University of Pretoria

COS 301 - Software Engineering

Team Fox

Software Requirements Specification and Technology Neutral Process Design

Author(s):
Gian Paolo Buffo

Student number(s):
14446619

February 15, 2016

1 Introduction

The requirements specification should ultimately contain sufficient information such that the system could be largely developed by a third party without further input. To this end the requirements must be precise and testable. The requirements need not be fully specified up-front. One might start with the vision, scope and architectural requirements, perform an upfront software architecture engineering phase and then iteratively elicit the detailed requirements for a use case, build, test and deploy the use case before adding the detailed requirements for the next use case. Such an approach follows solid engineering phase for the core software infrastructure/architecture with an agile software development approach within which the application functionality is developed iteratively.

2 Vision

A short discussion of the project vision, i.e. what the client is trying to achieve with the project and the typical usage scenarios for the outputs of the project.

3 Background

A general discussion of what lead to the project including potentially

- business/research opportunities,
- opportunities to simplify/improve some aspect of life/work or community,
- problems your client is currently facing,
- . . .

4 Architecture Requirements

The software architecture requirements include the access and integration requirements, quality requirements and architectural constraints.

4.1 Access Channel Requirements

Specify the different access channels through which the system's services are to be accessed by humans and by other systems (e.g. Mobile/Android application clients, Restful web services clients, Browser clients, . . .).

4.2 Quality Requirements

Specify and quantify each of the quality requirements which are relevant to the system. Examples of quality requirements include performance, reliability, scalability, security, flexibility, maintainability, auditability/monitorability, integrability, cost, usability. Each of these quality requirements need to be either quantified or at least be specified in a testable way.

4.3 Integration Requirements

This section specifies any integration requirements for any external systems. This may include

- the integration channel to be used,
- the protocols to be used,
- API specifications in the form of UML interfaces and/or technology-specific API specifications (e.g. WSDLs, CORBA IDLs, . . .), and
- any quality requirements for the integration itself (performance, scalability, reliability, security, auditability, . . .).

4.4 Architecture Constraints

This specifies any constraints the client may specify on the system architecture include

- technologies which MUST be used,
- architectural patterns/frameworks which must be used (e.g. layering, Services Oriented Architectures, . . .)
- . . .

5 Functional requirements and application design

This section discusses the application functionality required by users (and other stakeholders).
Additional information for this section is provided in the spec uploaded to the CS website.

5.1 Use case prioritization

5.2 Use case/Services contracts

5.3 Required functionality

5.4 Process specifications

5.5 Domain Model

6 Open Issues

Discuss in this section

- any aspects of the requirements which still need to be specified,
- around which clarification is still required, as well as
- any discovered inconsistencies in the requirements.