Normalization of Requirements Specification Document on Software Project Management

Rachida Hassani*, Younès EL Bouzekri EL Idrissi
bn Tofail University, National School of Applied Sciences, Kenitra, Morocco.

* Corresponding author. Tel.: +212 (0) 662 77 44 75, email: rachida.hassani22@gmail.com
Manuscript submitted August 10, 2017; accepted December 8, 2017.
doi: 10.17706/jsw.13.4.232-241

Abstract: Requirement specification phase is pivotal and central to every successful software development project, because when requirements are not clearly documented, the whole project and team members will suffer, and the project in question typically has little chance to succeed. In this paper, we identify the several reasons why software projects fail, however, poorly requirement quality, missing requirement, the use of use case diagram, tracing requirement and the inadequate verification requirement quality. Then, we have been proposing a standard method that will be helpful for authors to write correctly the requirement specification document and countermeasures the identified problems which can be improved and developed in the next researchers.

Key words: Problems in software project management, management risk, project management, IT project, IT project management, software project management, requirement specification, requirement specification document method.

1. Introduction

According to the Leveraging Business Architecture to Improve Business Requirements Analysis, 2014, 70% of software projects fail due to poor requirements. That is why the most critical phase of the software development life cycle is the requirements phase.

Poor requirements lead to the failure of the project, even if the subsequent was good. The quality of the requirements and the subsequent phase are related, and they affect the overall quality of the project. Writing good software requirements specification (SRS) is an important determinant of software quality [1]. The SRS document defines the capabilities of the provided software [2]. Therefore, if an analyst or a developer does not share the same understanding about requirements, the outcome of the development process will not satisfy the customers' needs [3]. The more progress in the software development life cycle, the more defects will emerge. Consequently, the earlier the detection of requirements defects, the much money and time of rework can be saved [4].

Many developers choose to work with a software requirements specification document as it typically contains the following [5]:

- A complete description of the software's purpose and functionality
- Details as to how the software will perform in terms of speed, response time, availability, portability, maintainability, recovery speed and more
- Use cases of how users will use the software
- The definition of how the application with interact with other hardware and program

- Non-functional requirements (e.g: performance engineering requirements, quality standards, or design constraints)

A requirement specification document “SRS” allows developers to be clear on the goals of the software and on what they should focus on. Furthermore, it allows them to [5]:

- Save time on communication
- Minimize development efforts
- Gives the customer feedback
- Eliminate task duplication
- Facilitate the transfer to new users or to new machines
- Breaks problems down into parts
- Serves as the main document to verify the validation and testing processes
- Referring to past SRS documents helps identify deficiencies and process flows.

So how project manager must write the Software Requirement Specification to success their projects and limit risks related to the requirement phase?

2. IT Project and IT Project Management

As to IT project, there different definitions. Q. Zhang thinks that in IT project mainly refers to the information management systems of computers [6]. Y. Li mentions three features of IT projects as follows:

• Urgency: IT projects have a deadline with a definite beginning or end. When their objectives are met or when they have to be terminated, they are ended.

• Uniqueness: It means that the supplier not only provides customers with their products, but also more importantly, offers various solutions according to their requirements.

• Uncertainly: It is impossible to complete an IT project within stipulated time, based on the stipulated budget or by the stipulated personnel fully [7].

Li. Xu defines an IT project as a series of projects related to information technology such as software, hardware, web systems, information systems, information service etc. Resulting from the requirements for informationization [8].

Regarding IT project management, there are different definitions as well. According to Z. Y. Li, IT project management is the project management based on information technology, a special type of project management, and a new kind of project management that came into being and is being improved continuously with the development of information technology [9].

N. Zhang and P. Wang think that IT project management refers to the management practice which ensures the smooth execution of engineering system development methods, based on the theories of management science, and integrating with the development practice of IT products as well as a series activities in which cost, personnel, progress, quality, risk, file, etc. are analyzed, managed, and controlled so that IT projects can be completed in terms of the budget, scheduled progress and quality [10]. L. Xu puts forwards three features of IT project management: abstractness, timeliness of information communication, and uncertainty [11].

3. Obstacles of Affecting Factors on IT Project Management

The Software Requirements Specification document (SRS) is used to collect the user requirements, which is used as an input for development process, and as a baseline for verifying the correctness of the software product occurring at each step throughout the software development process. It has been found that many organizations cannot deliver software products that satisfy the actual requirements of the customers, due to defects that frequently occur in the SRS, especially the use of ambiguous natural language in the requirements specifications and the inappropriate document structure, which negatively affects the software quality [12].

The following are some of the most affecting factors of the requirement phase in software project management:

- Poor requirement quality:

The quality of many requirements specified is poor, and they do not reflect the correct need of end users, customers, clients, developers, suppliers or business require from it coupled with some needs of the system for efficient functioning. In practice, many of requirements specifications are ambiguous, incomplete, incorrect, inconsistent, not cohesive, using technical terms rather than the user terminology or their activity area, lacking in necessary functionalities, and which contain long of text which is incomprehensible.

In practice, there is no standard way of writing a requirements specification document and improve quality requirements. Authors compose the Software Requirements Specification Document just to validate them with their clients, and respect the life cycle of the project. They spent a minimalist time during the requirement phase, they do not give the necessary importance of this stage, and they accelerate to begin the technical production.

Finally, the end client is obliged to validate the requirement specification document without understanding it, and many times without reading it. Simply because it is too long, it contains an incomprehensible text, unnecessary explanations, and it uses some technical terms.

- Over Emphasis on Simplistic Use Case Modeling:

In practice, writers use only the user case modeling as the only technic to identify and analyze the functional requirement and no functional requirement, rather than creating sequence/swim lane diagrams to capture the normal and exceptional paths through the use cases. They also fail to use text to capture use case path preconditions, triggers, steps, and post conditions. This incorrect practice results ambiguous, incomplete, unfeasible and unverifiable goals.

- Requirements Not Traced:

Requirements tracing is widely recognized and included in many requirements engineering methods and training classes. But in practice, many requirements are still not properly traced, neither in the specification requirement document, nor the test document.

In the era of digital transformation, projects tend to evolve rapidly and changing scope in any time, which complicates the continuation and the updates of the project especially when we talk about the functional requirement.

- Missing Requirements:

The possibility to face missing requirement increase according to the size and the complexity of projects. The authors try to write the whole of functionalities, by using a lot of texts, which increase the chance of forgotten some important functionalities. Because people retain only 20 percent of what they read.

- Inadequate Verification of Requirements Quality:

This problem is not about the verifying whether the as-built system implements its requirements. Rather, it is about verifying sufficiently early in the development process, whether the requirements have sufficient quality to avoid the many negative consequences resulting from poor requirements. Often, requirements are informally verified during small peer reviews and/or as a side effect of major 'dog and pony show' stakeholder reviews. While both reviews are somewhat helpful, they have not proven effective in identifying requirements defects [13].

To summarize, the quality of the SRS is an important factor that affects the development and verification of correctness for the software product to be delivered to the customers. If any defaults appear in the document, it can negatively affect the software development process [12].

4. Solution to Countermeasure the Identified Problems

There is no standard way of writing a requirements specification document. To countermeasures problems related to the requirement phase and to write an adequate requirement specification document, it is necessary to follow a standard guideline that will be helpful for organizations to define their requirements and adapt it to their process.

The objective of the purpose of this guideline will make everyone understand the specifications.
The requirement specification document must include the following outline and structure:

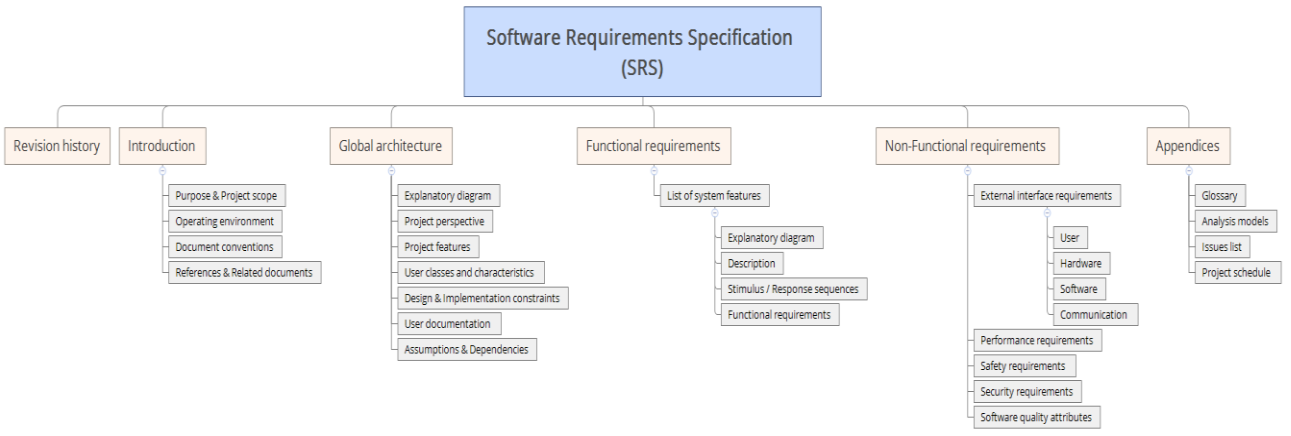

Fig. 1. Structure of the requirement specification document.

4.1. Revision History

The business analyst must write requirements in a fine-grained fashion and give every requirement a unique and stable identifier, and to ensure better traceability, we must automate support, such as using a commercial requirements management tool. Or use the following table:

Table 1. Revision History

Name	Date	Reason for changes	Version	Code color
…	…	……	…	…

For better visibility, you must put a different text background color for distinguishing the changes of the document current version.

4.2. Introduction

Purpose & Project scope

- What we will develop? (short description in two or three sentences)
- Which objectives and goals? (List of objectives & goals)

Operation environment:

- Where will the software be developed?
- Which hardware platform?
- Which operating system and versions?
- There are other software components or applications with which it must peacefully coexist?

Document conventions:

- Which standards or typographical conventions that must be followed when writing the SRS? (Font, highlighting, etc.).

References & Related documents

- There are any other documents or web addresses to which the SRS refers? (Example: user interface style guides, contracts, use case documents, etc.).
  *each reference must include the: title, author, version number, date, source or location.

4.3. Global Architecture

Explanatory diagram:

- Should identify all the functionalities that will include the future system in the form of a general diagram that must define in a simply way the global process. Adding to this, a table that identifies and summarize all the roles and their rights.
- The general diagram will define in a simple way the global process, the different actions and interactions of the system, including the different profiles and their workflow.
- The table and the general diagram will help the business analyst to avoid forgetfulness and facilitate understanding of the overall process to all stakeholders.

Project perspective:

- What is the context and origin of the software being specified in the SRS?
- To answer this question, the business analyst must put a simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces.

Project features:

- What are the major features the software contains? And what are the significant functions that it performs or let the user perform?
- To answer this question, the business analyst must only put a high level summary and organize the function to make them understandable to all readers of the SRS.

Also, he must put a picture of the major groups of related requirements and how they relate.

User classes & Characteristics

- What are the various user classes that will be used in this software?
- What is the pertinent characteristics of each user class?

Design & Implementation constraints

- What are the issues and items that will hurt the smooth development of the software development
- The identified issues must be defined as follows

Table 2. Design and Implementation Constraints

Issue or item	Short description
…	…
…	…

User documentation

- What are the documents to deliver with the software?
- For what format or standard will this be delivered?

Assumptions & Dependencies

- What are the assumed factors that may affect the requirements defined in the SRS? (As commercial components which can be used, operating system, etc.)
- What are the project dependencies on external factors?

4.4. Functional Requirements

List of system features: for every feature we must respect the following points:

- Feature name: we must put the feature name (in some words) and not say "system feature"
- Explanatory diagram & graphic interface: put an explanatory diagram to simplify the understood of the feature by including all the interactions, functionalities, and users. Because images are very comprehensible to texts.
- Description: put a short description of the feature (One or two sentences), its priority, penalty (if necessary), cost, and risk.
- Stimulus & Response sequences: list of dialog elements associated with use cases.
- Functional requirements:
  - Describe the software capabilities that must be present for the user to perform the services provided by the feature or to run the use case.
  - Define how the functionality should react in the errors or invalid entries cases.
  - Each requirement must be uniquely identified with any meaningful sequence number or tag.

Table 3. Functional Requirements

Requirement ID	Description
REQ 1	
REQ 2	

4.5. Non-functional Requirements

- External interface requirements: define all the external interface requirements type as the table follows:

Table 4. Non-functional Requirements

Interface	Description
User interfaces	Describing the logical characteristics of each interface between the software project and the users (sample screen images, project family style guides, screen layout constraints, error messages, etc.).
Hardware interfaces	Describing the logical and physical characteristics of each interface between the software project and the hardware system components (supported device types, data nature, control interactions between software and hardware, communication protocol, etc.).
Software interfaces	Describing the connection between the project and other specific software components (Database, operating system, tools, libraries, integrated commercial component, etc.).
	Describing the needed services and communications nature.
Communications interfaces	Describing the associated requirements of communication functions which are required by the project (email, web browser, network server communication protocol, etc.).

- Performance requirements: which will be helpful for developers to understand the intent and make suitable design choices.

Table 5. Performance Requirements

Performance	Explanation
Performance 1	
Performance 2	

Each performance requirements must be for an individual functional requirements or features.

- Safety requirements:
  - Identify any losses or damages that may occur

    - Define the actions to be taken and the actions to be prevented
    - Refer to external regulations that define the security issues which affect the project design or use
    - Define the security certifications that must be certified
- Security requirements:
    - Define the security requirements or the privacy issues which surround the project use or the protection of the used or the created data by the project
    - Specify the authentication requirements of the identified users
    - Identify the regulations which contain the security issues that affect the project
    - Define the security certifications that must be certified
- Software quality attributes: define the additional quality characteristics (as adaptability, availability, correctness, flexibility, etc.).

4.6. Appendices

- Glossary: definition of all the terms necessary to understanding the SRS, including acronyms and abbreviations. Because it should not make assumptions about the reader experience or expertise.
- Analysis models: this section is not required for all types of project. It must include the analysis models (data flow diagrams, class diagrams, state-transition diagrams, entity-relationship diagrams).
- Issues list: list of the open requirements issues that remain to be resolved.

To better explain the proposed solution, the business analysts must avoid the max possible the long descriptions, and replace them by illustrations, graphs, visuals and simplified and explicit diagrams. Simply because visuals can save 1000 words, and they are very easy and helpful to communicate any ideas better. Visual content reaches an individual’s brain in a faster and more understandable way than textual information. Or, more accurately, a person’s brain is hardwired to recognize and make sense of visual information more efficiently, which is useful considering that 90 percent of all information that comes to the brain is visual [14].

It is believed that human communication has existed for over 30,000 years, but text-based communication has only been around for 3,700 years. So for the vast majority of human history, people have had to communicate without the written word, meaning our brains had plenty of time to brush up on visual messages. Our brains became hard-wired to process visual data, including colors, which have been coded in our brains to represent certain messages or evoke specific emotions. Apparently, exposure to the color red increases pulse and breathing rates [15].

Modern applications are already successfully experimenting with this information, with many mobile apps focusing on images, from Instagram and Snapchat to Pinterest and Vine. And it doesn’t seem to be a passing trend: Engagement per follower is 58 times higher on Instagram than on Facebook [15].

Writers must use visuals, graphics, tables and diagrams, to explain the general system, and every module or subsystem constituting the product to implement. In this way, the end client, customers, developers or any person which will read the requirement specification document will not find itself re-reading the same sentence eight times or staring at a page and realizing it has no idea what it was read.
This method will be useful to avoid missing some important requirement, because global diagrams will help to identify all the functionalities that can be forgotten and simplify the quality and the quality verification of requirements.

Writers must avoid including things that may not need to be documented. Requirement Specification documents may get a bit long, so avoid packing in unnecessary information. Which will help to make easy reading and understanding the document and improve it quality. Because anything that is unclear or miscommunicated can lead to not-so-great consequences.

The writers must keep an online version of the requirement specification document to keep updating and tracing requirements change and improvement. This method will ensure management understands the negative consequences of not tracing requirements, and obtain support for proper tracing, including providing adequate resources to trace the requirements. Ensure that tracing occurs both early in the project development cycle as well as later during design, development, and maintenance. Finally, ensure that the evaluation of requirements tracing is a documented part of the requirements verification method [13].

To summarize all the above, the following table will summarize how the proposed SRS document structure will answer the problems already identified:

Table 6. Summary of the Proposed Solution

Problem	Solution
Poor requirement quality	The establishment of a general explanatory diagram will avoid the oversights and produce a complete SRS document.
	The use of illustration, explanatory diagrams, and images will simplify the SRS document understanding, and avoid misinterpretation of its contents as well as ambiguities.
	The establishment of a glossary, especially for technical terms, will allow the document to be understandable by any reader regardless of his profile and skills.
	The fact of putting very little text and too much explanatory diagrams will reduce the document size, something which will simplify the reading and the comprehension of all the described functionalities, and later the simplification of the validation process of the SRS document by the customer.
	Normalizing the SRS document will help to avoid oversights, make any document easy to read and easy to produce, and especially without ambiguity and incomprehensible texts.
Over emphasis on simplistic use case modeling	The proposed SRS document structure utilizes all aspects of use case modeling to ensure that all credible paths through the use case are identified and analyzed.
	The use of the explanatory diagrams will be as an identification and analysis technique, rather than as a requirements specification technique.
	Also, the proposed structure and template will make it possible to clearly distinguish between the functional requirements and the non-functional requirements, and will allow to frame the utilization of the use cases, the sequence diagram, the class diagram, etc.
Requirement not traced	The setting up of the summarizing table of all the SRS document changes will make it possible to countermeasure all the problems related to the SRS document evolution.
Missing requirements	The use of a general explanatory diagram will make it possible to avoid omissions.
	Reducing the SRS document size will allow the business analyst to check and recheck the SRS document, subsequently increase its quality and especially to add and complete the missing information and requirements.
	The production of a complete and the simplification of the SRS document reading and understanding will allow the customer to add, correct and complete his requirements in an efficient way.
Inadequate verification of requirements quality	Producing a complete document that is easy to read and understand will help to identify the requirements defects during the requirements engineering process which will positively impact all the subsequent activities.
	The identification of the requirements defects during the requirements engineering process will prevent the discovery of these defects after production, which will be more expensive in terms of budget and schedule.

5. Conclusion

The quality of the SRS is an important factor that affects the development and verification of correctness for the software product to be delivered to the customers. If any defaults appear in the document, it can negatively affect the software development process. Therefore, if the quality level and defects appearing in the SRS can be determined, it leads to those defects being corrected to make the document more complete during the earliest steps of the software development process. This research presents a standard guideline and structure of writing a requirements specification document that will be helpful for organizations to define their requirements and adapt it to their process. In each part of this guideline, we define what we must input in an easy way which allow a fluent understanding by any reader of the document.

This guide is based on limiting the writing of specifications by a human language that can be interpreted differently from one reader to another, and the development of illustrations, graphs, visuals and simplified and explicit diagrams. Simply because visuals can save 1000 words, and they are very easy and helpful to communicate any ideas better. Visual content reaches an individual's brain in a faster and more understandable way than textual information.

The future research should develop this guideline, while putting a standardized, complete and tested example on several projects.

References

[1] Aurum, A., & Wohlin, C. (2005). *Engineering and Managing Software Requirements Springer*. USA: Verlag New York Inc.

[2] Wilson, W. M. (1999). Writing effective natural language requirements specifications. *Crosstalk J. Def. Softw. Eng.,* 16-19.

[3] (2009). Effective requirements definition and management: improves systems and communication (White paper), (Ed.), *8310 North Capital of Texas Highway, Corporate Office: Whitepaper of Borland Software Corporation - The open ALM Company.*

[4] Stand. A. (2014). Title of paper with only first word capitalized. *Detecting Defects in Software Requirements Specification Alexandria Engineering Journal,* 513-527.

[5] Nedesk. retrieved from http://www.onedesk.com/writing-a-software-requirements-specification-document/

[6] Zhang, Q. (2008). Methods and thinking of IT project management. *Electric Power IT,* 20-24.

[7] Li, Y. (2007). Existing problems and their countermeasures in IT project management. *Information Research,* 115-117.

[8] Xu, L., & Wu, W. Y. (2010). Requirement management of IT projects. *China Management Informationization,* 80-82.

[9] Li, Z. Y. (2008). Solving problems in IT project management by psychology. *Science and Technology Innovation Herald*, 162-163.

[10] Zhang, N., & Wang, P. (2010). How to conduct IT project management. *Sysmanagement,* 42-44.

[11] Xu, L., & Wu, W. Y. (2010). Requirement management of IT projects. *China Management Informationization,* 80-82..

[12] *Proceedings of* the *2015 2nd International Conference on Trustworthy Systems and Their Applications.*

[13] (2007). Common requirements problems, their negative consequences, and the industry best practices to help solve them. *ETH Zurich, Chair of Software Engineering ©JOT.*

[14] Retrieved from https://www.eyeqinsights.com/power-visual-content-images-vs-text/

[15] The power of images. Retrieved form https://courses.thelifestylemarketer.co/lesson/the-power-of-images/

Rachida Hassani was born in 1991 in Rabat (Morocco), she has a degree in software engineering from ENSA Kenitra (Morocco), she is currently a Ph.D student in IT project management at the same school, and in parallel, she is working as an IT project manager in collaboration with several countries (Morocco, France, USA, etc.), she deals with IT project management issues from different angles, while integrating the artificial intelligence through innovative solutions.

Younès EL Bouzekri EL Idrissi obtained his international doctorate in computer science at ENSIAS Rabat (Morocco), and since 2013, he works as a professor of higher education with computer skills at ENSA Kenitra (Morocco), his current research are on various subjects as IT project management, smart cities and big data.