Requiremnts

Alex Sverdlov
alex@theparticle.com

1 Introduction

Every projects starts (or should start) with some business-need—and documentation of that need. The documentation of what-is-needed is essentially the requirements of what needs to be built. It is the first step in solving the problem using technology.

Requirements gathering often starts with a series of interviews, often of business users or other relevant parties. In some cases it may start with examination of an existing system, and its shortcomings. During requirements gathering, it is important to keep the focus on the *what*, as opposed to the *how*—the *how* question is for a later phase. The result of such interviews is some form of codification of requirement items, often called the 'user requirements specification'. The level of the documentation should be appropriate for all stake-holders (the designers and developers should understand the business concepts, and the business should understand the technical details that might be part of the requirements).

2 Audience

What is the audience of the document you are creating? Make the requirements appropriate to the audience. Sometimes business folks need business requirements, and techy folks need techy requirements. Database folks may need another set of requirements, and UI folks may have another set, etc.

3 Story

It is often desirable write a story of the project. The story is an overview of what the project does, how it does it, what users will interact with it, and how they will interact with it. Use clear and concise language, with commonly understood terminology. Avoid statements that can be misinterpreted or understood in multiple ways.

It doesn't have to be complete, just a gist of an idea: something you can explain during an elevator conversation with similarly minded techies.

4 Use Cases

From above story we can get use-cases. Essentially usage scenarios that the system must accommodate. Use cases often have an "actor" (the user who is performing the use-case), and some sort of action (what they are using the system for). The actors do not have to be real people—they may be job roles (e.g. clerk, customer, etc.) or other systems.

The actions should have a sequence of steps that are part of the use-case. For example, to a clerk can type in a customer's order number, and find the order.

Each use case is often its own document—and a whole set of use-cases is what we expect the system to be able to do.

Each use-case should be as independent of other use-cases as possible, so implementation can proceed on a use-case-by-use-case basis: we can judge progress by seeing how many use-cases are complete.

5 Entities/Actions

Whenever we see a noun in the project story it often indicates some sort of entity. Verbs in the story are actions. Often we can understand/model requirements in terms of entities performing actions. This is similar to use-cases—with the added benefit that entities can often be mapped to database or programming language objects, with actions between entities being API calls or linkage tables.