**Structure of the SRS (Software Requirements Specification)**

**as specified by IEEE standard 830-1993.**

**1 Introduction (Everyone)**

**1.1 Purpose**

*Comment: Purpose of the document and not the purpose of the software.*

Example: This document provides all of the requirements for the yyy. Parts 1 and 2 are intended primarily for customers of the application, but will also be of interest to software engineers building or maintaining the software. Part 3 is intended primarily for software engineers, but will also be of interest to customers.

**1.2 Scope**

*Comment: What aspects of the application this document tends to cover?*

Example: This document covers the requirements for release xxx of yyy. Mention will be made throughout this document of selected probable features of future releases. The purpose of this is to guide developers in selecting a design that will be able to accommodate the full-scale application.

**1.3 Definitions, Acronyms, and Abbreviations**

*Comment: Glossary of that will be used throughout the documents, as well as in other documents based on this one. Usually in the form of table.*

**1.4 References**

*Comment: References to all documents that are connected to this one.*

Example: Software Project management Plan for yyy, version xxx (*xxx is the vesrion of the document and not the version of the software*). Software Design Description for yyy, version xxx.

**1.5 Overview**

*Comment: Overview of the software. Its main goals, tasks and users.*

**2 Overall Description**

*Comment: Overall description of software - more detailed than in 1.5. This should be general enough so that it is unlikely to change much in future versions. Avoid statements that are repeated in later sections.*

**2.1 Product perspective**

*Comment: In this section software should be compared with other related or competing products, which is a good way to provide perspective of our product.*

2.1.1 System interfaces

*Comment: List all special interfaces to operating system.*

2.1.2 User interfaces

*Comment: Preliminary sketches and/or principles of key user interfaces only, used to provide perspective on the product. All user interfaces are described in detail in section 3.*

2.1.3 Hardware interfaces

*Comment: List all special hardware needed for software operation.*

Example: Joystick will be used as an input device for some functions of the software (enlisted later).

2.1.4 Software interfaces

*Comment: Interfaces with other software products.*

2.1.5 Communication interfaces

*Comment: List all communication interfaces.*

Example: Modem will provide access to Internet when necessary.

2.1.6 Memory constraints

*Comment: memory that is expected to be used by the software (external and internal).*

Example: yyy is expected to use no more than 16 MB of Ram and 20 MB of external storage.

2.1.7 Operations

*Comment: List all normal and special operations required by the user.*

Example: It must be possible to save and retirev the current state of the software.

2.1.8 Site adaptation requirements

*Comment: Requirements for execution on a particular installation*

Example: User interface must exist in three different languages: Hungarian, Serbian, and Slovak.

**2.2 Product functions**

*Comment: Summary of the major functions of the application. More detailed than section 1.5 but less detailed than section 3.*

Example: Use case can be used to describe the main functions. If done so, then here all use cases can be briefly described (and displayed).

**2.3 User characteristics**

*Comment: Indicate what kind of people the typical user are likely to be. For example: novice, software professional, accountant with 5 years of computer usage, etc.*

**2.4 Constraints**

*Comment: All conditions that may limit developer's options. These can originate from many sources.*

Example: yyy shall operate on PCs running Windows 95 or later at a minimum speed of 100 MHZ. Java shall be the implementation language.

**2.5 Assumptions and dependencies**

*Comment: Any assumptions being made.*

Example: Future versions of yyy shall operate on PCs running Linux.

**2.6 Apportioning of requirements**

*Comment: Order in which requirements are to be implemented.*

Example: The requirements described in sections 1 and 2 of this document are referred to as preliminary specifications; those in section 3 are referred to as requirements (or functional) specifications. The two levels of requirements are intended to be consistent. Inconsistencies are to be logged as defects. In the event that a requirement is stated within both preliminary and functional specifications, the application will be built from functional specification since it is more detailed.

'Essential requirements' (referred to in section 3) are to be implemented for this version of yyy. 'Desirable requirements' are to be implemented in this release if possible, but are not committed to by the developers. It is anticipated that they will be part of future release. 'Optional requirements' will be implemented at the discretion of developers.

**3 Specific requirements**

**3.1 External interface requirements**

3.1.1 User interfaces

*Comment: Description of user interface in section 2.1.2 showed only sketches of user interfaces in order to provide product perspective. It lacks details and should not be regarded as the last word. If user interfaces are not completely specified later in this document, then all details should be given in this section.*

3.1.2 Hardware interfaces

*Comment: Hardware that the software product deals with.*

3.1.3 Software interfaces

*Comment: Other software with which software product must interface.*

3.1.4 Communication interfaces

*Comment: Communication interfaces (Internet, modem, ...).*

**3.2 Classes/Objects**

*Comment: This style of SRS expects that detailed requirements are classified by classes. This section should list classes pertaining to the domain of the application and are adequate for organizing all of the requirements. These classes are not all of the classes that will be used by the application. Every function/class should be marked as 'essential', 'desirable', or 'optional'.*

**3.3 Performance requirements**

*Comment: Performance requirements include required speeds and/or time to complete. Unless documented in a different section of the SRS, they may also include memory usage (RAM and/or disk) noted either statically or dynamically (i.e., memory required at runtime).*

**3.4 Design constraints**

*Comment: Restrictions on design. If there is no material in this section, designers are free to create any (good) design that satisfies the requirements.*

**3.5 Software system attributes**

3.5.1 Reliability

Example: shall fail not more than once in a week. Reference to test documentation goes also here.

3.5.2 Availability

3.5.3 Security

Example: shall ask for user-name and password at the beginning. Passwords will be encrypted before saving.

3.5.4 Maintainability

*Comment: Lists of all functions/classes that are expected to change soon or to change frequently.*

**3.6 Other requirements**

**4 Supporting information**

**4.1 Table of Contents and Index**

**4.2 Appendices**

*Comment: This may include supporting or background information that can help the readers of SRS, description of the problem, special packaging instruction for the code and the media, results of user surveys, sample I/O formats, ...*

**3 Specific requirements**

**3.1 External interfaces**

**3.2 Functions**

**3.3 Performance requirements**

**3.4 Logical database requirements**

**3.5 Design constraints**

3.5.1 Standards compliance

**3.6 Software system attributes**

3.6.1 Reliability

3.6.2 Availability

3.6.3 Security

3.6.4 Maintainability

3.6.5 Portability

**3.7 Organizing the specific requirements**

3.7.1 System mode or

3.7.2 User class or

3.7.3 Objects or

3.7.4 Feature or

3.7.5 Stimulus or

3.7.6 Response or

3.7.7 Functional hierarchy or

3.7.1 Additional comments

1. Section 3 takes different structure depending on chosen style. This is "object-oriented" style.

2. Section 3 takes different structure depending on chosen style. This is "non object-oriented" style.