**Project documentation**

**Emergency call button**

Date: **14/02-11**

Advisor: Kim Bjerge

Project participants: Jørgen Vrou Hansen (20042728)

Saiid Shah Alizadeh (19963592)

Anders Hvidgaard Poder (19951439)

Revision: **A**

Document ID: **PRODDOC**

# Introduction

This document details the process, decisions and result of the Emergency button project. A prerequisite for reading this document is to have read the project proposal and the one-page intro.

The document begins by describing the process and then continues to describe the individual parts of the project as it progresses through the process. Naturally the process is iterative in nature, yet in the document any changes will be described under the section they relate to, and the document may therefore not be seen as a chronological detail of the process. Where changes occurs the point of origin will naturally be described along with its affect to earlier section in the change log.

# Process

1. Analyse project proposal and create SRD
   1. Domain model
   2. System level requirements
      1. Use cases.
      2. Sequence diagrams where needed.
      3. Non-functional requirements.
2. Refine SRD to SRS
   1. State and activity diagrams to clarify use cases.
   2. Detailed requirements from system requirements – Should we do this?
   3. Requirement traceability from system to detailed requirements.
   4. Traceability for changed requirements.
3. Overall architectural design.
   1. Identify blocks and create overall structure
   2. Mapping of blocks to requirements, both in diagram and RVTM (V is not part of report)
   3. Create internal block diagram for important blocks.
   4. Create activity, state, sequence and other diagrams where needed.
   5. Modify requirements if needed.
4. SystemC TLM of overall architectural design
   1. Map functional blocks to SystemC module.
   2. Create communication channels (mostly standard FIFO).
   3. Modify architecture/requirements if needed.
5. Architecture mapping
   1. Identify alternative architectures.
   2. Create architectural design for each alternative.
6. Process mapping for each alternative architecture
   1. Identify processes.
   2. Identify communication.
   3. Map Processes to PE and Communication channels to CE
7. SystemC Timed TLM for each alternative architectures
   1. Update and refine SystemC to the alternative architecture
   2. Identify delayes in the proceses and communication channel based on rough estimation.
   3. Implement delays in SystemC.
   4. Simulate the system and compare the results.
8. Conclusion
   1. Evaluate the pros and cons of the alternative architectures.
   2. Evaluate the process.

# Domain model



Figure 1 - Domain model drawing

In Figure 1 may be seen the overall parts of the Emergency call system including their communication paths. Only a small part of the above is part of this project, yet it is important to realize the domain in which the project operates.

The system functions as follows:

1. An emergency call base is installed in the home of the person receiving care and an emergency call button is issued to that person.
2. The Emergency button communicates with the emergency call base using the ISM network.
3. The emergency call base communicates with the phone company server using its built-in GSM modem and SIM card via a Base Tranceiver Station (BTS).
4. The phone company server forwards the communication via the internet to/from the emergency call server (naturally in a properly protected tunnel).
5. The emergency call server forwards the communication to/from the interested parties; technician, care giver or both.

The above may be formalized in a SysML Domain Basic Block Diagram as shown in Figure 2.



Figure 2 - Emergency call system Domain

In Figure 2 may be seen not only the System Of Interest, but also the environment in which it operates and the blocks with which it interconnects.

The environment in which the emergency call button operates is one where it may be operated by an elderly or disabled person, and may therefore be exposed to some degree of moisture and shocks, as well as worn continuously for a long time.

The ISM network naturally suffers from limited range, noise and interference, which is all part of the ISM network.

The Emergency call base is the communication gateway between the Emergency call server and the Emergency call button. It also has other responsibilities, yet they are irrelevant for this project. The communication goes via the GSM network, which also suffers from limited range, noise and interference as well as a third party service provider (the phone company). In the target area it is believed that with a sufficiently large and well placed GSM antenna a sufficient signal may always be achieved (assumption). Furthermore the GSM network has been thoroughly tested and the service providers have strong incentives to keep a very high availability.

The Emergency call server is placed in a protected environment with sufficient power and network access, and can reach the technician and caregiver on duty using either the head-office LAN , SMS Gateways and/or other third party services.

Focussing on the System-Of-Interest a set of requirements may be created from the project proposal. As the focus on this project is proof-of-concept some of the detail requirements (e.g. colour) will be excluded just like only the critical requirements, especially those containing implemtational risks, will be broken down in the architecture.

# Requirements

This section contains both the overall system requirements (SRD) and the refined requirements (SRS) in one, as the project is an in-house project with close proximity to the “customer”. It therefore does not make sense to create two separate documents, but is far more efficient to simply create the SRS right away.

Again the requirements are only the system requirements and do not include all detailed requirements, yet should contain sufficient data to complete a proof-of-concept.

TODO: Should we insert the requirements here, or refer to the document?

TODO: Rational for the individual requirements?

# Architecture