Chapter 4: Requirements Specification

Requirements Engineering
Objectives

In this chapter, you will learn about:

- How to write system requirements
  - Technical requirements from the perspective of developers
- Enumerated Requirements
- IEEE Standard for Software Requirements Specification
Requirements Specification

Requirements Specification covers same ground as Requirements definition but from the perspective of developers

- Requirements Specification defines
- Our goal is to use mainly our Functional Requirements developed in project 3 to describe the following system requirements in technical terms:
  1. Systems Interfaces
  2. Systems Required Functionality in terms of the interfaces’ input and outputs
  3. Performance Requirements
- Your deliverable is a technical, unambiguous and detail description of what the developers are supposed to produce (code) for an acceptable system
Systems Interface

1. **Systems Interface** refers to descriptions of interactions between your system and external entities

- **User interface**
  - Specifies requirements for control of the user interface, Screens and Displays
- **Other software systems**
- **Hardware Interface**
- **Operations Support (Communications) Interfaces**
  - Specifies requirements for communicating with External Systems
How about the “Internal Interfaces”? 

- These are typically derived from your requirements specification by the software architect to describe protocols for inter-process communications and access to any internal dB
Requirements Specification

Systems Interface

Detail Description of I/O

Using the Operational Scenarios captured in your USE Cases; and the Message Sequence Charts, “document” at least the following for each external interface:

- All inputs and Outputs in detail
- Source of inputs
- Destination of outputs
- Value Ranges
- Data formats of input
- Data formats of output
- Protocols governing the order (sequence) in which certain inputs and outputs must be exchanged
- Window format and layout
- Timing constraints

We will discuss specs for documenting technical requirements shortly
Requirements Specification

Functionality (Features)

Detail Description of System Functionality
Using the Operational Scenarios captured in your USE Cases; and the Message Sequence Charts, “document” the system features in terms of interfaces’ inputs and output

- State each feature Name
  - Short description of the feature
  - Prioritize feature as High, Medium, or Low.
  - Employ the Use Case diagrams and MSC to identify user actions and systems responses that will trigger the system behavior for this feature
Requirements Specification

Performance ...

- **Performance, Resource and Reliability:**

  - Input/Output Load Factors - factors that determine the measure of load which the feature must support
  - Database Load Factors - factors that determine the amount of data that must be retained in each data store
  - Response Time Factors - factors which measure the delay between the initiation of an operation and its conclusion
  - Resource Utilization Factors - maximum resources available to the feature
  - Reliability - how often a feature may fail in a specified period of time and still be acceptable
  - Availability - the amount of time a feature must be usable out of some period of time. Availability is a function of the amount of time to recover and restart because a failure is detected or operationally required downtime is needed.
Requirements Specification

Structuring Requirements Document

Requirements documentation can be very large

Solution approach:

- Organize requirements into well defined categories
  - Divide and Conquer

Benefits of structuring requirements include:

- Minimize the number of requirements
- Understand large amounts of information
- Find sets of requirements relating to particular topics
- Detect omissions and duplications
- Facilitates consistency across requirements
- Eliminate conflict between requirements
- Manage iteration (e.g., delayed requirements)
- Reject poor requirements
- Evaluate requirements
- Reuse requirements across projects
Structuring Requirements Document

- Proprietary requirements templates
- Boilerplate requirements templates
- IEEE 803-1998 SRS Standards
IEEE 803-1998 Recommendations for organizing software requirements specification by:

- Mode of Operation
- User class
- Feature
- Stimulus
- Functional hierarchy
IEEE 803-1988 Standard for SRS
Mode of Operation Template

1. Introduction to the Document
   1.1 Purpose of the Product
   1.2 Scope of the Product
   1.3 Definitions, Acronyms and Abbreviations
   1.4 References
   1.5 Outline of the rest of the SRS

2. General Description of Product
   2.1 Context of Product
   2.2 Product Functions
   2.3 User Characteristics
   2.4 Constraints
   2.5 Assumptions and Dependencies
3. Specific (“System”) Requirements
   3.1 External Interface Requirements
      3.1.1 User Interfaces
      3.1.2 Hardware Interfaces
      3.1.3 Software Interfaces
      3.1.4 Communications Interfaces

4. Open Issues

5. Acknowledgement
IEEE 803-1988 Standard for SRS
Mode of Operation Template

3. Specific Requirements (continuation)

3.2 Functional Requirements

3.2.1 Mode 1

3.2.1.1 Functional Requirements 1.1
...
...
3.2.1.n Functional Requirements 1.n

3.2.2 Mode 2

3.2.2.1 Functional Requirements 2.1
...
...
3.2.2.n Functional Requirements 2.n

3.2.m Mode m

3.2.m.1 Functional Requirements m.1
...
3.2.m.n Functional Requirements m.n
IEEE 803-1988 Standard for SRS
Mode of Operation Template

3.3 Performance Requirements
3.4 Design Constraints
3.5 Software System Attributes
3.6 Other Requirements

4.0 Open Issues

5.0 Acknowledgement
The Mode of Operation Template
Overview of the SRS sections

- **The Introduction (Section 1)**
  - **Purpose:**
    - Delineate the purpose of your SRS
    - Specify the intended audience of the SRS
  - **Scope:**
    - First identify your software product by name:
      - *AudioMan Translator, Bill Payment Made Easy, Home Management System, The Bumble Bee Self Driving Car, OpenSong*
    - Explain what your software product will, and if necessary, will not do
    - Describe the application of the software, benefits, objectives and goals
  - **Definitions, Acronyms and Abbreviations**
    - Provide definitions of all terms, acronyms and abbreviations used in your document
  - **References**
    - Provide a list of all documents referenced in your document
      - Title, ID number, Date, Publishing Organization
  - **Outline**
    - Explain how the rest of your document is organized
The Mode of Operation Template
Overview of the SRS sections

- General Description of Product (Section 2)
  This section provides the necessary background info for writing the enumerated technical requirements (in section 3)

Context of Product and Product Functions
- Provide graphical representations and narratives of the requirements
  - Context Diagram
  - Use Case Scenarios
  - Message Sequence Charts
  - Applicable Narratives
- User Characteristics
  - Describe (in a narrative) general characteristics of the intended users of the product, including educational level, experience and technical expertise
- Constraints
  - Provide (in a narrative) general descriptions of items that will limit the developer’s options (e.g., COTS, 3rd Party software, Interface to other applications, Protocols etc..)
The Mode of Operation Template

Overview of the SRS sections

- General Description of Product (Section 2) continued

  - **Assumptions and Dependencies**
    - List each of the factors that affect the requirements (stated in section 3)
      - As an example, we may assume that Linux version $n.m$ will run on the hardware. If the version is not available, then changes may be made to the requirements (core/platform)
The Mode of Operation Template

Specific (Systems) Requirements

- **Systems Requirements (Section 3):**
  - This is the most important section of the SRS
    - It specifies detailed enumerated requirements to enable the software architect to design the software
    - Specific requirements should be cross-referenced to section 2 and other related documents
    - All requirements should be uniquely identifiable
Specific Requirements
External Interface (section 3)

- **User Interfaces**
  - **Specify** the following:
    - Logical characteristics of each interface between the software product and its users:
      - Screen formats
      - Window layout
      - Content of any reports/menus

- **Hardware Interfaces**
  - **Specify** the logical characteristics of each interface between the software product and the hardware components of the system
    - Configuration characteristics – number of ports
    - Type of devices supported (Server, PDA, Blackberry etc..)
Specific Requirements
External Interface (Section 3)

■ Software Interfaces
  □ Specify the following:
    ■ Usage of other required software products (e.g., math package, data management system)
    ■ Interfaces with other application systems
    ■ Provide the following attributes for each:
      □ Name of software
      □ Mnemonic
      □ Version number
      □ Source

■ Communications Interfaces
  □ Specify the various interfaces to communications:
    ■ Network protocols
Specific (Systems) Requirements
The Language

The language of Systems Requirements is different from Stakeholders (i.e. operational scenarios, constraints) requirements

For Systems Requirements:
- Use consistent language to identify different kinds of requirements
- We will use “shall” as a key word to indicate the presence of a requirement in the text

Example of function with capacity:
The <system> shall <function> not less than <quantity> <object>
The Communications system shall sustain telephone contact with not less than 10 callers

Example with periodicity constraints:
The <system> shall <function> <object>every <performance> <units>
The coffee machine shall produce a hot drink every 10 seconds
Systems Requirements
The Language (Examples)

- The Spanish speech subsystem shall transcribe Spanish to text in UNICODE format.

- The speech subsystem shall allow the end-user to initiate voice control commands to a PDA and simultaneously transcribe the end-user’s words as a Command Request (CRQ) message.

- The Speech subsystem shall transmit each CRQ in less than 2 seconds to the Home Management System (HMS).
Systems Requirements
Criteria for Requirements Statements

- **Criteria for Writing Requirements Statements**
  - Use simple direct language
    - Each statement is clearly understandable (clear)
  - Write one requirement at a time
    - Each statement is precise and concise (precise)
  - Each statement carries a single traceable element (atomic)
  - Write testable requirements
    - Technically possible within the project schedule (Feasible)
    - Each statement is verifiable, and it is known how (verifiable)
  - Requirements statements that belong together are close together (Category/modular)
  - Each statement can be **uniquely identified** *(We will talk about uniqueness shortly)*
Systems Requirements

Uniqueness

- Each Systems Requirement is uniquely identified by an enumeration (a set of tags)
  - We will adopt the following notations:

  <HMS–Speech-10> Speech Command
  The speech subsystem shall allow the end-user to initiate voice control commands to a PDA and simultaneously transcribe the end-user’s words as a Command Request (CRQ) message
  The speech commands are processed exclusively by the 3rd party subsystem independent of the HMS PDA platform.
  Source: Nuance™ software
  Version: 2.5

  <End of HMS-Speech-10>

  <HMS–Speech-20> Speech Command
  The Speech subsystem shall transmit each CRQ in less than 2 seconds to the Home Management System (HMS)
  <End of HMS-Speech-20>
Requirements Specifications

Summary

- Define the outline structure at the outset (customized version of IEEE-803)
- Write down your requirements asap based on your thought process captured in the Use Cases, MSCs and other documentations
- If you find holes in requirements, go back to the Use Cases and MSCs then make appropriate corrections their prior to modifying your systems requirements
- Determine in advance what attributes you will need to elaborate on the textual enumerated systems requirements
- Produce an initial version, brainstorm and iterate (Use Cases, MSCs, Req.)
- Use simple direct language
- Write testable requirements
- Write one requirement at a time