Section 3: Requirements and Specification Document

Instructions

We will use a simple template for requirements and specification. Inevitably, in preparing this document you will discuss design and planning, but limit this document to requirements, a description of how the system should interact with the outs ide world.

The following pages are a requirements and specification document template. Please follow these directions for filling out this template carefully. Items in italics are information that you are to fill in, beginning with the project name, date, and version number.

Your first version of this document is version 1.0. After that, minor changes increment the minor version number (e.g., 1.1, 1.2, . . .), and major changes increment the major version number and set the minor number to zero (e.g., 2.0, 3.0, . . .). We will follow this convention with other documents as well.

If any section is not applicable, put N/A in the body of the section (do not delete the section). Also, if the information has already been given in another document, refer to that document.

Template

Project Name
Requirements and Specification Document
Date
Version *major/minor*

1. Project Abstract

A one paragraph summary of what the software will do.

2. Document Revision History

Rev. 1.0 *date* - initial version

3. Customer

A brief description of the customer for this software, both in general (the population who might eventually use such a system) and specifically for this document (the customer(s) who informed this document). Every project is expected to have a dummy customer. Requirements should not be derived simply from discussion among team members.

4. Competitive Landscape

Briefly identify the competitors in this market, this may need to be on a country and industry basis as the landscape varies dramatically. Any functional details about the competitive solutions should be provided. For example, the known strengths or differentiator points in the competitive solutions, and also their weakness. Identifying the strengths helps ensure that we

design a competitive solution; the weaknesses allow us to consider these as areas for potential differentiation. Remember there are all sorts of sources for competitive information – Marketing, Development, Customers, the internet, field staff, and those that we have hired from the competition (don't be afraid to be creative here. Note that this section may or may not be something that we want customers to see, depending on how 'honest' we wish to appear as a company).

What product features can create competitive differentiators? Competitive barriers? Can a patent position be obtained in this area? What is the current patent status in this arena?

5. System Requirements

A detailed specification of the system. Every possible execution should be covered, though not every aspect need be covered in extraordinary depth. Also, remember to consider non-functional requirements such as timing constraints, security and privacy issues, memory requirements, performance/speed requirements, data capacity requirements, etc. UML, or other diagrams, such as finite automata, or other appropriate specification formalisms, are encouraged over natural language.

Every major scenario should be represented by a use case, and every use case should say something not already illustrated by the other use cases. Ask the customer what are the most important use cases to implement by the deadline. You can have a total ordering, or mark use cases with "must have" "useful", "optional". For each use case, you may list one or more concrete acceptance tests (concrete Scenarios that the customer will try to see if the use case is implemented).

Be sure to describe any customer user interface requirements including graphical user interface requirements as well as data exchange format requirements. This also should include necessary reporting and other forms of human readable input and output. This should focus on how the feature or product and user interact to create the desired workflow. Describing your intended interface as "easy" or "intuitive" will get you nowhere unless it is accompanied by details.

Also, make a list of all the external entities, other than users, on which your system will depend. For example, if you require a web server, a database, or a particular operating system, be sure to include it as a requirement.

6. Checklist

The following checklist is provided to help you think about whether the document is complete and correct. You may want to add your own additional questions to the list.

- Have all the viewpoints of all the stakeholders been taken into account?
- Do the requirements completely specify what the function of the program should be?
- Are all non-functional requirements (e.g. speed, memory, capacity) specified?
- If there are requirements imposed by the environment in which the program will be used, are those requirements specified?
- Do the requirements avoid specifying a solution or a design?
- Are separate requirements listed separately and not lumped together?
- Are the requirements clear and precise, not ambiguous?

- Are the requirements consistent, no contradictions?
- Are the requirements accompanied by a rationale or justification?
- Are the requirements given in a consistent format?
- Are the requirements properly prioritized?
- How is the notation in the document? Are graphical and mathematical notation used appropriately? Is the technical jargon kept to a minimum?
- Does the requirements document make good use of scenarios and use cases?
- Are the requirements realistic?

Are the requirements verifiable?

Acknowledgement: Cornell University, New York University.