**Recommended Practice for Software Requirements Specifications**

**(Based on IEEE Standard 830)**

1. **Introduction**: This section contains an overview of the complete document.
   1. **Purpose**

The main purpose of this document is to provide an overview of the specifications and requirements for the proposed software, to be read by the client and the software design team.

* 1. **Scope**

This document specifies requirements for an application to manage Emergency Room (ER) patients.

The application allows users to:

* Input and read basic information about a patient
* Order labs, and record and read the results
* Diagnosis problems with patient
* Admit patient for extended stay
* Prescribe medication for patients
* Label discharges
* Bill patients upon discharge in an itemized bill

(How is the application storing information/ documents?)

(Does the application run offline and store locally or were we going to have a database?)

* 1. **Definitions, acronyms and abbreviations** This subsection contains definitions of all the terms, acronyms and abbreviations used in the document. Special attention should be paid to the clarification of terms and concepts from the domain of application. (As Needed)
  2. **References** References to all documents that are referred to in the remainder of the requirements specification. (TBD)
  3. **Overview**

The rest of this document will provide a general description of the product, including the product perspective, the product functions, the product user characteristics, the general constraints, as well as any assumptions and dependencies.

This document will also describe any specific requirements, including functional requirements, external interface requirements, performance requirements, design constraints, and quality attributes of the application.

1. **General** **Description** This section contains a description of matters that concern the overall product and its requirements. It provides a perspective for understanding the specific requirements from section 3 of this document. (Katie)
   1. **Product** **perspective** Does it concern an independent product or is it part of a larger product? In the later case, the other components should be identified, and the interfaces with those components should be described. In this section, we also give an identification of the hardware to be used. (Independent Software)

**Staff and Financial Katie)**

* 1. **Functional** **Requirements** In this subsection, a description is given of how the transportation of inputs to outputs is achieved. The description given for each class of functions, and sometimes for each individual function. To a certain extent, this description can be seen as a solution to the user. This component of the requirement specification is the main starting point for the design phase.
  2. **Product** **functions** An overview of the functions of the system to be delivered. This should be confined to an overview. A detailed discussion of the functions is given in section 3 of the requirements specification. (
  3. **User** **characteristics** An indication of general user characteristics, in as far as these are relevant for the requirements specification. Experience, training and technical expertise of future users may influence specific requirements of the system to be developed. (Describe actual users i.e. nurses, doctors, etc.)
  4. **General** **Constraints** An indication of any other constraints that apply. These may concern government regulations, hardware constraints, security regulations, and so on. Again, we are concerned with the rationale at this point. A further elaboration follows in section 3 of this document. (General access constraints)
  5. **Assumptions** **and** **dependencies** This does not concern constraints on the system to be developed, but things which may influence the requirements specification once they change. As an example, we may think of the availability of certain supporting software, such as some given operating system or numeric library. If that operating system or library turns out not to be available, the requirements specification will have to be adapted accordingly. (TBD)

1. **Specific** **Requirements** This section contains all the details which are relevant for the design phase to follow. The ordering given here is just one way to present the specific requirements in a logical way. Specific requirements should be such that one may objectively determine whether they are fulfilled or not.
   1. Functions
      1. Order Labs and Tests
         1. Introduction

The application shall allow users of certain credentials to place orders for blood tests, x-rays, and other exams to be performed on a patient by a technician

* + - 1. Inputs

The application will take both the patient to be administered as a string tests and the type of test to be performed, as a string listed within a predefined enumeration of available tests.

* + - 1. Processing

The application will check to ensure that the patient exists within the database and that the ordered test is one of the available types.

The application will respond to abnormal inputs by refusing to place and order and warn the user about correct inputs.

The application will function with an all-or-nothing paradigm, either completing the order or doing nothing at all.

* + - 1. Outputs

The application will output a form or ticket with the patient name and ordered test as a (raw text file? Pdf? Not sure.)

* + 1. Write Discharge Instructions
       1. **Introduction** A description of the purpose of this function and the approaches and techniques used. The introduction should include information to clarify the intent of the function.

The application will allow users of specific credentials to write documents containing information regarding patient discharges.

* + - 1. **Inputs** A precise description of the function’s inputs(source, quantities, range of acceptable values, and the like).

The application will take a long string from the user.

* + - 1. **Processing** A definition of the operations that must be performed, such as checking for acceptable values, reaction to abnormal situations, or a description of algorithms to be used. As an example of the later, one may think of the use of some mathematical model for strength computations within a CAD-program.
      2. **Outputs** A precise description of the outputs(destination, quantities, error messages, and the like).

The application will output the discharge information as a (text file probably)

* + 1. Prescribe Medications
       1. Introduction

The application will allow specific user classes to prescribe medications from a predefined list to patients in order to help treat illnesses and other health issues.

The application will allow the user to prescribe multiple medications to a single patient.

* + - 1. Inputs

The application will take a string containing the name of the medication and an integer containing the amount of medication to prescribe.

* + - 1. Processing

The application will check that the name given is within the predefined list of medications, and warn the user if the provided name is not found.

The application will check that the amount of medication is within the valid range of integers no less than one.

* + - 1. Outputs

The application will return the paired information within a single ticket, as a (raw text file?).

3.1.4 Diagnose Diseases

* + - 1. Introduction

The application will allow specific user classes to diagnose diseases and other health problems of a patient from a predefined list of issues.

* + - 1. Inputs

The application will take a string containing the name of the health issue.

The application will take a string containing the patient’s name.

* + - 1. Processing

The application will check that the provided issue is found within the predefined list, warning the user and failing to process if it is not.

The application will check that the provided patient name is found within the patient database, warning the user and failing to process if it is not.

* + - 1. Outputs

The application will return the patient name and diagnosis as part of a ticket in a (text file?)

3.1.5 Order Discharge

* + - 1. Introduction

The application will allow users of a certain class to order a discharge for a patient from the hospital.

* + - 1. Inputs

The application will take a string containing the patient’s name.

The application will take the date and time of the discharge.

* + - 1. Processing

The application will check that the patient exists within the database, warning the user and failing to process if otherwise.

The application will check that the date and time are valid, warning the user and failing to process if otherwise.

* + - 1. Outputs

The application will output both information as part of a single ticket, as a (text file?)

* 1. **External** **interface** **requirements**
     1. **User** **interfaces** A description of the characteristics of the user interfaces, such as screen layout, function keys, help functions. In order to support testing, verifiable requirements regarding learning time for the system functions should be included either here or in some subsection of 3.5 (Attributes).
     2. **Hardware** **interfaces** A description of the logical characteristics of hardware interfaces, such as interface protocols, or screen-oriented versus line-oriented terminal control. (Basic Work Computer, system already in place)
     3. **Software** **interfaces** A description of software needed, such as a certain operating system or subroutine package. Interfaces to other application software is also discussed here. (Basic software)
     4. **Communications** **interfaces** An example is a communication protocol for LANs. (N/A or TBD)
  2. **Performance** **requirements** Performance requirements encompass both static and dynamic requirements. Static requirements concern, amongst others, the number of terminals to be connected and the number of users that can be handled concurrently. Dynamic requirements concern the operational performance of the system: how frequently will certain functions be called for and how fast should the system’s reaction be. It is important that these requirements be stated in measurable terms. (TBD)
  3. **Design** **constraints** Design constraints may result from such things as the prescribed use of certain standards or hardware. (Ask about it, possibly comparable to max number of people admitted to ER)
     1. **Standards** **compliance** Which existing standards or regulations must be followed, and what requirements result from these. For example, certain report formats or audit procedures may be prescribed.
     2. **Hardware** **limitations** A description of the characteristics of the hardware environment, in as far as they lead to software requirements. An example of this might be the amount of memory available.
  4. **Attributes** In this section, particular attention is paid to quality aspects. These requirements must be measurable and verifiable. They must be stated in objective terms. The subsections below by no means comprise a complete list of such attributes.
     1. **Availability** Factors that guarantee a certain level of availability, such as restart procedures. In this subsection we may also enlist requirements regarding fault tolerance (with respect to both hardware failures and software failures).
     2. **Security** **Requirements** regarding unauthorized access and other forms of misuse. Certain cryptographic techniques may be prescribed, and we may put constraints on the communication between different parts of the system. (Ask)
     3. **Maintainability** Requirements to guarantee a certain level of maintainability of the system, such as a maximum allowable coupling between components. (Ask)
  5. **Other** **requirements** A description of requirements that are specific to certain software, and which have not been discussed yet. (TBD)