how to collect and present a good requirement specification.

In 2005, Stellman and Greene [9] described requirement management to be the process of documenting, tracing, prioritizing and agreeing on requirements and then controlling change and communicating to relevant stakeholders. In the same year, Abran and Moore [10] stressed the fact that requirement analysis involves frequent communication with system users to determine specific feature expectations, resolution of conflict or ambiguity in requirements as demanded by the users.

In 2007, Gorschek and colleagues [11] indicated the importance of analysis in comparing incoming requirements to the production strategies of the organization. In the same year, Zagajsek and colleagues [12] proposed a requirement management process model for software based legacy systems. They mentioned that using incomplete requirement specification and adopting improper requirement management techniques are the main reason behind any failed project.

In 2008, Jiang [13] presented requirement engineering process as a research object by proposing a formal method for describing the processes. He also highlighted different tools and technologies that can be used during the requirement engineering process. In the same year, Cheng and Liu [14] presented a framework, which uses view-based requirement elicitation techniques to merge different viewpoints of the stakeholders. Also in this year, Mishra and colleagues [15] highlighted the importance of requirement engineering and indicated that the backbone for the project success is selecting the right technique.

In 2009, Berkovich and colleagues [16] wrote about requirement analysis, specifically about validation and verification and stakeholder integration in the development process. They explained that requirement validation and verification ARE the process of checking whether the requirements, as identified, do not contradict the expectations about the system of various stakeholders, and do not contradict each other. It is sort of a requirement quality control.

In 2010, Pandey and colleagues [17] emphasized that it is extremely important to execute planning phase independently to achieve an effective requirement management. In the same year, Pavanasam and colleagues [18] proposed a model for requirement inspection and stated that the requirements change management and requirement analysis phases play an important role in software development process.

In 2012, Torkar and colleagues [19] described how analysis can predict the impact of requirement change. Also in 2012, Gorschek and colleagues [20] have concluded that if analysis is performed, then implementation costs and resources can be estimated for requirement comparison and prioritization.

In 2013, Khan and colleagues [21] discussed software requirement issues and provided a review on how to manage them. They studied existing methodologies for resolving and managing software requirements.

3. Requirement Analysis

Requirement analysis is an early stage in the more general activity of requirement engineering which encompasses all activities concerned with eliciting, analyzing, documenting, validating and managing software or system requirements [7]. These activities are described in the following sections.

3.1. Requirements

Requirements are actually customer's statement of scope or interest. One peculiar aspect of human nature is that we fail to understand each other completely since the human communications are imprecise naturally. Requirement gathering is also a form of human communications in which an attempt is made to pass on complex ideas from one mind to another. Requirements are also a sparse form of communications, using simple written words to attempt for precision [22]. These requirements must be quantifiable, relevant and detailed. In software engineering, requirements could be divided into functional and non-functional requirements.

3.2. Requirement Elicitation

Requirement elicitation is non-trivial because you can never be sure you get all requirements from the users by just asking them what the system should do. Requirement elicitation practices include interviews, questionnaires, user observation, workshops, brain storming, use cases, role playing and prototyping. Requirement elicitation is essential because it is the source of raw data about the problem domain [6].

3.3. Requirement Management

Requirement management is a continuous process throughout any project. Its purpose is to ensure that an organization documents verify and meet the needs and expectations of its users and internal or external stakeholders [9].

3.4. Requirement Change Control

A good process for managing change is essential. Lack of process change control leads to confusion and accordingly leads to bad quality products, overruns and overspends. Change control in requirement management is the process by which any needed changes to the requirement baseline are managed. Whether the change is big and complex or small and simple they still need to travel the same initial route into the change control process. With this in mind, it seems that change is inevitable and therefore a systematic way is needed to deal with it. To create awareness, this process should be agreed on and in place before the baseline is agreed upon.

3.5. Requirement Validation & Verification (V & V)

V & V phase is critical to successful system product development. It is necessary because the designers and implementers of computer based systems are human; who could make errors. These errors might result in undetected faults in delivered systems. Faults can result in dangerous failures causing loss of life, financial loss or property damage. The mission of V & V is therefore to find and correct errors as early as possible in the development life cycle [8] [16].

3.6. Role of an Analyst

To understand the nature of the program(s) to be built, the analyst must understand the information domain for the software, as well as required function's behavior, performance, and interface [11]. Analyst should ensure that the final system or product fits users' needs [10].

4. Requirement Verifier (ReqVerifier) Tool

The gap between stakeholders' vision of the proposed solution and the analysts' depiction of that solution is the cause of shortcomings in analysis. To narrow this gap, researchers were challenged to find a way to verify the requirement specifications against stakeholders' expectations. As a step forward in this direction, the current author proposes the ReqVerifier tool, which is developed for this purpose. The intension of the current author is to make the design of the tool simple and straight forward.

The tool depends on three perspectives: 1) analyst perspective; 2) stakeholder perspective and 3) tester perspective. The testing team could use the ReqVerifier tool to test and verify the requirement specification prepared by the analysts against the results and feedback on the requirement specifications received by the stakeholders. The ReqVerifier tool offers three main processes in accordance to the three perspectives: 1) Manage analysts' perspective; 2) Manage stakeholder perspective and 3) Manage tester perspective. These processes are further discussed below. Using this tool, the analysts, software developers and testers are expected to gain some knowledge about the product, which is to be developed. It could help them in making their discussion with the stakeholders more organized and focused.

4.1. Managing Analysts Perspectives

With respect to the analysts' perspective, analysts use the ReqVerifier tool to prepare two types of tests. The first type is true or false statements about the system and its functional and non-functional specifications. The second type is a scenario based test. This test will introduce input values to a predefined process and the user is requested to input the value or the behavior of the system, which he expects the system should be able to do.

These tests are uploaded to and stored in the database of the ReqVerifier tool. Also, in this perspective, analysts prepare and store the expected answers of these tests. Figure 1 shows the analysts workflow.

4.2. Managing Stakeholders' Perspective

With respect to this perspective, stakeholders download the two types of tests, which were prepared by the ana-

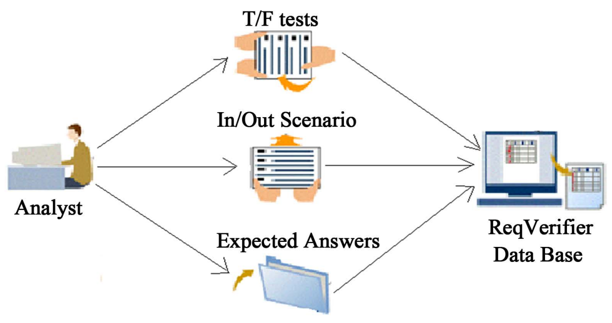

Figure 1. Analyst's workflow.

lysts and stored in the ReqVerifier database. For the first type of tests (e.g. True/False questions), the stakeholders provide the answers and store them in the ReqVerifier database. However, for the second type of tests (e.g. the Input Output Scenario test), the stakeholder could download the multiple input-process-output scenarios for verification. The stakeholder does a series of input-process-output scenarios pre-loaded by analysts. The stakeholder stores the answers in the ReqVerifier database. Stakeholders can also add extra feedbacks about the specifications of the test scenarios and extra concerns and observations about the desired and expected facilities, which were not specified or covered in the tests. This feedback is also stored in the ReqVerifier database. **Figure 2** demonstrates the workflow for the stakeholders.

4.3. Managing Testers' Perspective

With respect to the testers' perspective, testers request the report from the Req Verifier tool. Once this request is received, the ReqVerifier tool compares the analysts' expectations of the tests with the tests' answers received from the stakeholders. It calculates a percentage out of the True Or False Specification Test of how well the stakeholder knows the system. This could give an indication of how well stakeholders' requirements have been understood by the analysts. On the other hand, for the second type of the tests, the tool compares the output desired by the stakeholder in the Input Output Scenario with the analysts' expectations entered by the analysts. The tool will compile a report about the comparisons' results of the conducted tests. The tool compiles this result together with the stakeholders' feedback as a report stored in the database of the tool for the testing team. The reports add the additional feedback from the stakeholder for the analysts to see. This comparison will show how close the analysts' expectations were with the test answers of the stakeholders. This would also give an indication of how close the analysis was to what the users expected of the system. In addition, the users can add feedback about the system. **Figure 3** illustrates the workflow for the testers.

5. Conclusions and Future Work

All software requirements must be defined to develop high quality software. The basic technique used is requirement elicitation which is a critical part of the project and different techniques are required to determine the requirements. Fundamental way is to perform analysis in order to identify people, processes and different resources involved in the project. Every project should have been given an amount of analysis. Failure of a project is due to lack of some functional and non-functional requirements.

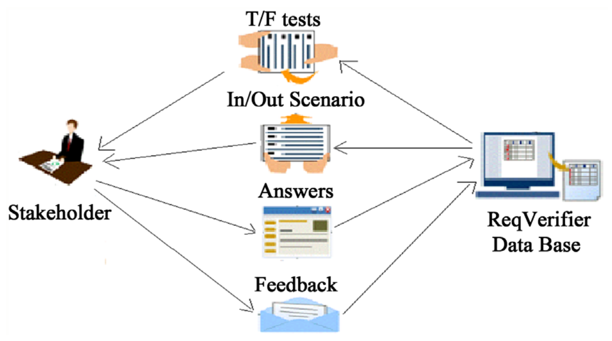

Figure 2. Stakeholder's workflow.

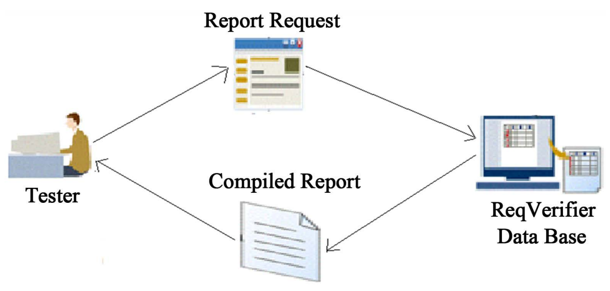

Figure 3. Tester's workflow.

This paper attempted to study the requirement analysis phase and some of its related issues. The current author proposed the ReqVerifier tool as one effort of CASE to help the software engineering community to resolve or at least to reduce the conflicts that might occur between the software developers and the stakeholders. The tool helped in providing information in both sides to the software developers and the stakeholders as a cross check technique. This would eventually lead to the development of a quality software. Analysts, software developers and testers could view reports of the test cases' results and stakeholders' feedback to make conclusions about the specifications they had derived from the stakeholders' requirements and how close they were from the stakeholders' expectations.

As a future work, the author would like to enhance the tool to be able to: classify reports according to results, associate priority with test cases, represent tests as XML files derived from development itself and store them in the tool's database and add specifications interrelations.

References

[1] Saeed, S., Khawaja, F.M. and Mahmood, Z. (2012) A Review of Software Quality Methodologies. In: Alsmadi, I., Ed., *Advanced Automated Software Testing*: *Frameworks for Refined Practice*, Information Science Reference, Hershey, 129-150.

[2] Alsmadi, I. and Saeed, S. (2013) A Software Development Process for Open Source and Open Competition Projects. *International Journal of Business Information Systems*, **12**, 110-122. http://dx.doi.org/10.1504/IJBIS.2013.050662

[3] Beichter, F.W., Herzog, O. and Petzsch, H. (1994) SLAN-4 A Software Specification and Design Language. *IEEE Transaction on Software Engineering*, **10**, 155-162.

[4] Pressman, R. (2001) Software Engineering: A Practitioner's Approach. McGraw-Hill, New York.

[5] Christel, M. and Kang, K. (1992) Issues in Requirements Elicitation. Technical Report, CMU/SEI-92-TR-012, ESC-TR-92-012.

[6] Alfeche, R. (1997) Requirements Engineering: A Good Practice Guide. John Wiley & Sons, Hoboken.

[7] Kotonya, G. and Sommerville, I. (1998) Requirements Engineering: Processes and Techniques. John Wiley & Sons, Chichester.

[8] Wiegers, K.E. (2003) Software Requirements. 2nd Edition, Microsoft Press, Redmond.

[9] Stellman, A. and Greene, J. (2005) Applied Software Project Management. O'REILLY, Sebastopol.

[10] Abran, A. and Moore, J. (2005) Chapter 2: Software Requirements. John Wiley & Sons, Hoboken.

[11] Gorschek, T., Garre, P., Larsson, S. and Wohlin, C. (2007) Industry Evaluation of the Requirements Abstraction Model. *Requirements Engineering Journal*, **12**, 163-190. http://dx.doi.org/10.1007/s00766-007-0047-z

[12] Zagajsek, B. Separovic, K. and Car, Z. (2007) Requirements Management Process Model for Software Development Based on Legacy System Functionalities. 9*th International Conference on Telecommunications* (*ConTEL* 2007), Zagreb, 13-15 June 2007, 115-122.

[13] Jiang, X. (2008) Modeling and Application of Requirements Engineering Process Metamodel. *IEEE International Symposium on Knowledge Acquisition and Modeling Workshop* (*KAM Workshop*, 2008). Wuhan, 21-22 December 2008, 998-1001.

[14] Cheng, J. and Liu, Q. (2008) Using Stakeholder Analysis for Improving State Chart Merging in Software Requirement Management. *The* 9*th International Conference for Young Computer Scientists* (*ICYCS*, 2008), Beijing, 18-21 November 2008, 1217-1222,

[15] Mishra, D., Mishra, A. and Yazici, A. (2008) Successful Requirement Elicitation by Combining Requirement Engineering Techniques. *International Conference on the Applications of Digital Information and Web Technologies* (*ICADIWT*'08), Ostrava, 4-6 August 2008, 258-263.

[16] Berkovich, M., Leimeister, J. and Krcmar, H. (2009) Suitability of Product Development Methods for Hybrid Products as Bundles of Classic Products, Software and Service Elements. *Proceedings of the ASME* 2009 *International Design Engineering Technical Conferences & Computers and Information in Engineering Conference*. *IDETC/CIE* 2009, 30 August-2 September 2009, San Diego, 885-894.

[17] Pandey, D., Suman, U. and Ramani, A. (2010) An Effective Requirement Engineering Process Model for Software Development and Requirements Management. *International Conference on Advances in Recent Technologies in Communication and Computing* (*ARTCom*, 2010), Kottayam, 16-17 October 2010, 287-291.

[18] Pavanasam, V., Subramaniam, C., Srinivasan, T. and Jain, J. (2010) Membrane Computing Model for Software Requirement Engineering. 2*nd International Conference on Computer and Network Technology* (*ICCNT*), Chennai, 23-25 April 2010, 487-491.

[19] Torkar, R., Gorschek, T., Feldt, R., Svahnberg, M., Akbar Raja, U. and Kamran, K. (2012) Requirements Traceability: A Systematic Review and Industry Case Study. *IJSEKE*, **22**, 1-49.

[20] Gorschek, T., Gomes, A., Pettersson, A. and Torkar, R. (2012) Introduction of a Process Maturity Model for Market-Driven Product Management and Requirements Engineering. *Journal of Software: Evolution and Process*, **24**, 83-113.

[21] Khan, M., Khalid, M. and Haq, S. (2013) Review of Requirements Management Issues in Software Development. *International Journal of Modern Education and Computer Science*.

[22] Young, R. (2004) The Requirements Engineering Handbook. Artech house. Inc., Norwood.