

Trackme RASD

REQUIREMENTS ANALYSIS AND SPECIFICATION DOMAIN (RASD)

Release date: 11/11/2018

Version: 0.1 (usiamo 0.xx per indicare le bozze, la 1 sarà per la definitiva)

Come table of content propongo il sommario, che poi sarà pure quello che intende il prof. Come font propongo Georgia o Calibrì light (come quello del progetto di reti logiche), come tema il terzo da sinistra di office. Grazie ciao

Table of content

[Introduction 2](#_Toc529290215)

[Purpose 2](#_Toc529290216)

[Scope 2](#_Toc529290217)

[Goals for the client 2](#_Toc529290218)

[Goals for the third party 2](#_Toc529290219)

[Definitions, acronyms, Abbreviations 3](#_Toc529290220)

[Definitions 3](#_Toc529290221)

[Revision history 3](#_Toc529290222)

[Reference Documents 3](#_Toc529290223)

[Document structure 3](#_Toc529290224)

[Overall description 3](#_Toc529290225)

[Product perspective 4](#_Toc529290226)

[Product functions 4](#_Toc529290227)

[User characteristics 4](#_Toc529290228)

[Assumptions, dependencies and constraints 4](#_Toc529290229)

[Specific requirements 4](#_Toc529290230)

[External Interface Requirements 4](#_Toc529290231)

[User interfaces 4](#_Toc529290232)

[Hardware Interfaces 4](#_Toc529290233)

[Software Interfaces 4](#_Toc529290234)

[Communication Interfaces 4](#_Toc529290235)

[Functional Requirements 4](#_Toc529290236)

[Non functional Requirements 4](#_Toc529290237)

[Performance 4](#_Toc529290238)

[Accuracy 4](#_Toc529290239)

[Design Constraints 5](#_Toc529290240)

[Standards compliance 5](#_Toc529290241)

[Hardware limitations 5](#_Toc529290242)

[Any other constraint 5](#_Toc529290243)

[Software System Attributes 5](#_Toc529290244)

[Availability and reliability 5](#_Toc529290245)

[Security 5](#_Toc529290246)

[Formal analysis using alloy 5](#_Toc529290247)

[Effort spent 5](#_Toc529290248)

[References 5](#_Toc529290249)

# Introduction

## Purpose

## Scope

In this part we expose the phenomena we consider relevant in order to model the part of world of interest: they guided us to spot goals and requirements.

### World phenomena

* Diseases /absence of disease
* Emergency situations
* Ambulance moves and arrives
* Ambulance breakdown
* GPS does not work
* Server breakdown

### Shared phenomena

[Controlled by the world]

* Detection of health values
* Request for data from the patient or the third part
* Ambulance ack
* Confirmation of good health status by the patient
* Registration to the service by a user or a third party

[The following ones have been put here and not in machine phenomena because, in our modelling, they can be in some ways detected by the server or other components of our machine]

* Sensor breakdown
* Application breakdown
* Network breakdown

**[controlled by the machine]**

* Sending data to the third party
* Showing data to the patient
* Sending an alert to an ambulance
* asking the patient to confirm his health status

### Machine phenomena

* Database queries
* Database inserts
* Database creation
* Data analysis and comparation with thresholds
* Communication between the application and the server: i.e., sending and receiving messages between the app and the server
* Matching ambulances and emergencies
* Data elaboration for showing
* Data storing by the application

### Goals for the client

* [G1] Provide a form of unique identification (registration/login) of all patients using the application;
* [G2] prevent third parties from associating a single user to his data without his permission;
* [G3] Whenever a user is in danger of life, the application is working and there is internet connection, an ambulance is alerted;
* [G4] If something is not working as expected (the sensor, the application, the network), the patient’s family is alerted within an hour;
* [G5] Allow the user to see, under request, reports on his vital parameters and data about his health;

### Goals for the third party

* [G6] Provide a form of unique identification (registration/login) of all companies using the application;
* [G7] Allow third parties to access data if and only if they could be anonymized;
* [G8] Allow third parties to access data to specific individuals’ data under their permission;
* [G9] Allow third parties to specify constrains in their researches;
* [G10] Allow third parties to subscribe to new data;

## Definitions, acronyms, Abbreviations

### Definitions

* “***Health status***”: when in the following parts we state “health status” we are meaning the following values:
  + Blood saturation: it’s an indicator of the status of lungs and of respiratory system in general (e.g. to detect suffocation)
  + Hearth rate: it’s an indicator of hearth diseases (to detect hearth attacks)
  + Blood pression: it hardly ever helps to detect an emergency, but it’s useful for third parties and statistics (blood pressure out of range can indicate/cause a huge number of chronical diseases)
  + Body temperature: it’s an indicator of fever
  + Patient’s falling: if the patient has suddenly fallen there could be various causes and effects that, though other values are not able to detect them, put in serious risk the patient’s life.
* ***“Data Anonymization”***: deleting the fiscal code associated to every data tuple obtained by the query;

### Acronyms

* [G-n]: n-goal
* [D-n]: n-domain assumption
* [R-n]: n-functional requirement
* RASD: Requirements analysis and specification domain

## Revision history

## Reference Documents

## Document structure

# Overall description

## Product perspective

## 

## Product functions

## User characteristics

### Actors

* Patient: the person who uses the application, wearing the device and allowing the application to monitor his health status and to manage his data;
* Third party: a company which is interested in monitoring population’s health status and obtaining a useful resource of data (e.g. a health insurance, a pharmaceutical company, the government, an hospital);

## Assumptions, dependencies and constraints

# Specific requirements

## External Interface Requirements

### User interfaces

### Hardware Interfaces

### Software Interfaces

### Communication Interfaces

## Functional Requirements

### [G1] Provide a form of unique identification (registration/login) of all patients using the application

* [R1] If the user does not insert username and password the application does not let the user access any functionality[login]
* [D1] The user has correctly downloaded the application from the online store on his device (smartwatch, smartphone, etc) [i.e. he is not using a crack version]
* [R3] If the user declares that it has not a valid username or password (i.e. it’s the first access), first the application shows him a registration form;
* [R4] If the user does not fill the registration form with his fiscal code and all other personal data, the application refuses the registration
* [R5] If the username provided in the registration form is already in use, the application refuses the registration

### [G2] Prevent third parties from associating a single user to his data without his permission

* [R5] If a third part asks for data of a single user, data are shown if and only if he concedes his permission
* [R6] if a third part asks for data that involves less than 1000 people, the application refuses
* [R7] if a third part asks for data that involves more than 1000 people, the application anonymizes data before sending

### [G3] Whenever a user is in danger of life, the application is working and there is internet connection, an ambulance is alerted

* [R8] If the application does not read properly input data every 500 Ms, it informs the user that the sensor is not working as expected, suggests contacting the customer service as soon as possible and asks for confirmation of good health status.
* [R9]If the user does not respond to confirmation within 5 minutes, the machine calls the emergency number

### [G4] If something is not working as expected (the sensor, the application, the network), the patient’s family is alerted within an hour

* If the application does not send data for back up purpose every hour, the server sends a message to the emergency number, provided through the registration form (R)
* The emergency number is correct (D)
* If the user does not fill the registration form with his fiscal code and all other personal data, the application refuses the registration (R) (requirement taken from goal 1)
* If the application informs the server that the sensor is not working as expected, the server sends a message t the emergency number (R) [I propose, to avoid including machine phenomena*,* If the application *notices* that the sensor is not working as expected, the server sends a message to the emergency number (R)]

## Non functional Requirements

Performance

Of course, we need a fast reaction to emergency. Concretely, we state that the machine must guarantee a reaction time of less than 5 seconds from the time the parameters are below the thresholds. In addiction to this, 4G connection is required to ensure immediate communication with the server.

* + Concerning the server, we opt for an event-based architecture
  + [QUI NON STO ESAGERANDO??????….] SLIDE 5D
  + [FORSE CI TORNERà UTILE PER IL DD]

Accuracy

This is the non-functional requirement we consider the most relevant. While assuming (here and in all other parts of RASD, cfr, *domain assumptions*) that GPS works properly, our biggest concern is about the sensor which detects values: there is a concrete risk of frequent false positives, caused by some little offsets. Such cases are particularly annoying because we can’t ignore the alarms but at the same time, they make us waste time, money and resources, threatening the correct working of our machine. We pointed out that there is a trade-off between this issue (which would request to wait for a lot of consecutive signals of emergency to be sure) and the issue of performance (which would request to react immediately). We decided to wait for 1,5 sec before alerting, which means, values below thresholds for three times in a row. However, provided that false positives are defined in statistics as *type one errors,* we state that the sensor must have prob (type I err) < 1%;

### Availability and reliability

* Concerning the server: We need a server available 24/7 to handle emergency messages as fast as possible because, among other reasons, there is no way for the app the handle an emergency without the server;
* Concerning the app: we also need availability for the app, but not (with so much relevance) reliability , due to the fact that it takes time to detect a disease ( the sensor does not send data in real time ecc..), in which there could be an app break down with no significative consequences for the service ( provided that recovery time is under 500 ms).

Security

Thinking of a market such as the American one, where health care is subject to negotiation between patients and companies, security of all sensitive information which could advantage malicious companies, is a very important concern for our application. Due to this reason, data encryption should be implemented in communications between the app and the server and the web apps.

## Design Constraints

### Standards compliance

### Hardware limitations

Our software needs 2 physical devices to work and communicate properly at the same time. In order to deal with this strong hardware limitation, we came up with some mechanisms of detection of breakdowns

* iOS or Android smartphone with 2G/3G/4G connection and Bluetooth connection
* 2G/3G/4G connection
* GPS connection
* Wearable device with Bluetooth connection

In alternative

* iOS or Android smartwatch
* 2G/3G/4G connection
* GPS connection

For visualizing data,

* Modern browser able to render graphs and statistical models

### Any other constraint

## Software System Attributes

# Formal analysis using alloy

# Effort spent

# References