POLITECNICO DI MILANO



SOFTWARE ENGINEERING 2 COURSE

**Data4Help + AutomatedSOS (TROVARE ALTRO TITOLO)**

**Requirements Analysis and Specification Document**

Kevin Mato, Antonio Mazzeo

Github: https://github.com/antoniomazzeo/MatoMazzeo

DATA

**Sommario**

[**1** **Introduction** 2](#_Toc528255142)

[**1.1** **Purpose** 2](#_Toc528255143)

[**1.2** **Scope** 2](#_Toc528255144)

[**1.3** **Definitions, Acronyms, Abbreviations** 3](#_Toc528255145)

[**1.4** **Revision History** 3](#_Toc528255146)

[**1.5** **Reference Documents** 4](#_Toc528255147)

[**1.6** **Document Structure** 4](#_Toc528255148)

[**2** **Overall Description** 5](#_Toc528255149)

[**2.1** **Product Perspective** 5](#_Toc528255150)

[**2.2** **Product Functions** 6](#_Toc528255151)

[**2.3** **User Characteristics** 6](#_Toc528255152)

[**2.4** **Domain assumptions, dependencies and constraints** 6](#_Toc528255153)

# **Introduction**

This section will explain which the main scopes of the Data4Help and AutomatedSOS applications are and will provide a general overview of all the features involved.

## **Purpose**

This document is the RASD (Requirement Analysis and Specification Document) and its main purpose is to describe the Data4Help system developed by TrackMe company, its components, constraints, functional and non-functional requirements and relationships with third party software, particularly with the AutomatedSOS system developed by the same company.

This document will also provide different scenarios along with typical user use cases, a structure of the system using UML diagrams and a formal description of application features through alloy language.

This document is aimed at giving a detailed system specification to developers, testers and quality assurance.

## **Scope**

The scope of the entire project is to develop a new mobile application, Data4Help, compatible with the most popular smartwatches and bands available in the market which stores and shares health and location data with third party services, this document will particularly focus on the interaction with AutomatedSOS system, a software aimed at helping elderly people by contacting an ambulance in case of emergency (i.e. heart beat under or above the safe parameters, etc.).

Data4Help must share and store user data only by explicit consent respecting the General Data Protection Regulation (GDPR).

AutomatedSOS must constantly monitor user’s health, if the values go outside the safe intervals given by the WHO (World Health Organization), it must contact an ambulance.

This RASD (Requirement and Analysis Document) aims at describing in the most clear way the domain in which our system is going to work and determine the main use cases for this system.

This document will also define the primary users of the system, why the system is developed, the condition in which this is going to operate. Moreover we will illustrate the functional and non-functional requirements for a better understanding of the system.

The goals achieved by this solution are :

1. Third parties can analyze the users individually or in groups the users thy are interested in.
2. Third parties are constantly up-to-date on new data about the users.
3. Users can decide whether to give away their location to other stakeholders of the system.
4. Users can decide whether to give away their health parameters to other stakeholders of the system.
5. Elderly users can receive automatically help if there is an anomaly in their health parameters.
6. Any kind of user has a form identification for proving their identity.

## **Definitions, Acronyms, Abbreviations**

**Definitions**

* Platform: system/application as a whole.
* User: An end user who is currently registered to the Data4Help application and has credentials to access.
* Guest: Person not yet registered and with limited access to features.
* Framework: Reusable set of libraries or classes for a software system.
* Cross-Platform: software able to run on different platforms.

**Acronyms**

* RASD: Requirements Analysis and Specification Document
* DB: Database
* DBMS: Database Management System
* OS: Operating System
* HTML: HyperText Markup Language
* CSS: Cascading Style Sheets
* JS: JavaScript
* JSON: JavaScript Object Notation
* API: Application Programming Environment
* HTTP: HyperText Transfer Protocol
* HTTPS: HyperText Transfer Protocol Secure
* TCP: Transmission Control Protocol

**Abbreviations**

* Gn: n-th goal
* Rn: n-th functional requirement
* Dn: n-th domain assumption

(Eliminare definizioni, acronimi e abbreviazioni non utilizzati prima della consegna)

## **Revision History**

Version, date and summary

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Summary** |
| 1.0.0 | Data prima consegna | First Release |

## **Reference Documents**

The following documents were used:

1. Assignment document: <https://beep.metid.polimi.it/documents/121843524/3f744351-7378-4162-86b0-45eddaf10713>
2. IEEE Std 1016tm-2009 Standard for Information Technology-System Design-Software Design Descriptions.
3. IEEE Std 830-1998 IEEE Recommended Practice for Software Requirements Specifications.

## **Document Structure**

The document is divided in the following sections:

* **Section 1:** General description, overview and goals of the TrackMe applications (Data4Help and AutomatedSOS).
* **Section 2:** Overall description of the software features and implemented functions. Description of world and shared phenomena, constraints, domain assumptions and dependencies.
* **Section 3:** Characterization of functional and not functional requirements.
* **Section 4:** Formal model of the main aspects of the application using Alloy modelling language.
* **Section 5:** Effort spent for writing the project.
* **Section 6:** References and software used for the creation of the system.

# **Overall Description**

## **Product Perspective**

Both Data4Help and AutomatedSOS will be cross-platform web application which will be able to run on every mobile device.

AutomatedSOS will gather data from Data4Help through custom API.

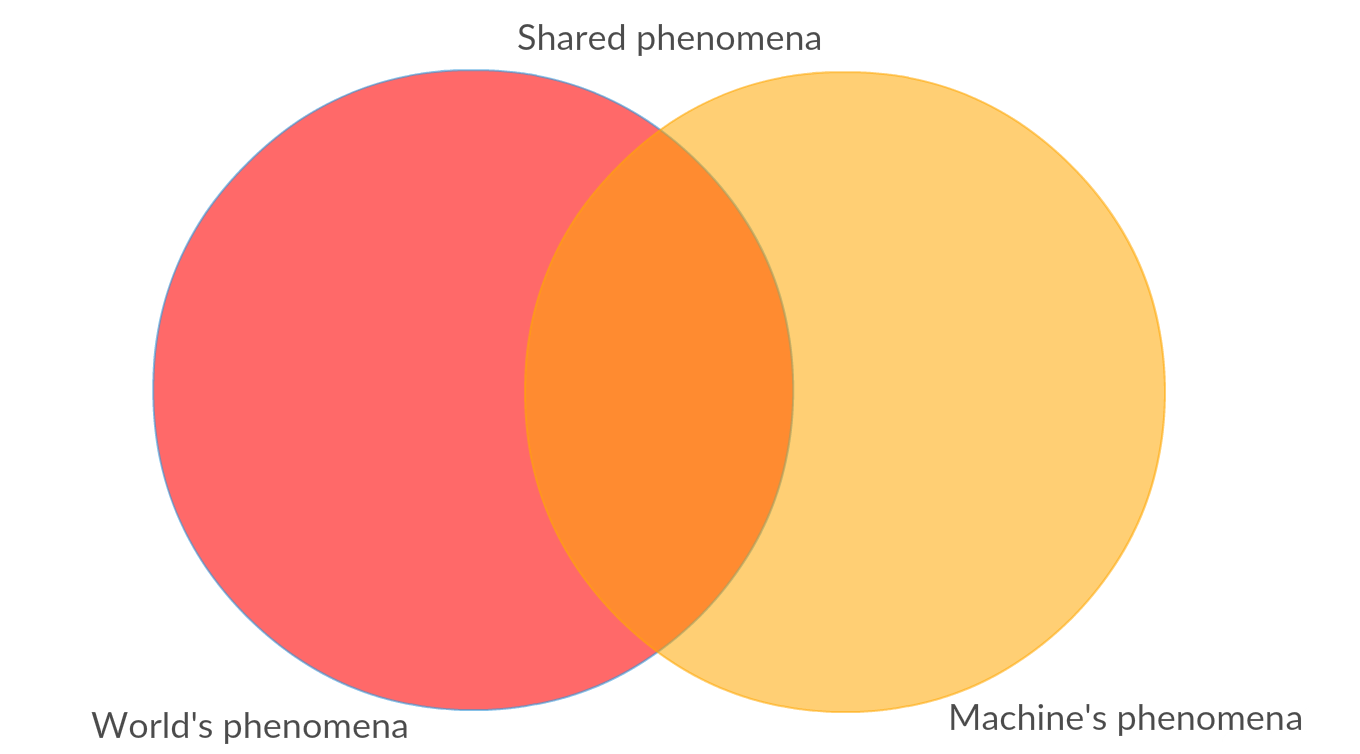


Figure 2.1 Relation between world and machine phenomena

**World** is everything that is outside the system scope and **machine** isthe opposite. Therefore, **world’s phenomena** are all the external events happening in the outside world and **machine’s phenomena** are events related to the system. **Shared phenomena** are the intersection of the two i.e. events observable by both parts.

What can be a **world’s phenomenon**?

For instance:

1. the weather for a certain day is extremely hot, accompanied by a high humidity in the air.
2. The diet followed by the user is rich in unsaturated fats.

What can be a **machine’s phenomenon**?

1. The registration of a new user in the database.
2. A thread spins up to save an event, after the computation of an anomaly.

What can be a **shared phenomenon**?

1. Locating a user.
2. Notifying an ambulance with the address of a user.
3. User’s data.

2.2 Product Function

Our system should be very simple to use and its strong points should be the ease with which the users could understand the statistics concerned about their health. The third parties should also have data about groups of users.

The system should also be able to:

1. Let the user register and log in.
2. Ask users for their data on behalf of the third parties.
3. Let a user choose and edit his privacy preferences.
4. Make itself easy-to-use to everyone.
5. Compute aggregations of users according to third parties preferences.
6. Give a real time evaluation of the data.
7. Notify a third party if certain data are not available and why.
8. Interact with sharing services.
9. Send emergency help to users interpreted as in danger.
10. Avoid any misuse of data.
11. Understand and store location and health status of a user.
12. Can identify people as unique entities thanks to the information they provided.

## **User Characteristics**

To use the application, no special skills are required. To understand the data and the information computed by the system a user needs a certain level of knowledge, which differs from what usage he wants to make.

The actors in the system are:

* **Guests**: People not registered yet. Guests cannot access any application functionality apart sign up.
* **Users**: People currently registered to the system, they can log in and access full service with no restriction.

There’s a big difference between the two kinds of users we have.

* **The third parties** possess very technical skills. They have statitians working on the result of the system, because they can understand every feature of the data and organize a strategy according to these.
* **The normal user** is someone who has the ability to use an app but can’t keep track of his health parameters or just wants an easy and understandable way for doing this.

## **Domain assumptions, dependencies and constraints**

**Constraints**

1. **Hardware limitations:** The client application under study must run on every mobile device as smartphones and tablets, regardless of the OS (i.e. Android, iOS, …).

For AutomatedSOS the user must have a smartwatch or smartband with heart beat monitor capabilities. Unfortunately, there’s no standard API to gather information from such devices, so only the most famous companies’ devices will be supported (Fitbit, Samsung, Garmin, Xiaomi, Sony).

1. **Interface to other applications:** As said in the previous point, the application needs to interface itself with other third-party applications to read health data from the user.
2. **Parallel operations:** The application must be able to handle not blocking parallel requests from several users grating a high reactivity.
3. **Availability limitations:** The system’s uptime and availability must be around 99.999%, corresponding to a maximum downtime of 5 minutes in the whole year, since the application deals with people’s lives.
4. **Privacy constraints:** The application must only handle data that the user previously accepted to share.

Moreover, a third-party can access data of single or group of users only if:

1. The single user accepts the third-party request.
2. The cardinality of the group of users fitting the given criteria is greater or equal than 1000.

**Domain assumptions and Dependencies**

1. **Health-smart device**.  
   All the users have a device that collects health data.
2. **GPS devices.**All users have a device that possesses a built-in GPS functionality.
3. **Connectivity**.  
   All the devices that take part to the system, have a stable connection.
4. **Veracity of the data.**  
   All the information about the users are correct.
5. **Well rounded algorithms.**  
   The algorithms for anomaly detection are infallible.
6. **Help** **availability in time.**  
   Ambulances can be contacted anytime.
7. **Help Range.**  
   Ambulances are spread on the areas of the users of the system in the most optimized way.
8. **Data Mining possibilities.**  
   Users give all the personal information needed to be clustered in groups thanks to different objective functions.
9. **Privacy Regulations.**  
   There are strict regulations about privacy.
10. **Big Data oriented.**  
    All the information is structured.
11. **There are no privileged users.**
12. **Every user is independent from each other.**
13. **Every user has several contact info.**
14. **Timestamps sharing.**  
    Every agent in the system share the same time.
15. **API Availablity.**  
    Whatever will interact with the system has available API.
16. **Battery Duration.**  
    Every user keeps its devices charged continuously.
17. **OS Permission Granted.**   
    The user will always grant to his OS’s device the permission to access to all the needed services.
18. **Sharing services availability.**  
    The sharing services interacting with the system have a greater uptime than the system itself.
19. **Host Availability.**The system has a greater uptime than the single clients.

# **Specific Requirements**

## **External Interfaces Requirements**

**User Interface**

The user interface must be user-friendly to guarantee to the final user an easy way to interact with the application. The development of the front end is realized through HTML and CSS to provide a wide range of combability among devices.

MOCKUP HERE

**Software Interface**

The system will use the following programming, markup and representation languages:

* JavaScript
* HTML
* CSS
* Python
* SQL
* JSON

**Communication Interface**

Communication between client and server happens through HTTPS by TCP protocol using the standard port 443.

The system must preserve user’s privacy, all the communications involving sensitive data must be held in a crypted channel.

## **Functional Requirements**

## **Scenarios, use case diagrams and Flow Diagrams**

### Scenario 1