
Requirement Specification for Local Monitoring System

**Embedded Real Time Systems
Spring 2010**

Amos Hoste hoste_amos@hotmail.com
Michalis Voliotis mvoliotis@gmail.com
Torben Grøn Helligsø thel@gmx.com
José Antonio Esparza jaesparza@gmail.com

Version Hisotry

| Ver. | Date | Initials | Description |
|------|------------|--------------|---|
| 1.00 | 15.04.2010 | TGH & JAE | First version of the document |
| 2.0 | 16.04.2010 | TGH & JAE | Use cases description added to the document |
| | | | |

Table of Contents

| | | |
|-------|--|----|
| 1. | INTRODUCTION | 2 |
| 1.1 | Purpose | 2 |
| 1.2 | References | 2 |
| 1.3 | Reading Guide | 2 |
| 2. | OVERALL DESCRIPTION | 3 |
| 2.1 | System Description | 3 |
| 2.1.1 | System Overview | 3 |
| 2.1.2 | Actor-Context Diagram | 4 |
| 2.1.3 | Actor Descriptions | 4 |
| 2.1.4 | Relations between Actor Roles, Job Position and Hardware | 5 |
| 2.2 | System Functions | 5 |
| 2.2.1 | Use Case Diagram(s) | 6 |
| 2.3 | System Constraints | 7 |
| 2.4 | System Future | 8 |
| 2.5 | User Characteristics | 8 |
| 2.6 | Development Process Requirements | 8 |
| 2.7 | Customer Deliveries | 9 |
| 2.8 | Prerequisites | 9 |
| 3. | SPECIFIC REQUIREMENTS (USE CASES) | 10 |
| 3.1 | Edit alarm | 10 |
| 3.2 | Add new alarm | 10 |
| 3.3 | Delete alarm | 11 |
| 3.4 | Configure display settings | 11 |
| 3.5 | Register new input | 12 |
| 3.6 | Trigger alarm | 13 |
| 3.7 | Display readings | 13 |
| 3.8 | Store readings locally | 14 |
| 3.9 | Update software | 14 |
| 3.10 | Supervise I-Pump | 15 |
| 4. | EXTERNAL INTERFACE REQUIREMENTS | 16 |
| 4.1 | User Interfaces | 16 |
| 4.2 | Hardware Interfaces | 16 |
| 4.3 | Communication Interfaces | 16 |
| 4.4 | Software Interfaces | 16 |
| 5. | PERFORMANCE REQUIREMENTS | 17 |
| 6. | SYSTEM QUALITIES | 17 |
| 7. | DESIGN CONSTRAINTS | 17 |
| 8. | OTHER REQUIREMENTS | 17 |
| 8.1 | Authority Requirements | 17 |
| 8.2 | Other requirements | 17 |
| 9. | PART DELIVERIES | 18 |
| 10. | Appendixes | 19 |
| 10.1 | Data Definitions | 19 |
| 10.2 | Word List | 19 |

1. INTRODUCTION

1.1 Purpose

The following pages define the requirements for the system LMON: Local Monitoring System. The Local Monitoring System is a device that displays vital signals such as Electrocardiographic signal, pulse or oximetry, that are relevant to supervise the patient state.

The system will be developed as an academic project for the course Embedded Real Time System, taking place at the Engineering College of Aarhus.

1.2 References

This requirement specification is based upon the following documents:

- TI-IRTS Project Interface Specification (version 16.02.2004)
- Project description for Local Monitor System (LMON)

1.3 Reading Guide

The present document is composed by 9 main chapters. The first one introduces the project and references to relevant papers to the requirement analysis stage.

In the second chapter, Overall Description, an overview of the system requirements will be presented.

The third chapter, Specification Requirements, will detail the functional requirements by using UML compliant use-case diagrams.

The non functional requirements, will be exposed in the chapters 4 -6, External Interface Requirements, Performance Requirements and System Qualities.

In chapter 7 the design constraints that the system should meet will be presented.

Chapter 8 will detail additional requirements that affect the system.

In chapter 9 the estimated deliveries to the customer will be detailed.

2. OVERALL DESCRIPTION

2.1 System Description

2.1.1 System Overview

The LOCAL monitor system is an electronic device that is able to display a set of vital signs to allow constant patient monitoring. This system is intended to be used in combination with special transducers, able to transform human variables like pulse, oximetry or temperature into electrical signals, which will be provided as direct inputs to the system.

Patients' associated readings will be directly presented to the sanitary personal by a LCD touch screen display, allowing direct access to the current measurements through a window-based friendly user interface.

Depending on patient state, several alarms can be configured and automatically triggered if there is a potential threat to the patient health, displaying a message on the screen and playing an alarm sound.



Figure 1: System Overview Diagram. The Patient vital signs are measured by using the appropriate transducers and sent to the LMON patient monitor system.



Figure 2: Infusion pump device (left) and ECG transducers (right).

The system will be portable and compact, and it will be possible to operate it in a standalone mode or connected to an Infusion Pump, in order to control the medicine flow to the patient depending on current patient conditions.

2.1.2 Actor-Context Diagram

The following diagram shows the main actors involved with the Local Monitor Operation. Each actor represents a single role that one or several users can hold at a single time. In the following sections the involved actors will be described in detail.

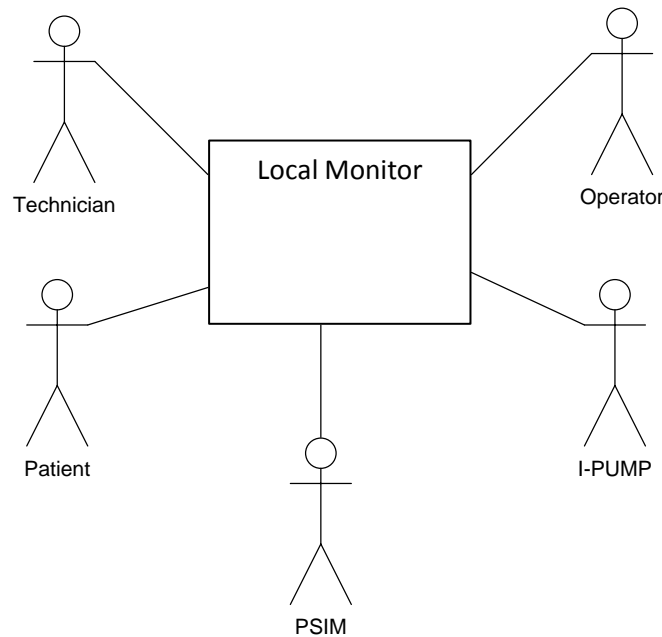


Figure 3: Actor-Context Diagram

2.1.3 Actor Descriptions

| | |
|------------------------------------|--|
| Actor Name | Operator |
| Type [primary/secondary] | Primary |
| Description | The operator represents any sanitary personal that is controlling the patient vital signs by using the LMON system |
| Number of concurrent actors | 1 |

| | |
|------------------------------------|---|
| Actor Name | Patient |
| Type [primary/secondary] | Primary |
| Description | The patient is connected to the LMON systems and represents the subject under supervision, providing the system active readings on ECG signals, pulse, temperature and any required variable. |
| Number of concurrent actors | 1 |
| Actor Name | PSIM – Patient Simulator |

| | |
|------------------------------------|---|
| Type [primary/secondary] | Primary |
| Description | The patient simulator is an artificial system capable of producing human vital signs, so it can be used in training scenarios. The LMON system can be used in combination with a PSIM system. |
| Number of concurrent actors | 1 |

| | |
|------------------------------------|---|
| Actor Name | Technician |
| Type [primary/secondary] | Secondary |
| Description | The technician is responsible for maintaining the system, performing updates and fixing possible technical problems |
| Number of concurrent actors | 1 |

| | |
|------------------------------------|---|
| Actor Name | I-PUM |
| Type [primary/secondary] | Secondary |
| Description | The I-PUM is an automatic drug delivery system that is controlled by the LMON. The Local Monitor will be able to precise the required drug flow and the pump state. |
| Number of concurrent actors | 1 |

2.1.4 Relations between Actor Roles, Job Position and Hardware

The following table shows the relations between a given job position and the roles played by the given person. Use X for a primary role and (X) for a more secondary role.

| Actor / Job Position | Patient | Operator | I-PUM | Technician | LMON | PSIM |
|----------------------------------|----------------|-----------------|--------------|-------------------|-------------|-------------|
| Nurse | | X | | | | |
| Doctor | | X | | | | |
| Drug delivery responsible | | X | (X) | | X | X |
| Vital Signs Provider | X | | | | | (X) |
| System expert | | | | X | | |

Figure 4: Relations between actor roles and job positions

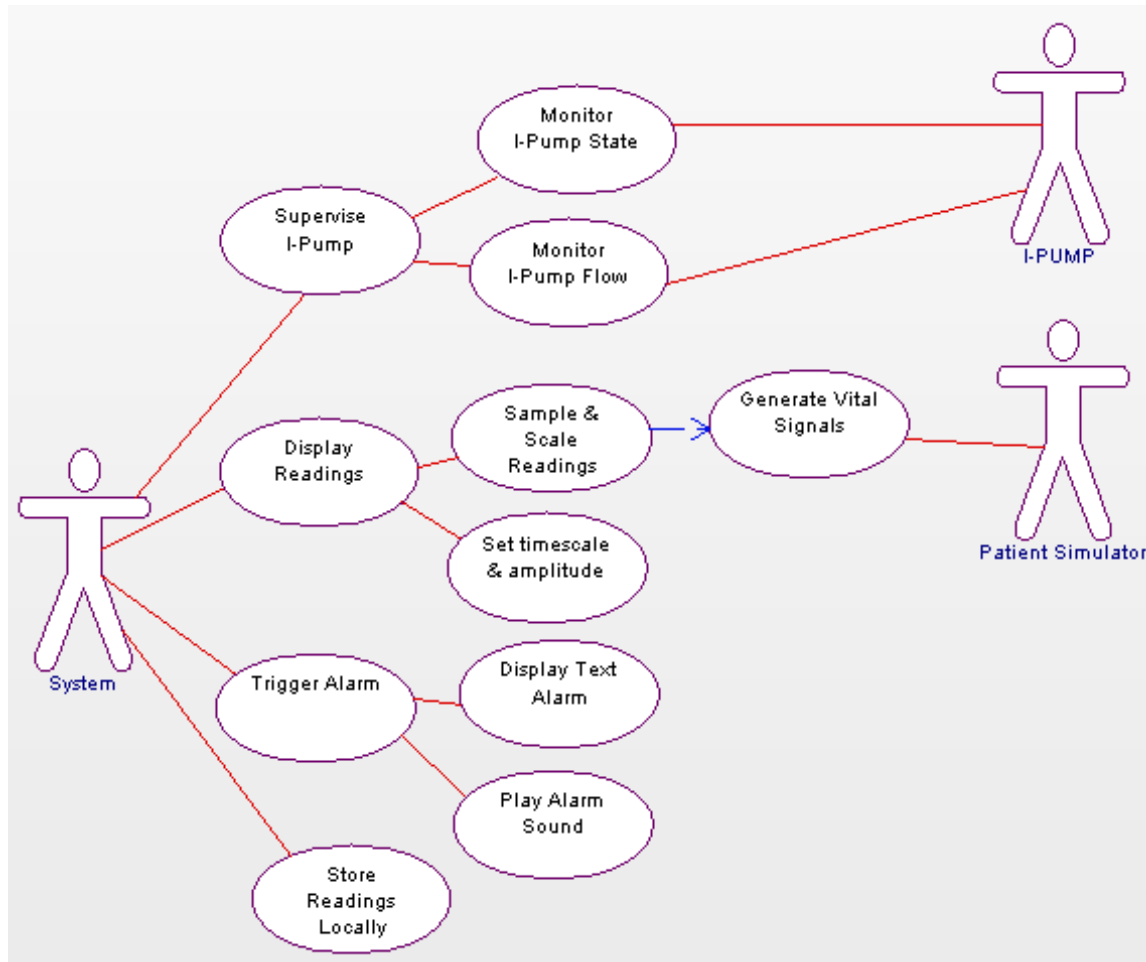
2.2 System Functions

The system functions is found and described by the Use Case technique. The following diagrams show the system functionality expressed in one or more Use case diagrams. The purpose of these diagrams is to give an overview of the wanted system

functionality. Each of the Use Cases on the diagrams is specified in detail in chapter 3. Specific Requirements.

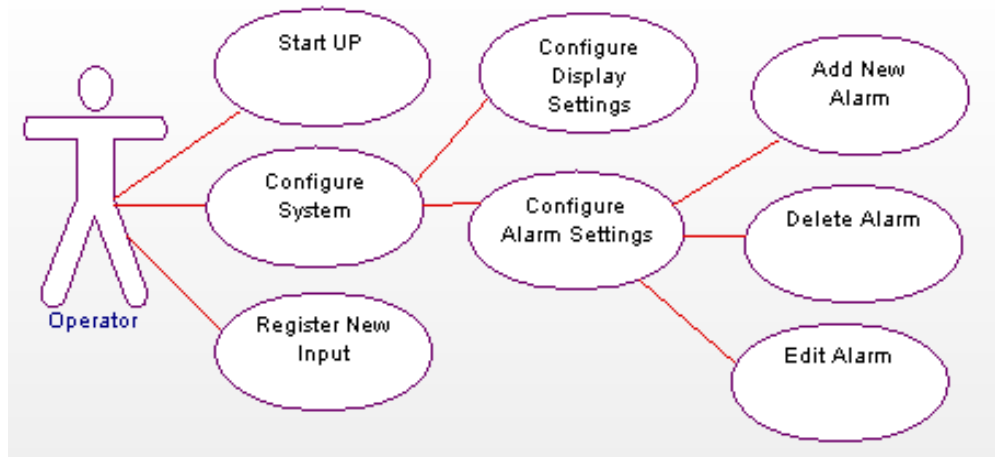
2.2.1 Use Case Diagram(s)

In the following diagrams the main functionalities the system should provide are presented by UML use case diagrams. They are also showing how the different actors are related depending on the interaction they are maintaining with the system.



Figur 4. Use Case Diagram

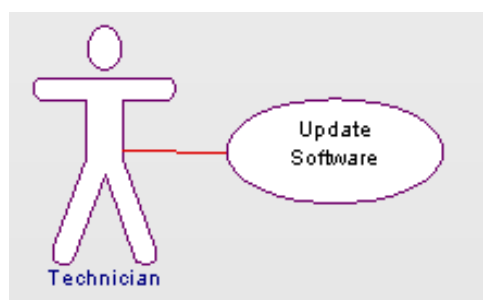
The system is autonomous and it should be able to control and automate some actions related to the patient state. That implies it should be able to trigger different alarms depending on the patient vital signs and to control the medicine flow through an infusion pump. An additional feature consists on keeping a log about previous readings that can be consulted by the physiatrist or operator, a primary actor that interacts with the system as it is shown in the next use case diagram.



The operator will be responsible for setting the variables he is interested to monitor and configure the appropriate alarms associated with them. It will be possible for him to configure the desired display settings as well.

A typical workflow involving the actors System, Operator, I-PUMP and Patient Simulator would be the following one:

A new patient needs to be monitored in order to keep track on his vital signs. The operator connects the probes and starts the LMON system and configures the alarms parameters, so in the case vital signs are critical an alarm will be triggered. The system starts sampling and scaling the readings coming from the probes and display them in the LCD, while simultaneously started to store them in a local file for later retrieval in the case it is necessary. After 20 minutes the patient presents serious arrhythmia so an alarm sound is played and a message is display in the system LCD. The doctor configures an I-PUMP system so the appropriate drug can be administrated to the patient automatically controlled by the LMON. After this the system can monitor the patient vital signs and keep track on the medicine flow and I-PUM state as simultaneously.



In the case that some maintenance is required, a technician can manually update the LMON software.

2.3 System Constraints

The LMON system will not be able to monitor several patients at the same time.

The LMON system will work locally and it will not present any kind of networking capabilities.

Although the LMON system is able to control the drug flow while connected to an I-PUMP, an expert Operator should supervise the process and check if it is being correctly performed.

2.4 System Future

The system will provide the appropriate interfaces so it can be connected to a wired Ethernet network and monitor remote patients. At this point, it will be possible to monitor several patients at the same time.

The software architecture will be conceived in order to support updates in the case they are required.

2.5 User Characteristics

The system can be connected to a human patient or a simulated one. These two subjects are supposed to be in conditions to generate vital signs.

The system should be operated by personal with the necessary background an education to connect and interpret the information the LMON is showing. That implies the operator should be a nurse or a doctor with experience with this kind of equipment.

In the case the system is operated by a technician, he should have received concrete instructions to update the software and to determine if the system presents a malfunctioning that cannot be handled by him.

2.6 Development Process Requirements

The system will be developed by using iterative incremental process. This will allow the development team to show periodically the system evolution to the customer through periodical deliveries. The customer will provide feedback based on the deliverables and his comments and suggestions will be taken in consideration in the following iterations.

The LMON system software will be developed with object oriented techniques, particularly using the C++ language. Even though the main software architecture will be developed in C++, some specific controllers and drivers will be developed in C, but they should be wrapped in order to reach a perfect integration with the other components.

The LMON will run the Linux OS system, increasing software portability.

The user interface will be developed by using the QT libraries, that are portable to several platforms.

The development team will provide the customer the following documentation:

- Requirement specification document: This document will expose the functionalities the system will present and define the scope of the project. (present paper)
- Project report: an academic document explaining how the concepts introduce in the course “Embedded Real Time Systems” have been applied to the present project.
- Product report: A technical paper containing documents regarding the software architecture, UML diagrams, software-hardware integration, testing and QA, provided interfaces and other technical issues that do not fit in the project report.
- Developed source code.

The previous documents will be provided printed and recorded in a CD-ROM. The source code will be delivered to the customer recorded in the CD-ROM, but not printed.

2.7 Customer Deliveries

Scheduled deliveries

| Date | Delivery |
|------------------------------|---|
| 16 th April, 2010 | Requirement Specification Document |
| 4 th June, 2010 | Project Report, Product Report, Source Code |

Weekly status report will be presented to the customer in order to get feedback on the already developed components.

2.8 Prerequisites

The development team should have an target system devkit8000 capable to run the Linux OS with an add-on board attached, providing the required interfaces to communicate with the required hardware.

A patient simulator should be available in order to perform integration tests.

3. SPECIFIC REQUIREMENTS (USE CASES)

3.1 Edit alarm

| | |
|-----------------------------|---|
| Goal | To edit an existing alarm |
| Initiation | Initiated by operator |
| Actors and Stake Holders | Operator |
| No of concurrent instances | 1 |
| Frequency | Occasional |
| Non functional requirements | |
| References | Add new alarm |
| Preconditions | The target alarm should be already registered in the system |
| Postconditions by success | System uses updated settings |
| Postconditions by failures | The alarm settings should remain unaltered |
| Main Scenario | <ol style="list-style-type: none"> 1. Operator selects 'Alarms' menu 2. Display shows list of defined alarms. [note: alarms are named by input instance and alarm limit] 3. Operator selects alarm to edit. 4. Operator pushes 'Edit' button 5. Display shows alarm settings for the selected alarm [note: settings include data input, when to raise alarm (above, below, between), and associated limits, severity 1-3 (currently unused)] 6. Operator makes his changes. 7. Operator pushes 'Save' button. 8. System saves alarm settings 9. System shows data sample window. |
| Extensions | 3a, 5a, 7a: Operator selects 'Cancel' button System shows data sample window. Use case stops. |

3.2 Add new alarm

| | |
|-----------------------------|--|
| Goal | To define a new alarm |
| Initiation | Initiated by operator |
| Actors and Stake Holders | Operator |
| No of concurrent instances | 1 |
| Frequency | Occasional |
| Non functional requirements | |
| References | Edit Alarm. |
| Preconditions | The alarm should not be already registered in the system |

| | |
|----------------------------|---|
| Postconditions by success | The alarm is registered by the system |
| Postconditions by failures | The alarm is not registered in the system and this does not alter previous state. |
| Main Scenario | <ol style="list-style-type: none"> 1. Operator selects 'Alarms' menu 2. Display shows list of defined alarms. [note: alarms are named by input instance and alarm limit] 3. Operator pushes 'Add' button 4. Display shows alarm settings for the new alarm [note: settings include data input, when to raise alarm (above, below, between), and associated limits, severity 1-3 (currently unused)] 5. Operator makes his changes. 6. Operator pushes 'Save' button. 7. System saves alarm settings 8. System shows data sample window. |
| Extensions | 6a: Operator selects 'Cancel' button System shows data sample window. Use case stops. |

3.3 Delete alarm

| | |
|-----------------------------|--|
| Goal | To delete an existing alarm |
| Initiation | Initiated by operator |
| Actors and Stake Holders | Operator |
| No of concurrent instances | 1 |
| Frequency | Occasional |
| Non functional requirements | |
| References | Add new alarm, Edit existing alarm |
| Preconditions | The alarm should be registered in the system |
| Postconditions by success | The alarm will be deleted from the system |
| Postconditions by failures | The alarm will remain unaltered |
| Main Scenario | <ol style="list-style-type: none"> 1. Operator selects 'Alarms' menu 2. Display shows list of defined alarms. [note: alarms are named by input instance and alarm limit] 3. Operator pushes 'Delete' button 4. |
| Extensions | 6a: Operator selects 'Cancel' button System shows data sample window. Use case stops. |

3.4 Configure display settings

| | |
|------|---|
| Goal | Used to change how samples from an input are presented on the display |
|------|---|

| | |
|-----------------------------|---|
| Initiation | Initiated by operator |
| Actors and Stake Holders | Operator |
| No of concurrent instances | 1 |
| Frequency | Often |
| Non functional requirements | The system should have an LCD or an external monitor connected |
| References | |
| Preconditions | The desired input should be registered in the system |
| Postconditions by success | The display settings configured by the operator is used. |
| Postconditions by failures | The system remain unaltered. |
| Main Scenario | <ol style="list-style-type: none"> 1. Operator selects 'Input' menu 2. Display shows list of defined inputs. 3. Operator selects an input and pushes the 'Display settings' button. 4. Display shows current display settings for the selected input. 5. Operator changes settings. [Settings are type (waveform, numerical), color. For a waveform amplitude, time scale, sample rate. For a numeric input number of decimals] 6. Operator selects the 'Save' button. 7. System shows data sample window. |
| Extensions | 5a, 6a: Operator selects 'Cancel' button System shows data sample window. Use case stops. |

3.5 Register new input

| | |
|-----------------------------|--|
| Goal | To show samples from an additional input |
| Initiation | Initiated by operator |
| Actors and Stake Holders | Operator |
| No of concurrent instances | 1 |
| Frequency | Often |
| Non functional requirements | |
| References | |
| Preconditions | <p>The input should present a signal suitable to be measured by the LMON system.</p> <p>Physical inputs should be available.</p> |
| Postconditions by success | The systems shows samples from an additional input on its display . |
| Postconditions by failures | The system will remain unaltered |

| | |
|---------------|--|
| Main Scenario | <ol style="list-style-type: none"> 1. Operator selects 'Input' menu 2. Display shows list of defined inputs. 3. Operator pushes the 'New' button. 4. Display shows a list of input types. 5. Operator selects an input type. 6. Operator selects the 'Register' button. 7. System shows data sample window. |
| Extensions | 5a, 6a: Operator selects 'Cancel' button System shows data sample window. Use case stops. |

3.6 Trigger alarm

| | |
|-----------------------------|---|
| Goal | To give the operator visual and sound feedback when an alarm is raised. |
| Initiation | Initiated by the system |
| Actors and Stake Holders | System |
| No of concurrent instances | 1 |
| Frequency | |
| Non functional requirements | The system should have a LCD or external monitor connected in order to display the messages and a speaker to play the alarm sound. |
| References | |
| Preconditions | A potentially harmful situation that could threat patients health should occur. |
| Postconditions by success | The system shows and sounds an alarm. |
| Postconditions by failures | |
| Main Scenario | <ol style="list-style-type: none"> 1. The system flashes background of readout in case of numeric input, or the background of legend in case of waveform input. The system displays a text alarm. 2. The system plays alarm sound |
| Extensions | |

3.7 Display readings

| | |
|-----------------------------|---|
| Goal | To update the display with new data read from an input. |
| Initiation | System initiated |
| Actors and Stake Holders | A single input |
| No of concurrent instances | One for each defined input |
| Frequency | User defined, for instance once per second |
| Non functional requirements | A LCD or monitor should be connected to the system |

| | |
|----------------------------|--|
| References | |
| Preconditions | Readings should be registered as inputs |
| Postconditions by success | Display shows data read from input |
| Postconditions by failures | |
| Main Scenario | <ol style="list-style-type: none"> 1. System reads data from input 2. System scales the data read. 3A. For numeric input: System formats and updates display with data read. 3B. For waveform data: System updates graph to show last 10 samples read. |
| Extensions | |

3.8 Store readings locally

| | |
|-----------------------------|--|
| Goal | To maintain a history of samples from the last 4 hours |
| Initiation | System initiated |
| Actors and Stake Holders | A single input |
| No of concurrent instances | One for each defined input |
| Frequency | As defined by the operator |
| Non functional requirements | A permanent storage device should be connected to the system. |
| References | Register new input. |
| Preconditions | Readings should be sampled scaled and registered as inputs in the system |
| Postconditions by success | Input sample history contains has entries for no more than 4 hours |
| Postconditions by failures | |
| Main Scenario | <ol style="list-style-type: none"> 1. System reads data from input 2. System scales, formats and stores the data read. |
| Extensions | |

3.9 Update software

| | |
|-----------------------------|---|
| Goal | To install a new version of the system software |
| Initiation | Technician |
| Actors and Stake Holders | Technician, System |
| No of concurrent instances | 1 |
| Frequency | Rare |
| Non functional requirements | System should not be in use, nor connected to a patient |
| References | |

| | |
|----------------------------|--|
| Preconditions | A system out of date or with erroneous software |
| Postconditions by success | The system uses the new software version |
| Postconditions by failures | The system still uses the old software version |
| Main Scenario | 1 The doctor takes apart the LMON 2 The technician replaces existing firmware 3 The technician execute the maintenance task defined by the company after a software update |
| Extensions | |

3.10 Supervise I-Pump

| | |
|-----------------------------|--|
| Goal | To send commands to the I-PUMP |
| Initiation | Initiated by operator |
| Actors and Stake Holders | Operator |
| No of concurrent instances | 1 |
| Frequency | Occasionally |
| Non functional requirements | |
| References | Display readings, register new input, trigger alarm |
| Preconditions | The operator has added the infusion pump as an input by performing the Register new input use case The I-PUMP is powered on. |
| Postconditions by success | The command is correctly sent to the I-PUMP. |
| Postconditions by failures | Trigger alarm |
| Main Scenario | 1. The system sends a command to the I-PUMP 2. The system receives an acknowledgement from the I-PUMP |
| Extensions | 2A. <i>The system does not receive an acknowledgement from the I-PUMP.</i> 2A.1. An alarm is raised using the Trigger alarm use case. |

4. EXTERNAL INTERFACE REQUIREMENTS

4.1 User Interfaces

The user interface will be using English to interact with the users.

The system will display readable messages to the user in the case an error happen as well as an associated error code, so it can be reported to the technician in the case additional maintenance is required.

The system will play loud different alarms depending on the critical situations that could threat the patient state.

4.2 Hardware Interfaces

The system should have the following interfaces:

- RS232 interface Connection with the I-PUM system
- A/D converter. Connected to the transducers that are reading signals from the patient.
- Analog inputs. Connected to the patient temperature sensor.
- LCD touch screen. Allowing human-machine interaction.

4.3 Communication Interfaces

The communication protocol that is used to interact with the I-PUMP system is explained in the document titled “TI-IRTS Project Interface Specification (version 16.02.2004).

The use of the D/A converters under the devkit8000 is described in the website dev8000.wikispaces.com

4.4 Software Interfaces

The software will run under the Linux OS system.

The software will provide the appropriate interfaces to work with the D/A converter mounted on the devkit8000, as well as an appropriate set of functions that will allow it to deal with different D/A converters. For example, in the case that a higher resolution is required a new D/A converter could be used.

The software will implement an interface between the displayer software component and the application core, in this way the main processing task will be separated from the graphical user interface. This interface will be useful in the case a remote monitor should be developed. In that case the remote monitor should implement the same interface and communicate it with the system through a network connection.

5. PERFORMANCE REQUIREMENTS

To be defined.

6. SYSTEM QUALITIES

Reliability

The system will offer a reliable representation of the received signals at the inputs.

Software portability

The software responsible for LMON system will be portable to other Linux platforms easily.

Extensibility

The developed software will be easy to extend in order to support further functionalities.

Friendly user interaction

The Graphical User Interfaces that the system will be presenting to the operator will be user-friendly, displaying only the relevant information at each particular case.

7. DESIGN CONSTRAINTS

The system must be developed in order to be deployed on a devkit8000 board.

The system will be running on top of a Linux OS.

The software must be developed under the object oriented paradigm.

The software will be developed in C++ and it will use the QT libraries in order to implement the graphical user interface.

8. OTHER REQUIREMENTS

8.1 Authority Requirements

Since this is an academic project, the system is not obligated to fulfil any kind of standards or limitations.

8.2 Other requirements

9. PART DELIVERIES

Partial Delivery 1

Requirement Specification Document
Scheduled for 16th of April 2010

Partial Delivery 2

Design Document, detailing the software architecture and the class design
Scheduled for 23rd of April 2010

In the deliveries 3 to 7, the following use cases will be presented to the customer.

Partial Delivery 3

Display readings functionality

Partial Delivery 4

Alarm control

Partial Delivery 5

Variable registration
System configuration

Partial Delivery 6

Readings storage

Partial Delivery 7

I-PUM supervision

Final Delivery

Project documentation, Product documentation and Software
Scheduled for 4th of June 2010.

10. Appendixes

10.1 Data Definitions

The electrocardiographic and oximetry signals will be continuous in time.

The blood pressure and temperature will be discrete values.

The temperature will be expressed in Celsius degrees.

The blood pressure will be expressed in mmHg.

10.2 Word List

- | | |
|----------------------|--|
| ▪ A/D | Analog to Digital Converter. Electronical device responsible for sampling and scaling an analog continuous signal in order to convert it into a digital one. |
| ▪ Electrocardiograph | Continuous signal representing the electrical activity of the heart |
| ▪ Ethernet | Standard for network communications |
| ▪ GUI | Graphical User Interface. Software responsible for user interaction through elements such as buttons, windows, text areas... |
| ▪ I-PUMP | Infusion Pump. Electromechanical device responsible for automatic drug delivery. |
| ▪ Linux | Open-source operating system |
| ▪ LMON | Local Monitor. Device introduced in this paper, able to monitor patient vital signs |
| ▪ mmHg | Millimeter of mercury, pressure unit |
| ▪ Oximetry | Continuous signal representing the patient's hemoglobin oxygenation |
| ▪ P-SIM | Patient Simulator. Electromechanical device able to reproduce human vital signals. Used in medical training. |
| ▪ QT | Open-source libraries used to create Graphical User Interfaces |
| ▪ RS232 | Serial communication standard |
| ▪ Transducer | Electromechanical device able to convert a mechanical signal into an electrical one. |

