Requirements Documents

Gordan Konevski

Electronic Engineering 2021

Abstract

This paper is concerned with the uses and categorizations of requirements documents. However, the further the "uses" of these documents are explored, the more it becomes clear that each category is heavily dependant on its idiosyncrasies and aims. Furthermore, the paper explores the concept of using "System Models" as an alternative, or an upgrade, to typical requirements documents, in order to establish some kind of a reference point, which would allow to concretize the qualities of requirements documents.

Chapter 1

Introduction

Before any product can begin its production, its requirements must be first specified. This is typically done through Natural Language (NL), as it is referred to in the world of Engineering. NL is essentially what it suggests that it is and that is normal human language. Typically, the preferred choice for requirements documents is English, or the native language, but as the world shifts further towards globalism, English is quickly becoming the norm even amongst those who could comfortably hold such discussions in their native language. According to a paper written by Alessio Ferrari, Giorgio Oronzo Spagnolo and Stefania Gnesi , several authors have applied natural language processing (NLP, which is used to connect human language with machine language) techniques in requirements engineering (RE) to address multiple tasks, including: model synthesis, classification of requirements into functional/non-functional categories, classification of online product reviews, traceability, ambiguity detection, structure assessment, detection of equivalent requirements, completeness evaluation and information extraction

Chapter 2

Context

So, given all that was stated above, what are requirements documents anyway?

Here is an example: a document containing all the requirements to a certain product, inclusive of all kinds of intrinsic details about the product from start to finish, is what is referred to as a product requirements document (PRD). A PRD is just one example of a requirements document, and we can categorize nine in total as such:

1. Business Requirements Document (BRD)

   Also referred to as a Business Needs Specification, it is the very first stage in a product life cycle. It details the problems that a product/service/system is trying to solve by logically listing high-level business requirements in relation to customers' needs. As well as non-negotiables, it also details features the project should provide, which can be interpreted as goals for the development team.

   It often includes:

   - A preliminary outline

   - List of rank-ordered objectives

   - A needs (problem/solution) statement, detailing why the project is needed and how it will meet those needs

   - Financial statements, demonstrating how the project will be funded and its effect on the company's balance sheet, accounting for risk

   - Functional requirements and features

   - A list of possible outsourcing options, as well a proper overview of what is to be outsourced and what would be more practical to develop without outsourcing.

- Schedules, timelines and deadlines

- A cost-benefit analysis, both in the perspective of the product developer and the customer

A BRD is generally prepared by project managers, business analyst and approved by upper management.

2. Functional Requirements Document (FRD)

An FRD is used to define, in logical terms, how a system or project will accomplish the requirements laid out in the BRD. It outlines the functionality of the system in detail by capturing the intended behaviour of the system, expressed as services, tasks or functions that the developers have agreed to provide. Rather than define the 'inner-workings' and specifications, an FRD focuses on what users might observe when interacting with the system. It is described in a washed-down, step-by-step basis, guiding the reader as to how, for example, the interface would function and react. Depending on the complexity, FRDs can vary in length from 10 pages to several hundred. An FRD is generally written by business analysts (in order to account for the 'competition factor') and systems analysts, in parallel with the engineers and designers (in order to account for the "development capabilities" factor). The biggest obstacles generally arise, as expected, between engineering departments and design departments.

3. Market Requirements Document (MRD)

Sometimes referred to as a Marketing Requirements Document, an MRD focuses on the target market's needs. It typically explains: What the product is, who the target customers are, what products are in competition with it and why customers are likely to want this product. This falls in line with BRDs and are used as their primary source of information regarding competition.
An MRD typically includes:

- A definition of the target market, with great focus put on the customer's perspective

- A comprehensive list of market requirements, although this heavily depends on the partnerships involved and whether or not the product is developed in a B2C (business to customer) environment or B2B (business to business)

- A prioritized list of requirements from the customers's point of view

- A timeframe for the product's launch, worked in parallel with PRD

An MRD is generally prepared by the marketing managers in parallel with the product managers.

4. Product Requirements Document (PRD)

   A PRD is used to elaborate on and delineate the use of a product, and is used by product developers to plan out development strategies. It is generally best to avoid anticipating or defining how the product will perform in the market, in order to later allow interface designers and engineers to use their expertise to provide the optimal solution to the requirements, rather than having them be concerned with marketability. The market cannot be accurately predicted with enough consistency to allow for this 'anticipation' factor, thus PRDs are limited to the description of the product itself in accordance with its *utility*, rather than its *marketability*. A well thought out, well-defined PRD, which has already settled on its costs and has allocated enough resources to account for risk, would greatly ease the jobs of both the designers and engineers and would have a greater impact on the success of the execution of such requirement, i.e. successful pre-planning allows for a more successful execution.

   "PRDs are most frequently written for software products, but can be used for any type of product and also for services. Typically, a PRD is created from a user's point-of-view by a user/client or a company's marketing department (in the latter case it may also be called Marketing Requirements Document (MRD)). The requirements are then analyzed by a (potential) maker/supplier from a more technical point of view, broken down and detailed in a Functional Specification (sometimes also called Technical Requirements Document)." [2]

5. User Interface Requirements Document (UIRD)

   A UIRD provides a comprehensible delineation of the UI and its 'artistic' direction. This idea of an artistic direction may fall completely in or out of line, depending the overall management of the company, but also with the idiosyncrasies of the product in question. It often defines aims like: How the product is experienced by the user, user navigation, colour themes and hints, tips and suggestions.
   A UIRD more often than not includes mockup screenshots and wireframes to give readers an idea of what the finished system will look like. It's written by the UIX design team.

6. Technical Requirements Document (TRD)

   A TRD is quite in line with PDRs, except it deals specifically with the technical details of the development process. This would include: programming languages used, hardware limitations, team-oriented agile tools, PLM tools (to the degree to which such tools exist), etc.

Key items of a TRD would include:

- An executive summary of the project and its background.

- Assumptions, risks, and factors that may affect the project

- Functional and non-functional requirements, in context to the specific choice of software and hardware tools

- References, supporting documents

TRDs, naturally, are written by the engineers themselves and each choice is made within the limits of the project's budget.

7. Quality Requirements Document (QRD)

    The quality requirements document outlines the expectations of the customer for the quality of the final product. It consists of various criteria, factors and metrics that must be satisfied. Key factors of these documents are: revolve around reliability, consistency, availability, usability, maintainability and overall customer experience. Each comapny has their own scope of what exactly is detailed in QRDs, but they might include factors like 'product testing' and 'demo distribution', depending on the product, naturally. This document can be written by the project managers, or business analysts.

8. Software Requirements Document or Software Requirements Specification (SRS)

    An SRS outlines the features and the intended behaviour of a system. It describes the business's understanding of the end user's needs while laying out functional and nonfunctional requirements. An SRS is related to the FRD and PRD but written with a specific IT project in mind. An interesting study developed by Colin Werner, Ze Shi Li and Neil Ernst explores this interensting idea of a SRS sentiment and they explore the idea of potentially analyzing this sentiment. They describe the idea as follows: [3]

    > Sentiment analysis is a growing field of software engineering research and there is a keen interest in applying sentiment analysis to the requirements engineering domain. Sentiment analysis has been used to analyze the sentiment of support tickets, in conjunction with machine learning techniques to predict support ticket escalations, assess user's sentiment towards application features through application reviews, and detect a developer's sentiment, in particular the direction of anger in collaborative software development.

Following that, they elaborate further: [**3**]

> The implications that can be drawn based on our data are not yet well defined. Part of the incentive for this paper is to publish the sentiment data to enable the community and readers to provide meaningful contributions on how this data can be used. In addition, based on the sample twenty-two SRS documents, we can definitively infer that an SRS has an overly neutral sentiment; however, what this means to the research community is something that needs to be further explored. Some potential research questions are posed to the audience for future consideration:
>
> - Given the overly neutral sentiment of an SRS document, can sentiment analysis be used as an indicator to evaluate the quality of an SRS document?
> - Can sentiment analysis of an SRS document be used as an input feature to classify whether a particular sentence is part of an SRS document?
> - Can sentiment analysis be used to as an input feature to classify whether a sentence is part of a SRS domain type?
> - Does the sentiment of a SRS document changed based on the specific theme of requirements (e.g. security or privacy are perhaps overly negative)?
> - How does the sentiment of an SRS compare to the sentiment of a user story?

This study rightfully opens up an important debate about a company's "morale" during the development process of their product.

9. Customer Requirements Document (CRD)

CRDs are essentially tailor-made PRDs for specific customers, moslty used in B2B cases. There are no structural differences between the two, except that CRDs are particularly focused on the idiosyncrasies of the project in question, so the biggest difference is essentially the priorities in question.

Chapter 3

System Models Introduction

Leonid Kof aimed to create a model with which requirements documents could be translated into system models, in order to improve their usability. He explains as follows: [**4**]

> At the beginning of every software project, some kind of requirements document is usually written. The majority of these documents are written in natural language. This results in the fact that the requirements documents are imprecise, incomplete, and inconsistent. It is one of the goals of requirements analysis, to find and to correct the deficiencies of requirements documents. A practical way to detect errors in requirements documents is to convert informal specifications to system models. In this case, errors in documents would lead to inconsistencies or omissions in models, and, due to more formal nature of models, inconsistencies and omissions are easier to detect in models than in textual documents.
>
> ..
>
> The tool was evaluated in two stages: firstly, the tool author used the tool in order to translate four specifications to system models. The Steam Boiler specification, Autopilot, Bay Area Rapid Transit (BART), and Instrument Cluster were used for this purpose. For all four specifications, the tool kept up with the expectations of the tool author.
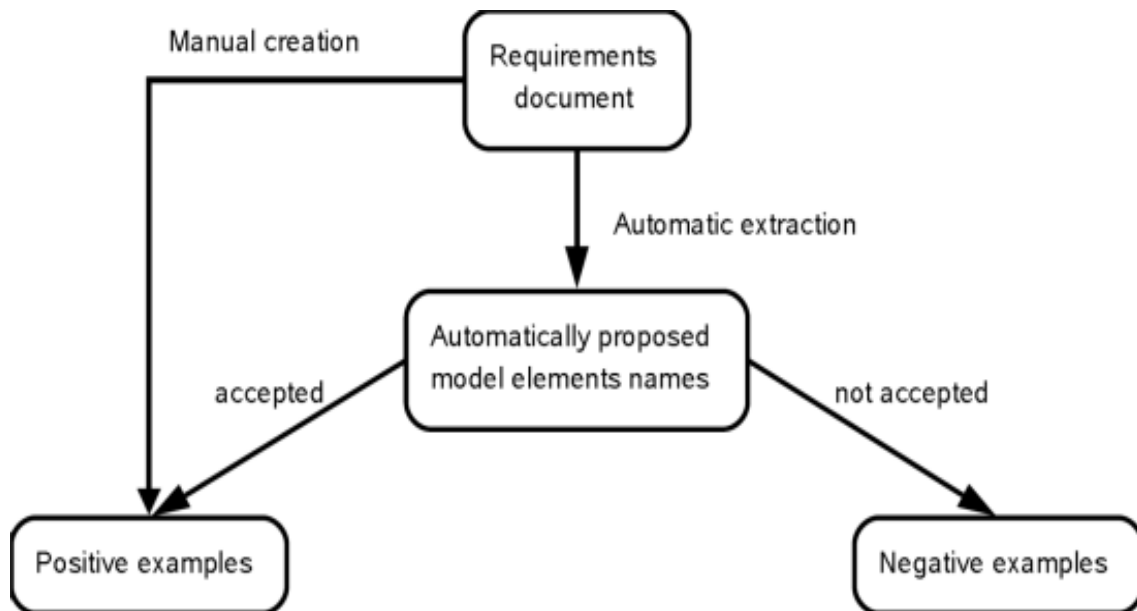
Figure 3.1: Example Diagram of Kof's tool [4]

Chapter 4

Conclusions

The main point behind that can be taken out of this research paper is that the requirements documents has some definitive flaws. Most of these flaws stem from its overreaching nature. Requirements documents, regardless of their category, tend to easily become too large and heavy. It is usually necessary to provide simple handouts on top of the requirements documents in order to make them more accessible. This makes the entire process of executing on requirements documents quite slow and cumbersome, and often convoluted due to its interdisciplinary nature. Translating these documents into specific models, which are chosen in accordance to the project, would greatly increase their accessibility and would make communicating with different departments and sectors far easier. However, requirements are a must, that is to say: no project could ever be developed without them. Even *if* they are eventually translated into different systems models, they are a necessary prerequisite to any project taken seriously.

Bibliography

[1] PURE: A Dataset of Public Requirements Documents - Alessio Ferrari; Giorgio Oronzo Spagnolo; Stefania Gnesi

[2] Wikipedia source: https://en.wikipedia.org/wiki/Product_requirements_document

[3] What Can the Sentiment of a Software Requirements Specification Document Tell Us? - Colin Werner; Ze Shi Li; Neil Ernst

[4] From Requirements Documents to System Models: A Tool for Interactive Semi-Automatic Translation - Leonid Kof