Organization of the SRS Document

In this section, we discuss the organization of an SRS document as prescribed by the IEEE 830 standard [IEEE 830]. Please note that IEEE 830 standard has been intended to serve only as a guideline for organizing a requirements specification document into sections and allows the flexibility of tailoring it, as may be required for specific projects. Depending on the type of project being handled, some sections can be omitted, introduced, or interchanged as may be considered prudent by the analyst. However, organization of the SRS document to a large extent depends on the preferences of the system analyst himself, and he is often guided in this by the policies and standards being followed by the development company. Also, the organization of the document and the issues discussed in it to a large extent depend on the type of the product being developed. However, irrespective of the company's principles and product type, the three basic issues that any SRS document should discuss are-functional requirements, non-functional requirements, and guidelines for system implementation.

The introduction section should describe the context in which the system is being developed, and provide an overall description of the system, and the environmental characteristics. The introduction section may include the hardware that the system will run on, the devices that the system will interact with and the user skill-levels. Description of the user skill-level is important, since the command language design and the presentation styles of the various documents depend to a large extent on the types of the users it is targeted for. For example, if the skill-levels of the users is "novice", it would mean that the user interface has to be very simple and rugged, whereas if the user-level is "advanced", several short cut techniques and advanced features may be provided in the user interface.

It is desirable to describe the formats for the input commands, input data, output reports, and if necessary the modes of interaction. We have already discussed how the contents of the Sections on the functional requirements, the non-functional requirements, and the goals of implementation should be written. In the following subsections, we outline the important sections that an SRS document should contain as suggested by the IEEE 830 standard, for each section of the document, we also briefly discuss the aspects that should be discussed in it.

Introduction

Purpose: This section should describe where the software would be deployed and how the software would be used.
Project scope: This section should briefly describe the overall context within which the software is being developed. For example, the parts of a problem that are being automated and the parts that would need to be automated during future evolution of the software.
Environmental characteristics: This section should briefly outline the environment (hardware and other software) with which the software will interact.

Overall description of organization of SRS document

Product perspective: This section needs to briefly state as to whether the software is intended to be a replacement for a certain existing system, or it is a new software. If the software being developed would be used as a component of a larger system, a simple schematic diagram can be given to show the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.
Product features: This section should summarize the major ways in which the software would be used. Details should be provided in Section 3 of the document. So, only a brief summary should be presented here.
User classes: Various user classes that are expected to use this software are identified and described here. The different classes of users are identified by the types of functionalities that they are expected to invoke, or their levels of expertise in using computers.
Operating environment: This section should discuss in some detail the hardware platform on which the software would run, the operating system, and other application software with which the developed software would interact.
Design and implementation constraints: In this section, the different constraints on the design and implementation are discussed. These might include—corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; specific programming language to be used; specific communication protocols to be used; security considerations; design conventions or programming standards.
User documentation: This section should list out the types of user documentation, such as user manuals, on-line help, and trouble-shooting manuals that will be delivered to the customer along with the software.

Functional requirements

This section can classify the functionalities either based on the specific functionalities invoked by different users, or the functionalities that are available in different modes, etc., depending what may be appropriate.

1. User class 1

(a) Functional requirement 1.1

(b) Functional requirement 1.2

2. User class 2

(a) Functional requirement 2.1

(b) Functional requirement 2.2

External interface requirements

User interfaces: This section should describe a high-level description of various interfaces and various principles to be followed. The user interface description may include sample screen images, any GUI standards or style guides that are to be followed, screen layout constraints, standard push buttons (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, etc. The details of the user interface design should be documented in a separate user interface specification document.

Hardware interfaces: This section should describe the interface between the software and the hardware components of the system. This section may include the description of the supported device types, the nature of the data and control interactions between the software and the hardware, and the communication protocols to be used.

Software interfaces: This section should describe the connections between this software and other specific software components, including databases, operating systems, tools, libraries, and integrated commercial components, etc. Identify the data items that would be input to the software and the data that would be output should be identified and the purpose of each should be described.

Communications interfaces: This section should describe the requirements associated with any type of communications required by the software, such as e-mail, web access, network server communications protocols, etc. This section should define any pertinent message formatting to be used. It should also identify any communication standards that will be used, such as TCP sockets, FTP, HTTP, or SHTTP. Specify any communication security or encryption issues that may be relevant, and also the data transfer rates, and synchronization mechanisms.

Other non-functional requirements for organization of SRS document

This section should describe the non-functional requirements other than the design and implementation constraints and the external interface requirements that have been described in Sections 2 and 4 respectively.

Performance requirements: Aspects such as number of transaction to be completed per second should be specified here. Some performance requirements may be specific

to individual functional requirements or features. These should also be specified here.

Safety requirements: Those requirements that are concerned with possible loss or damage that could result from the use of the software are specified here. For example, recovery after power failure, handling software and hardware failures, etc. may be documented here.

Security requirements: This section should specify any requirements regarding security or privacy requirements on data used or created by the software. Any user identity authentication requirements should be described here. It should also refer to any external policies or regulations concerning the security issues. Define any security or privacy certifications that must be satisfied.

For software that have distinct modes of operation, in the functional requirements section, the different modes of operation can be listed and in each mode the specific functionalities that are available for invocation can be organized as follows.

Functional requirements

1. Operation mode 1

(a) Functional requirement 1.1

(b) Functional requirement 1.2

2. Operation mode 2

(a) Functional requirement 2.1

(b) Functional requirement 2.2

Specification of the behavior may not be necessary for all systems. It is usually necessary for those systems in which the system behavior depends on the state in which the system is, and the system transits among a set of states depending on some prespecified conditions and events. The behavior of a system can be specified using either the finite state machine (FSM) formalism or any other alternate formalisms. The FSMs can used to specify the possible states (modes) of the system and the transition among these states due to occurrence of events.

Example: Personal library software

It is proposed to develop a software that would be used by individuals to manage their personal collection of books. The following is an informal description of the requirements of this software as worked out by the marketing department. Develop the functional and non-functional requirements for the software.

A person can have up to a few hundreds of books. The details of all the books such as name of the book, year of publication, date of purchase, price, and publisher would be entered by the owner. A book should be assigned a unique serial number by the computer. This number would be written by the owner using a pen on the inside page of the book. Only a registered friend can be lent a book. While registering a friend, the following data would have to be supplied - name of the friend, his address, land line number, and mobile number. Whenever a book issue request is given, the name of the friend to whom the book is to be issued and the unique id of the book is entered. At this, the various books outstanding against the borrower along with the date borrowed are displayed for information of the owner. If the owner wishes to go ahead with the issue

of the book, then the date of issue, the title of the book, and the unique identification number of the book are stored. When a friend returns a book, the date of return is stored and the book is removed from his borrowing list. Upon query, the software should display the name, address, and telephone numbers of each friend against whom books are outstanding along with the titles of the outstanding books and the date on which those were issued. The software should allow the owner to update the details of a friend such as his address, phone, telephone number, etc. It should be possible for the owner to delete all the data pertaining to a friend who is no more active in using the library. The records should be stored using a free (public domain) data base management system. The software should run on both Windows and UNIX machines.

Whenever the owner of the library software borrows a book from his friends, would enter the details regarding the title of the book, and the date borrowed and the friend from whom he borrowed it. Similarly, the return details of books would be entered. The software should be able to display all the books borrowed from various friends upon request by the owner.

It should be possible for anyone to query about the availability of a particular book through a web browser from any location. The owner should be able to query the total number of books in the personal library, and the total amount he has invested in his library. It should also be possible for him to view the number of books borrowed and returned by any (or all) friend(s) over any specified time.

Functional requirements

The software needs to support three categories of functionalities as described below:

1. Manage own books

R.1.1: Register book

Description: To register a book in the personal library, the details of a book, such as name, year of publication, date of purchase, price and publisher are entered. This is stored in the database and a unique serial number is generated.

Input: Book details

Output: Unique serial number

R.1.2: Issue book

Description: A friend can be issued book only if he is registered. The various books outstanding against him along with the date borrowed are first displayed.

R.1.2.1: Display outstanding books

Description: First a friend's name and the serial number of the book to be issued are entered. Then the books outstanding against the friend should be displayed.

Input: Friend name

Output: List of outstanding books along with the date on which each was borrowed.

R.1.2.2: Confirm issue book

If the owner confirms, then the book should be issued to him and the relevant records

should be updated.

Input: Owner confirmation for book issue.

Output: Confirmation of book issue.

R.1.3: Query outstanding books

Description: Details of friends who have books outstanding against their name is displayed.

Input: User selection

Output: The display includes the name, address and telephone numbers of each friend against whom books are outstanding along with the titles of the outstanding books and the date on which those were issued.

R.1.4: Query book

Description: Any user should be able to query a particular book from anywhere using a web browser.

Input: Name of the book.

Output: Availability of the book and whether the book is issued out.

R.1.5: Return book

Description: Upon return of a book by a friend, the date of return is stored and the book is removed from the borrowing list of the concerned friend.

Input: Name of the book.

Output: Confirmation message.

2. Manage friend details

R.2.1: Register friend

Description: A friend must be registered before he can be issued books. After the registration data is entered correctly, the data should be stored and a confirmation message should be displayed.

Input: Friend details including name of the friend, address, land line number and mobile number.

Output: Confirmation of registration status.

R.2.2: Update friend details

Description: When a friend's registration information changes, the same must be updated in the computer.

R.2.2.1: Display current details

Input: Friend name.

Output: Currently stored details.

R2.2.2: Update friend details

Input: Changes needed.

Output: Updated details with confirmation of the changes.

R.3.3: Delete a friend record

Description: Delete records of inactive members.

Input: Friend name.

Output: Confirmation message.

3. Manage borrowed books

R.3.1: Register borrowed books

Description: The books borrowed by the user of the personal library are registered.

Input: Title of the book and the date borrowed.

Output: Confirmation of the registration status.

R.3.2: Deregister borrowed books

Description: A borrowed book is deregistered when it is returned.

Input: Book name.

Output: Confirmation of deregistration.

R.3.3: Display borrowed books

Description: The data about the books borrowed by the owner are displayed.

Input: User selection.

Output: List of books borrowed from other friends.

4. Manage statistics

R.4.1: Display book count

Description: The total number of books in the personal library should be displayed.

Input: User selection.

Output: Count of books.

R.4.2: Display amount invested

Description: The total amount invested in the personal library is displayed.

Input: User selection.

Output: Total amount invested.

R.4.3: Display number of transactions Description:

Description: The total numbers of books issued and returned over a specific period by one (or all) friend(s) is displayed.

Input: Start of period and end of period.

Output: Total number of books issued and total number of books returned.

Non-functional requirements

N.1: Database: A data base management system that is available free of cost in the public domain should be used.

N.2: Platform: Both Windows and UNIX versions of the software need to be developed.

N.3: Web-support: It should be possible to invoke the query book functionality from any place by using a web browser.

Observation:

Since there are many functional requirements, the requirements have been organized into four sections: *Manage own books, manage friends, manage borrowed books,* and *manage statistics*. Now each section has less than 7 functional requirements. This would not only enhance the readability of the document, but would also help in design.