# **1 Introduction**

Some content under level 1.

## **1.1 Purpose**

Identify the purpose of this SRS and its intended audience.

## **1.2 Scope**

In this subsection:

(1)  Identify the software product(s) to be produced by name

(2)  Explain what the software product(s) will, and, if necessary, will not do

(3)  Describe the application of the software being specified.  As a portion of this, it should:

     (a)  Describe the relevant benefits, objectives, and goals as precisely as possible

        (b)  Be consistent with similar statements in higher-level specifications if they exist.

## **1.3 Definitions, Acronyms, and Abbreviations**

Provide the definitions of all terms, acronyms, and abbreviations required to properly interpret the SRS.  This information may be provided by reference to an appendix or other document(s).

## **1.4 References**

In this subsection:

(1)  Provide a complete list of all documents referenced elsewhere in the SRS.

(2)  Identify each document by title, report number  (if applicable), date, and publishing organization.

# **2 External Interface Requirements**

Describe the general factors that affect the product and its requirements.  This section usually consists of the five subsections that follow.  This section does not state specific requirements; each of its subsections makes those requirements easier to understand; they do not specify design or express specific requirements.  Such detail is provided in section 3.

## **2.1 Product Perspective**

(1)  If the product is independent and totally self-contained, it should be stated here.

(2)  If the SRS defines a product that is a component of a larger system or project:

        (a)  Describe the functions of each component of the larger system or project, and identify interfaces

        (b)  Identify the principal external interfaces of this software product (not a detailed description)

        (c)  Describe the computer hardware and peripheral equipment to be used (overview only)

A block diagram showing the major components of the larger system or project, interconnections, and external interfaces can be very helpful.

**EM-HLR-F-REQ-001**

Name : Equipment state

Description : Equipments regularly send signals (a frame) to give their state. Signals differ according to the equipment category.

Priority : Optional

Aasda asda sd sadaf dhgdhtrgr rt htr r a.

**EM-HLR-F-REQ-005**

Name : Permissions

Description : The target system should allow administrating the equipments and their categories.

Priority : Mandatory

Refine : EM-HLR-F-REQ-002

## **2.2 Product Functions**

Provide a summary of the functions that the software will perform. Sometimes the function summary that is necessary peratör part can be taken directly from the section of the higher-level specification (if one exists) that allocates particular functions to the software product.  The functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document peratö first time.  Block diagrams showing the different functions and their relationships can be helpful.  Such a diagram is not a requirement on the design of a product itself; it is simply an effective explanatory tool.

**EM-HLR-F-REQ-002**

Name : Case of failure

Description : In case of failure assumption (particular signal), advanced analysis should be performed and correlation should be done with other existing information to check the effective failure.

Priority : Mandatory

**EM-HLR-F-REQ-003**

Name : Handle failure

Description : If failure is confirmed, a procedure should be presented to the maintenance peratör to handle the failure.

Priority : Mandatory

## **2.3 User Characteristics**

Describe those general characteristics of the eventual users of the product that will affect the specific requirements.

Many people interact with a system during the operation and maintenance phase of the software life cycle.  Some of these people are users, operators, and maintenance and systems personnel.  Certain characteristics of these people, such as educational level, experience, and technical expertise impose important constraints on the system’s operating environment.

## **2.4 General Constraints**

Provide a general description of any other items that will limit the developer’s options for designing the system.  These can include:

(1)  Regulatory policies

(2)  Hardware limitations; for example, signal timing requirements

(3)  Interface to other applications

(4)  Parallel operation

(5)  Audit functions

(6)  Control functions

(7)  Higher-order language requirements

(8)  Signal handshake protocols; for example, XON-XOFF, ACK-NACK.

(9)  Criticality of the application

(10) Safety and security considerations

**EM-HLR-F-REQ-004**

Name : Procedure

Description : The procedure differs according to the equipment type.

Priority : Mandatory

**EM-HLR-F-REQ-006**

Name : Failures statistich

Description : Statistics about failures and incident resolutions should be activated if needed and should be exported as reports if requested by the chief peratör.

Dependency to : EM-HLR-F-REQ-003

Priority : Mandatory

This section of the SRS should contain all the details the software developer needs to create a design.  This is typically the largest and most important part of the SRS.

(1)  The details within it should be defined as individual specific requirements, following the guidelines for sound requirements (verifiable, unambiguous, etc.)

(2)  Specific requirements should be organized in a logical and readable fashion.

(3)  Each requirement should be stated such that its achievement can be objectively verified by a prescribed method.

(4)  Sources of a requirement should be identified where that is useful in understanding the requirement.

(5)  One way to classify the specific requirements is as follows:

        (a)  Functional Requirements

        (b)  Performance Requirements

        €  Design Constraints

        (d)  Attributes

        €  External Interface Requirements

The organization of this section of the SRS should be chosen with the goal of properly specifying the requirements in the most readable manner.

# **3 Specific Requirement**

This section of the SRS should contain all the details the software developer needs to create a design.  This is typically the largest and most important part of the SRS.

(1)  The details within it should be defined as individual specific requirements, following the guidelines for sound requirements (verifiable, unambiguous, etc.)

(2)  Specific requirements should be organized in a logical and readable fashion.

(3)  Each requirement should be stated such that its achievement can be objectively verified by a prescribed method.

(4)  Sources of a requirement should be identified where that is useful in understanding the requirement.

(5)  One way to classify the specific requirements is as follows:

        (a)  Functional Requirements

        (b)  Performance Requirements

        €  Design Constraints

        (d)  Attributes

        €  External Interface Requirements

The organization of this section of the SRS should be chosen with the goal of properly specifying the requirements in the most readable manner.

## **3.1 Functional Requirements**

This subsection of the SRS should specify what is to be done by the product,  to what level or specific requirement, what inputs should be transformed to what outputs (not **how** this is done), what specific operations are required.  Where the rationale for a requirement is not obvious, provide an explanation.  Where issues need to be resolved, cite those.

For each function, specify requirements on inputs, processing, and outputs.  These are usually organized with these four subparagraphs:

(1)  Purpose of the function:  Provide rationale to clarify the intent of the function.

(2)  Inputs:  sources, valid ranges of values, any timing concerns, peratör requirements, special interfaces

(3)  Operations to be performed: validity checks, responses to abnormal conditions, types of processing required

(4)  Outputs: destinations, valid ranges of values, timing concerns, handling of illegal values, error messages, interfaces required

## **3.2 External Interface Requirements**

This should specify:

The characteristics that the software must support for each human interface to the software product.  For example, if the user of the system operates through a display terminal, the following should be specified:

        (a)  Required screen formats

        (b)  Page layout and content of any reports or menus

        ©  Relative timing of inputs and outputs

        (d)  Availability of some form of programmable function keys.

(2)  All the aspects of optimizing the interface with the person who must use the system.  This may simply comprise a list of do’s and don’ts on how the system will appear to the user

 Specify the logical characteristics of each interface between the software product and the hardware components of the system. Include such matters as what devices are to be supported, how they are to be supported, and protocols.

 Specify the use of other required software products (for example, a data management system, an operating system, or a mathematical package), and interfaces with other application systems .

For each required software product, the following should be provided:

(1)  Name

(2)  Mnemonic

(3)  Specification Number

(4)  Version number

(5)  Source

For each interface:

(1)  Discuss the purpose of the interfacing software as related to this software product.

(2)  Define the interface in terms of message content and format.  It is not necessary to detail any well-documented interface, but a reference to the document defining the interface is required.

 Specify the various interfaces to communications such as local network protocols, etc.

## **3.3 Performance Requirements**

Software or on human interaction with the software, as a whole. Static numerical requirements may include:

        (a)  The number of terminals to be supported

        (b)  The number of simultaneous users to be supported

        ©  Number of files and records to be handled

        (d)  Sizes of tables and files

Static numerical requirements are sometimes identified under a separate section entitled capacity.

(2)  Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions.

All of these requirements should be stated in measurable terms, for example, 95% of the transactions shall be processed in less than 1 s, rather than, ort he shall not have to wait ort he transaction to complete.

Note:  Numerical limits applied to one specific function are normally specified as part of the processing subparagraph description of that function.

# **4 Supporting Information**

The supporting information; that is, the Table of Contents, the Appendices, and the Index, make the SRS easier to use.  The Appendices are not always considered part of the actual requirements specification and are not always necessary.  They might include:

        (a)  Sample I/O formats, descriptions of cost analysis studies, results of user surveys.

        (b)  Supporting or background information that can help the readers of the SRS.

        (c)  A description of the problems to be solved by the software.

        (d)  The history, background, experience and operational characteristics of the organization to be supported.

        (e)  A cross-reference list, arranged by milestone, of those incomplete software requirements that are to be completed by specified milestones.

        (f)  Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

(3)  When Appendices are included, the SRS should explicitly state whether or not the Appendices are to be considered part of the requirements.