



# POLITECNICO MILANO 1863

## Software Engineering 2 Project

### **TrackMe**

### *Data4Help & AutomatedSOS*

## Requirement Analysis and Specification Document

*A.Y. 2018/2019*

*Giorgio Polla – 921260*

*version 1.0*

*Alberto Tiraboschi – 920775*

*11/11/2018*

# Table of Contents

<b>1 Introduction</b>	<b>5</b>
1.1 Purpose	5
1.2 Scope	5
1.2.1 Description of the problem	5
1.2.2 Current system	6
1.2.3 Goals	6
1.3 Definitions, acronyms and abbreviations	7
1.3.1 Definitions	7
1.3.2 Acronyms	7
1.3.3 Abbreviations	8
1.4 Revision history	8
1.5 Reference documents	8
1.6 Document structure	9
<b>2 Overall Description</b>	<b>10</b>
2.1 Product perspective	10
2.1.1 Class Diagram	10
2.1.2 State Diagram	11
2.2 Product functions	11
2.2.1 Ambulance management	11
2.2.2 Health parameters' threshold tuning	11
2.3 User characteristics	12
2.3.1 Actors	12
2.4 Assumptions, dependencies and constraints	12
2.4.1 Assumptions and dependencies	12
2.4.2 Constraints	14
<b>3 Specific requirements</b>	<b>15</b>
3.1 External interface requirements	15
3.1.1 User interfaces	15

3.1.2 Hardware interfaces	17
3.1.3 Software interfaces	17
3.1.4 Communication interfaces	17
3.2 Functional requirements	18
3.2.1 Use case diagrams	18
3.2.2 Scenarios	20
3.2.3 Use cases	22
3.2.4 Requirements specification	31
3.2.5 Sequence diagrams	34
3.3 Performance requirements	37
3.4 Design constraints	37
3.4.1 Standard compliance	37
3.4.2 Hardware limitations	37
3.4.3 Any other constraint	37
3.5 Software system attributes	38
3.5.1 Reliability	38
3.5.2 Availability	38
3.5.3 Security	38
3.5.4 Maintainability	38
3.5.5 Portability	38
<b>4 Formal analysis using Alloy</b>	<b>39</b>
4.1 Introduction	39
4.2 Simplifications	39
4.3 Signatures	40
4.4 Facts	41
4.5 Assertions and predicates	43
4.6 Results	43
4.6.1 Executed commands	43
4.6.2 Execution results	44
4.6.3 Generated world	44
<b>5 Appendix</b>	<b>45</b>

5.1 Used tools	45
5.2 Hours of work	45
5.2.1 Alberto Tiraboschi	45
5.2.2 Giorgio Polla	46

# 1. Introduction

## 1.1 Purpose

This is the Requirement Analysis and Specification Document (RASD). As defined by the IEEE Standards, the goals of this document are to give a complete description of the system-to-be, in terms of both functional and non-functional requirements, to analyse the needs of the costumer in order to model the system and to show the constraints the system itself will be subjected to.

## 1.2 Scope

### 1.2.1 *Definition of the problem*

#### 1.2.1.1 *Data4Help*

**TrackMe** is a company that wants to develop a software-based service allowing third parties to monitor the location and health status of individuals. This service is called **Data4Help**. The service supports the registration of individuals who, by registering, agree that TrackMe acquires their data (data acquisition can happen through smartwatches or similar devices). Also, it supports the registration of third parties. After registration, these third parties can request:

- Access to the data of some specific individuals (we can assume, for instance, that they know an individual by his/her social security number or fiscal code in Italy). In this case, TrackMe passes the request to the specific individuals who can accept or refuse it.
- Access to anonymized data of groups of individuals (for instance, all those living in a certain geographical area, all those of a specific age range, etc.). These requests are handled directly by TrackMe, that will accept any request for which the number of individuals whose data satisfy the request is higher than 1000, in order to protect the identity of its clients.

As soon as a request for data is approved, TrackMe makes the previously saved data available to the third party. Also, it allows the third party to subscribe to new data and to receive them as soon as they are produced.

### *1.2.1.2 AutomatedSOS*

TrackMe decides to build a new service, called **AutomatedSOS**, on top of Data4Help. AutomatedSOS monitors the health status of the subscribed customers and, when such parameters are below certain thresholds, sends to the location of the customer an ambulance, guaranteeing a reaction time of less than 5 seconds from the time the parameters are below the threshold.

### *1.2.2 Current System*

TrackMe has no pre-existing services that need to be integrated with the new ones, Data4Help and AutomatedSOS. The whole system-to-be is going to be build completely from scratch, therefore no current system will be considered during the design process.

### *1.2.3 Goals*

#### *1.2.3.1 Data4Help*

- [G1]: Allow an individual to become a Registered User, by providing credentials and agreeing to TrackMe's privacy policy.
- [G2]: Allow a third-party company to become a Registered Third Party by providing credentials.
- [G3]: Allow a Registered Third Party to have an access to a specific, properly identified Registered User's data, if and only if the said User gives his/hers consent to that action.
- [G4]: Allow a Registered Third Party to access anonymized data of groups of individuals, if and only if the said group comprehends at least 1000 individuals.
- [G5]: By previous approval by the use send to a Registered Third Party with a subscription to new data the new version of the saved data, as soon as this version is produced.

### 1.2.3.2 AutomatedSOS

- [G6]: Allow a Registered User to become a Registered AutomatedSOS User
- [G7]: Send an ambulance to a Registered AutomatedSOS User's location, if and only if his/hers health parameters drop below a certain threshold.

## 1.3 Definitions, acronyms and abbreviations

### 1.3.1 Definitions

- **Registered User**: an individual registered to the Data4Help service, and that has therefore given consent to TrackMe to acquire their data.
- *[Registered User's]* **Data**: information about a RU's location and health status. As no more precise information are reported in the problem description, further information regarding the definition of "health status" are reported in section 2.4.1.1.
- **Current Saved Data**: it's the data currently saved in the Data4Help system.
- **Registered Third Party**: a third-party company, independent from TrackMe, that registered itself to the Data4Help service.
- *[Health parameters]* **threshold**: a precisely defined threshold of the health parameters: if these parameters descend below the said threshold, an individual is considered to be in danger. As no more precise details are reported in the problem description, further information regarding the definition of "health parameters' threshold" is reported in section 2.4.1.1.
- **Consenting Registered User** *[w.r.t. a request from a RTP]*: a RU that has accepted a request for his/hers data by a RTP.

### 1.3.2 Acronyms

- **RU**: Registered User
- **CRU**: Consenting Registered User
- **RAU**: Registered AutomatedSOS User

- **RTP**: Registered Third Party
- **RASD**: Requirement Analysis and Specification Document

### 1.3.3 Abbreviations

- **[Gn]**: the goal number  $n$ .
- **[Dn]**: the domain assumption number  $n$ .
- **[Rn]**: the functional requirement number  $n$ .

## 1.4 Revision history

At the moment there is no revision history to show, as this current version is the only existent one.

## 1.5 Reference documents

- Specification document: “Mandatory Project Assignment AY 2018-2019”
- IEEE Std 830-1993 - IEEE Guide to Software Requirements Specifications
- IEEE Std 830-1998 - IEEE Recommended Practice for Software Requirements Specifications



## 1.6 Document structure

This document is composed by five main parts, including a final appendix.

- I. **Introduction.** In the first part the problem is presented, with all the main basic information needed to fully understand the scope and the objective of the document itself. All the goals of the system-to-be are already specified and explained.
- II. **Overall Description.** The second part is a comprehensive general description of the system: there is the presentation of its boundaries and the specification the main actors that will use the system itself.
- III. **Specific Requirements.** The third part is a precise specification of the aspects already discussed in the previous section, in order to clarify the details for the development team. There is here an identification of every specific requirement, functional or non-functional. Furthermore, here is the use case diagrams, use cases, scenarios, performance analysis, design constraints and finally the attributes of the software system.
- IV. **Analysis of the System.** The fourth part provides a more formal analysis of the system, using a formal approach (using Alloy). This more rigorous approach is used to define the critical aspects of the system.
- V. **Appendix.** The fifth and last part comprehends a list the tools used to redact this document and its contents and a detailed report of the hours and efforts spent by each member of the group during the project.

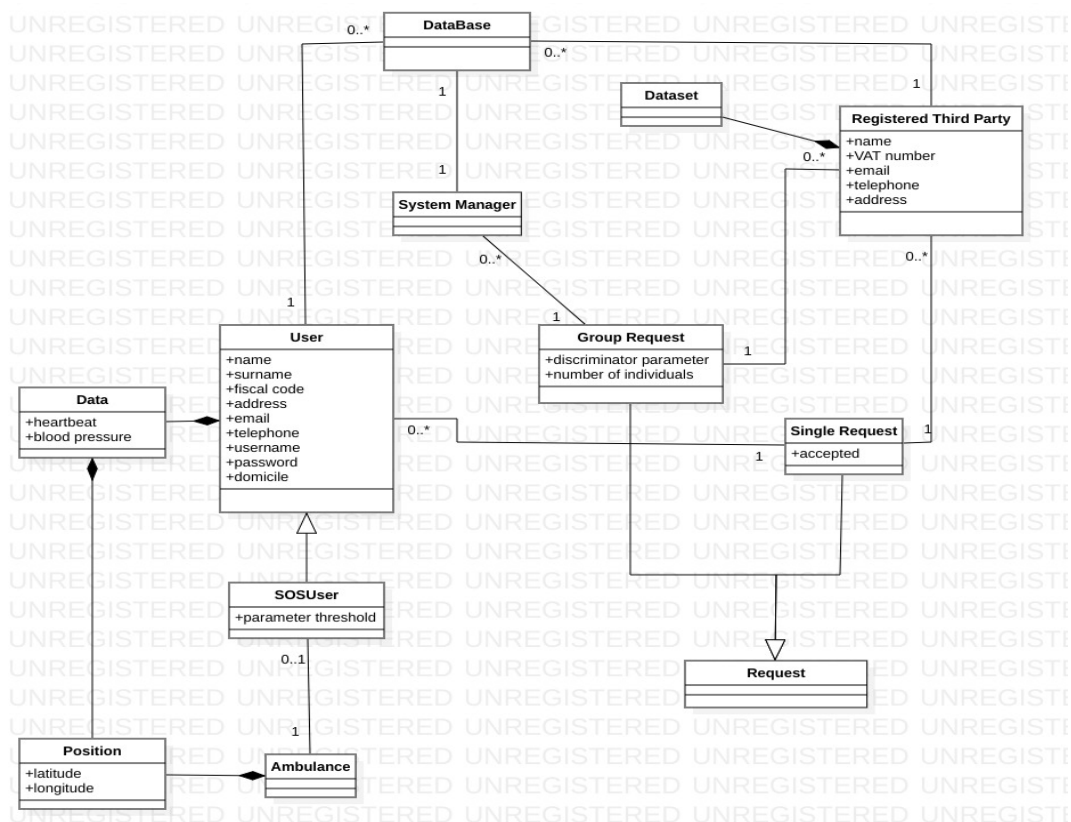
## 2. Overall Description

### 2.1 Product perspective

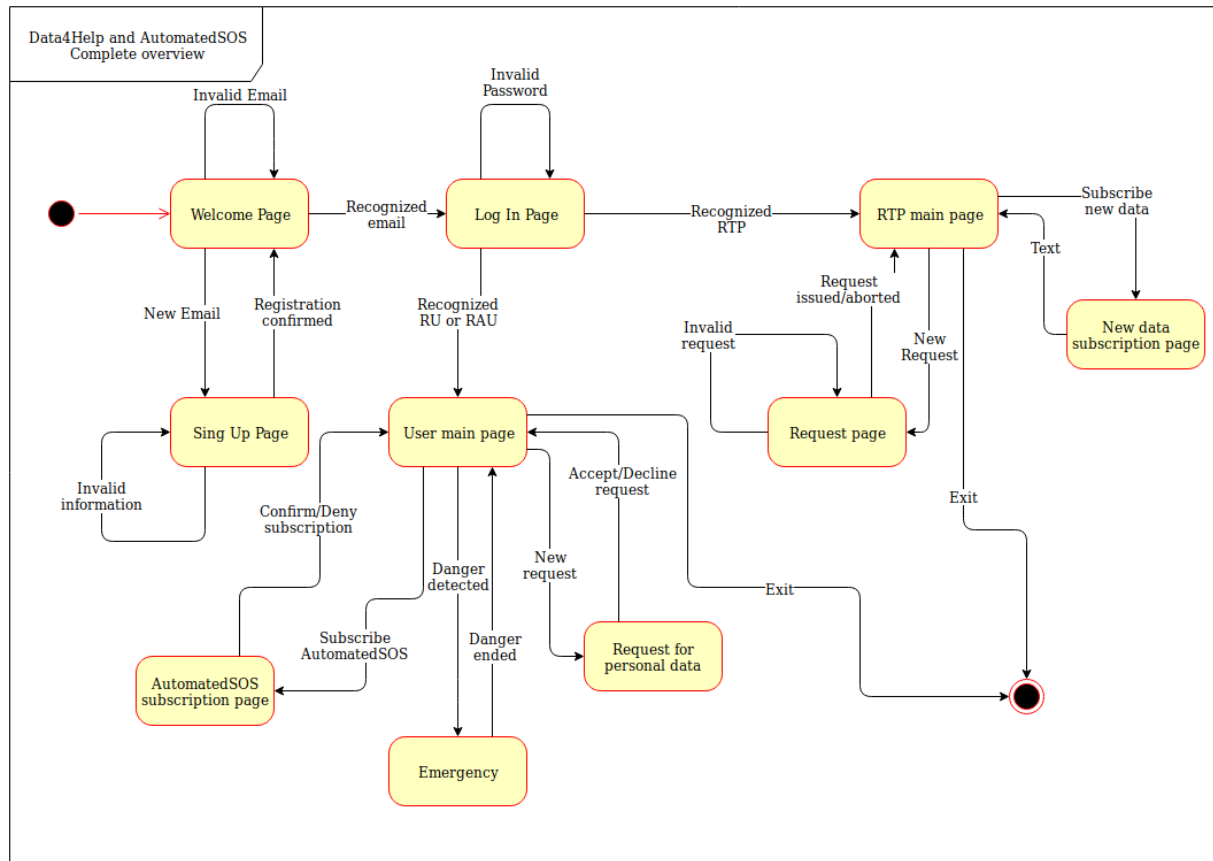
The whole system is going to be developed from scratch, since there are not any pre-existing systems to integrate with. Referring to the theory of the World and the Machine, by Jackson & Zave, some of the most interesting shared phenomena considered in this project are:

- the sending of an ambulance, and the communication with it (machine controlled)
- the user's data retrieval (world controlled)
- the action of registering or subscribing, from individuals and companies (world controlled)
- the action of requesting data, from companies (world controlled)
- the action of accessing data, from companies (machine controlled)
- the action of requesting data, from companies (world controlled)

#### 2.1.1 Class Diagram



## 2.1.2 State Diagram



## 2.2 Product functions

### 2.2.1 Ambulances management

The ambulances previously mentioned in the description of the AutomatedSOS service are not logistically related in any way to the new system, with the only exception being the dispatching of an ambulance by the system itself. Every other action related to the ambulance life cycle, such as for example the maintenance routine, is not supported in any way by the system.

### 2.2.2 Health parameters' threshold tuning

The health parameters' threshold mentioned in the description of the AutomatedSOS service may need to be tuned to the specific RAU's conditions, w.r.t TrackMe's policies, as reported in section 2.4.1.1.

At the first release, the system is not supporting this process of tuning in any way, and relies on further specification from TrackMe in order to effectively retrieve these thresholds.

## 2.3 User characteristics

### 2.3.1 Actors

- **Visitor:** an individual that is exploring TrackMe's services without being registered. The only thing he/she can do is to proceed with him/her registration.
- **Registered User:** often referred to as simply “User”, a RE is an individual who has successfully registered him/herself to the Data4Help service, and therefore can log in to the system. His/her data will be sent to TrackMe.
- **AutomatedSOS Registered User:** a RE that also subscribed to the AutomatedSOS service, on top of Data4help, and that can therefore utilise all the AutomatedSOS features.
- **Visiting Third Party:** a third-party company that is exploring TrackMe's services without being registered. The only thing he/she can do is to proceed with him/her registration.
- **Registered Third Party:** a third-party company, independent from TrackMe, that registered itself to the Data4Help service. It can ask for a specific individual's data, or for anonymized data from a batch of individuals.

## 2.4 Assumptions, dependencies and constraints

### 2.4.1 Assumptions and dependencies

In order to clarify some points that lacked precision in the specification document are hereby introduced the following assumptions.

#### 2.4.1.1 Text Assumptions

- In order to become a Registered User, an individual has to provide the following information while registering: name, surname, address, email, telephone number and a unique form of identification.
- The unique form of identification needed for individuals is here considered to be the individual's fiscal code.
- In order to become a Registered Third Party, a company has to provide the following information while registering: name, address, telephone number, email and a unique form of identification.
- The unique form of identification needed for third party companies is here considered to be its VAT number.
- The “health parameters” previously mentioned are considered to be the following: heartbeat rate, blood pressure, respiratory rate.
- The “health parameters' threshold” previously mentioned are considered to be precisely defined values of said parameters, tuned by TrackMe itself. Since no further information is reported and since the system isn't affected in any way by this lack of information, it is considered here that a set of thresholds is associated to each RAU, without specifying in the details how these thresholds are calculated.
- It is never mentioned in the specification document if and how any of the services described should have a subscription fee or a cost of any kind. The only small reference to the economic aspect is in the “Track4Run” service, here not considered, where that service itself is regarded as a potential “*great source of revenues*”: anyway, it is still never specified if and how that revenues should be acquired and if and how any direct cash flow towards TrackMe should be considered in the scope of the project (subscriptions, paying-on-request, advertisement, etc.). Due to the complete lack of information, it is assumed that any economic implication is out of the scope of this analysis.

#### 2.4.1.2 Domain Assumptions

- [D1]: Users' locations are correctly retrieved by GPS.
- [D2]: Users' health parameters retrieved by smartwatches or similar devices are the actual health parameters of the user.
- [D3]: The user's unique form of identification, here considered to be the individual's fiscal code, are actually unique for each person.

- [D4] The company's unique form of identification, here considered to be its VAT number, is actually unique for each company.
- [D5]: An ambulance will always be available to assist any RAU, at any time.
- [D6]: The user's current location will always be reachable by the ambulance.
- [D7]: When the order to dispatch an ambulance is correctly sent, then the said ambulance will surely arrive to the target location.
- [D8]: The threshold parameters are always correctly tuned, so that there will never be a false alarm.

## *2.4.2 Constraints*

### *2.4.2.1 Regulatory Policies*

Data4Help will ask for users' permission to access to their location and health status information, in order to acquire, store and eventually transmit their data to third-party companies. As mentioned before, Data4Help grants the user's anonymity when transmitting his/her data as part of a bigger batch of individuals, and directly asks the user if a Registered Third Party is interested in his/her data in particular, in order to acquire the individual's consent.

### *2.4.2.2 Interface to other applications*

There will be no public interfaces, therefore third party services won't be able to interoperate with neither Data4Help or AutomatedSOS.

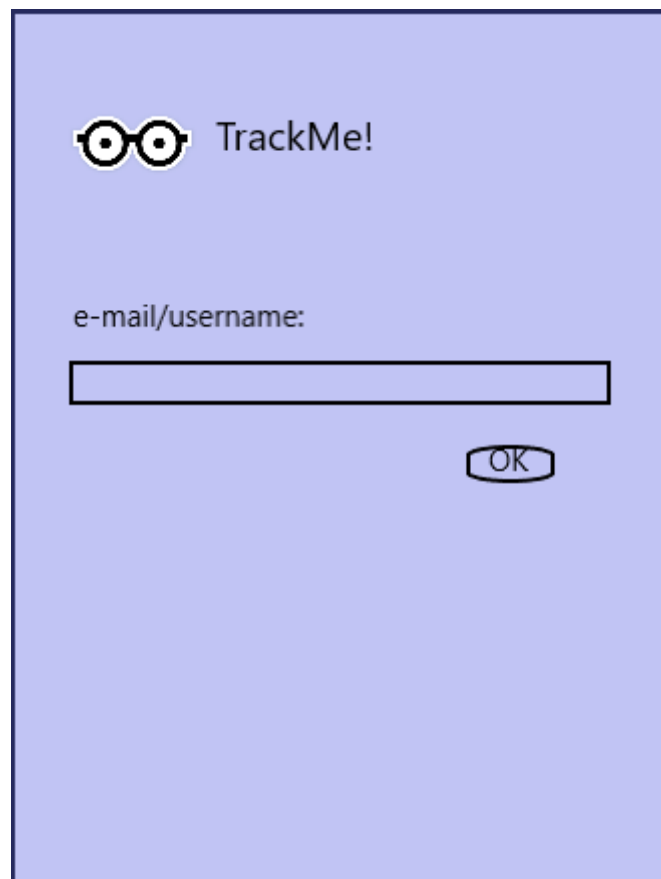
## 3. Specific Requirements

### 3.1 External Interface Requirements

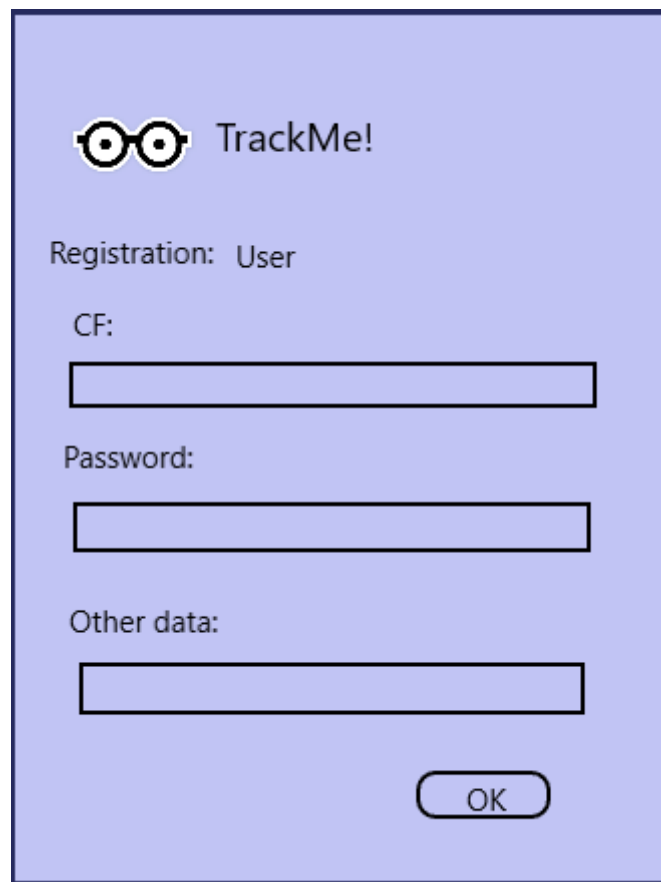
In this section we will specify how the modules of this system will interact the one to each other.

#### 3.1.1 User Interfaces

Here are reported some mock-ups of a possible user interface. Being just temporary drawings, these figures may be very different from the final one found in the first release of the system, and their objective is just to give an idea about how the user interface could appear: therefore, the figures are in any way accurate or descriptive enough to rely only on them in order to understand the functions of the system. A better and more accurate version will be presented in the Design Document.



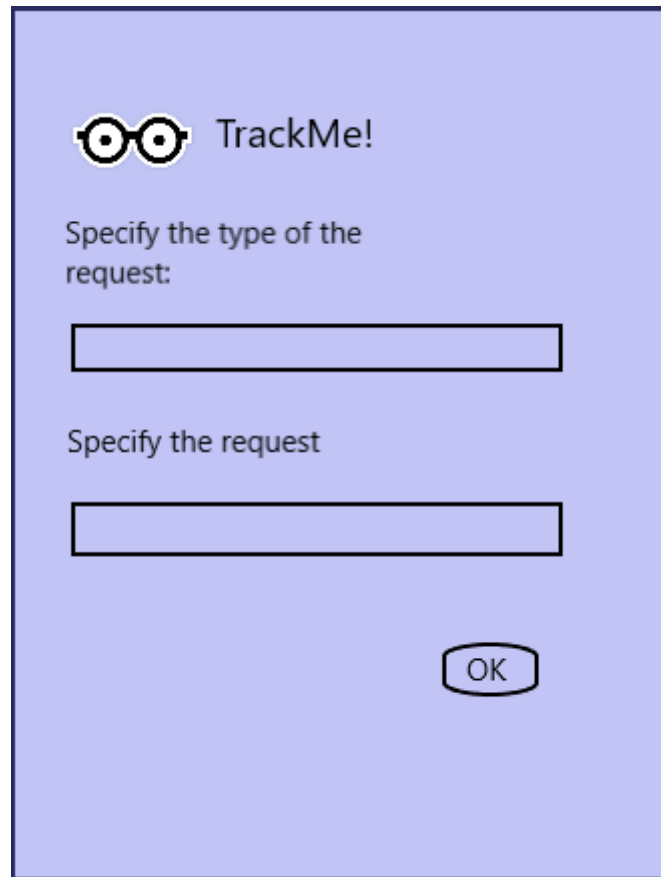
*Illustration 1: Welcome page. In the last version, only the email is here required.*



The image shows a registration form for 'TrackMe!'. At the top left is a logo consisting of a pair of glasses. To the right of the logo is the text 'TrackMe!'. Below this, the text 'Registration: User' is displayed. There are three input fields: the first is labeled 'CF:', the second is labeled 'Password:', and the third is labeled 'Other data:'. Each label is followed by a rectangular input box. At the bottom right of the form is a rounded rectangular button labeled 'OK'.

*Illustration 2: Registration page. In the last version there are more fields like the one present here, in order to acquire all the information needed. Also, there will be the choice to register as a RU or RTP.*





*Illustration 3: Request page. In the final version there will be an option to select which kind of request to issue, and, basing on that choice, all the fields needed in order to file that request.*

### 3.1.2 Hardware Interfaces

The system must be able to communicate with a smartwatch or any other similar devices, as previously described, in order to acquire the user's health parameters: the only hardware interfaces needed are the one used to communicate with the said devices. Since the GPS service is also needed, it is required to utilise a smartphone that can offer that functionality.

### 3.1.3 Software Interfaces

The system is not using any external service, and therefore it doesn't need any direct software interface. In this project the communication with the ambulances does not require a specific software interface.

### 3.1.4 Communication Interfaces

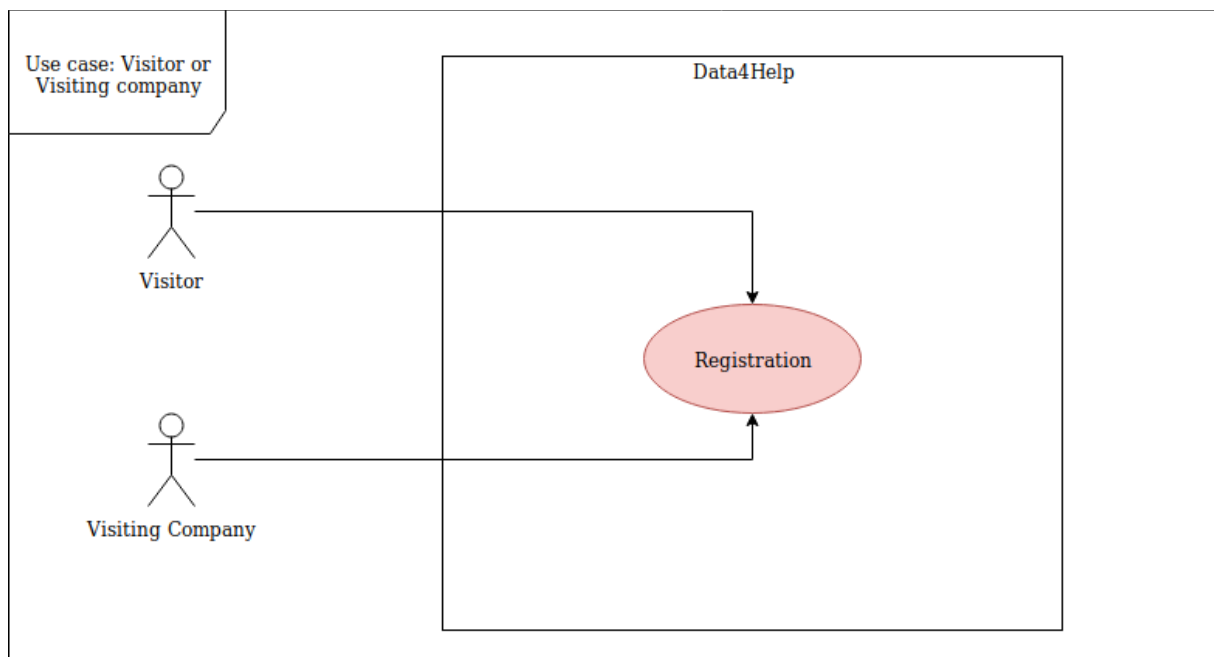
The system utilises Internet to transmit data, for every single action in the network. Therefore, interfaces are needed in order to exploit that connectivity, both in iOS and Android environments.

## 3.2 Functional Requirements

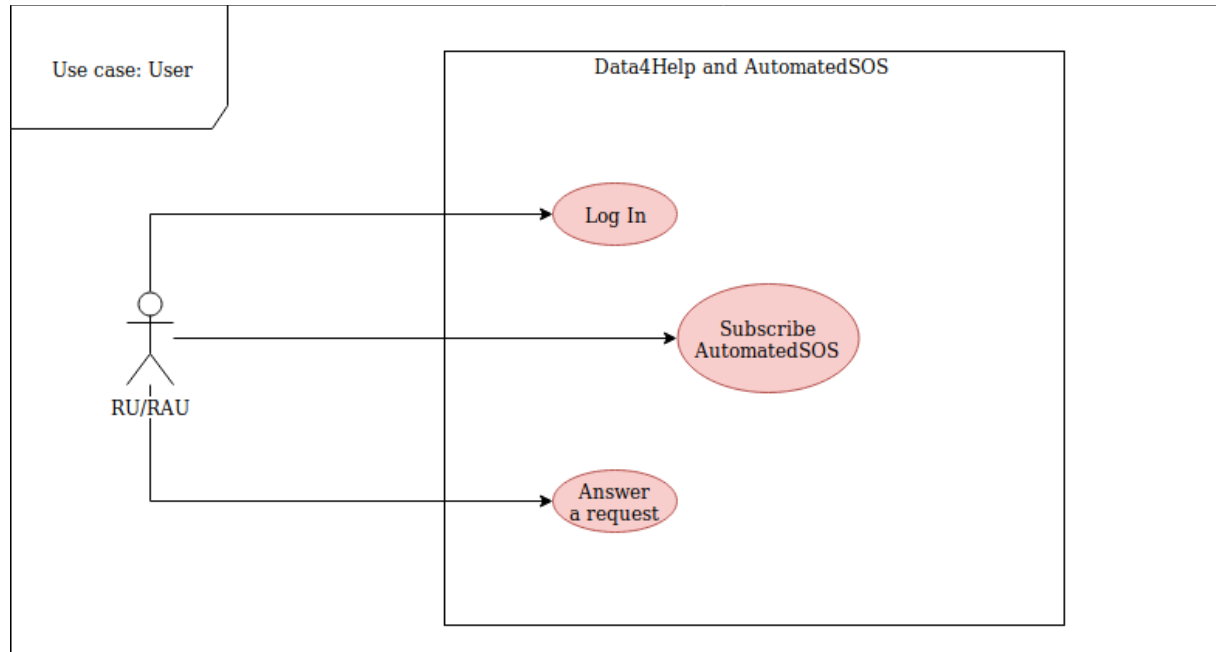
In this section will be displayed the use case diagrams, use cases and the full state diagrams, the Goals of the system, and the mapping of the requirements.

### 3.2.1 Use case diagrams

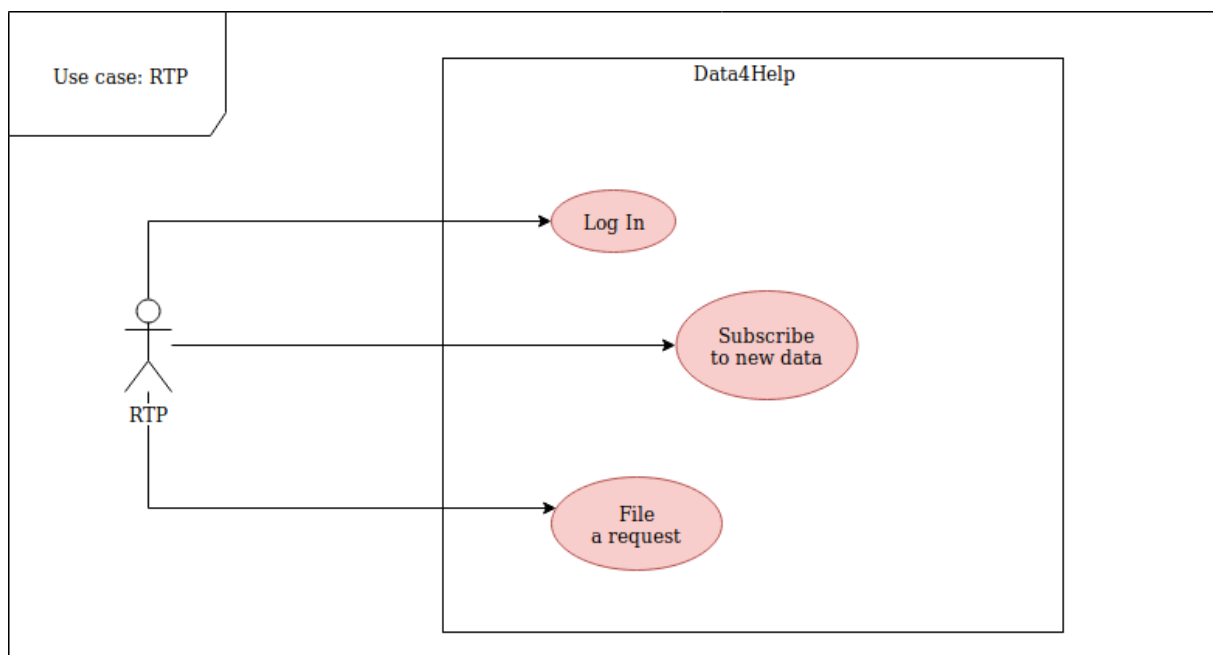
#### 3.2.1.1 Visitor and Visiting Company



### 3.2.1.2 RU and RAU



### 3.2.1.3 RTP



## 3.2.2 Scenarios

### 3.2.2.1 Scenario 1

Alice sees this new app called Data4Help, and she thinks she can give it a try. So she suggests to her boyfriend Bob, who always wears a smartwatch, to sign up to the system. Bob provides the requested data and agrees to the privacy policy. After registration he logs in to the platform.

### 3.2.2.2 Scenario 2

*TrackingBusiness* is a newly started business, which acquires data from the users and shares them. Thomas is the head of this company and thinks it might be a good idea to join the platform Data4Help. So he download the app, and then register his company. The idea of registering as a user, not only as a third party came across his mind, but he prefers to not be tracked. After the data are entered into the system, he can start by logging in the platform.

### 3.2.2.3 Scenario 3

The business of Thomas is now connected to the network of TrackMe. He decides then to make his first request. This request is of a specific user, a friend of him called Mike, who is registered to the system and whose fiscal code is known by Thomas. The system receives that request and subsequently asks the appropriate user, Mike himself, to choose if he will allow the business to get his data. Mike accepts that request and now his data can be accessed by Thomas' company.

### 3.2.2.4 Scenario 4

Thomas now wants to file a request for another friend of his, Miguel, who is registered and whose fiscal code is once again known by Thomas. The request is issued, but Miguel, not recognising that the company asking for his data is Thomas' one, denies it. His data cannot be directly accessed by *TrackingBusiness*.

### 3.2.2.5 Scenario 5

Now Thomas wants to try the general anonymized data retrieval feature: he files a request looking for 22-year-old males living in Via Giuseppe Tartini in Milan. The system receives the request, but since the number of users satisfying the request itself are not at least 1000, the request is declined. Thomas receives a notification saying that unfortunately the system wasn't able to provide the anonymized data that he needed.

### *3.2.2.6 Scenario 6*

Eager to try every functionality the system has to offer, Thomas now decides to try the subscription to new data. He subscribes his company, and happily goes on with his life. After a few hours, he receives a notification: new data is available regarding the information about his friend Mike! Thomas can now see that Mike was registered less 10 minutes ago at Esselunga di via Feltre.

### *3.2.2.6 Scenario 7*

Bob is an old user, and has recently subscribed to the AutomatedSOS service. Out of the blue he doesn't feel very well, his heartbeat is becoming low, and the pressure drops in free fall. But since Bob is provident, the smartwatch he always wears detects the imminent problem and then the AutomatedSOS system detects an anomaly. The system then promptly sends an ambulance to Bob's position to rescue him, probably saving his life.

### 3.2.3 Use cases

#### 3.2.3.1 Registration of an individual

ACTORS	Visitor
GOALS	[G1]
INPUT CONDITIONS	There are no entry conditions.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. On the welcome page, the Visitor inserts his e-mail and presses the 'Enter' button.</li><li>2. The system recognised an email not associated to any RU or RTP and redirects the Visitor to the 'Sign Up' page.</li><li>3. The Visitor fills every mandatory field in this page providing his information, specifying he wants to become a Registered User.</li><li>4. The Visitor clicks on the 'Confirm' button.</li><li>5. The systems saves the data and sends a confirmation email to the Visitor.</li><li>6. The Visitor clicks on the link in the confirmation email and completes his registration.</li></ol>
OUTPUT CONDITIONS	The Visitor has completed the registration and can now log in to the service.
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The Visitor has already registered himself.</li><li>2. The Visitor inserts invalid data in some mandatory fields.</li><li>3. The visitor chooses a fiscal code already associated to another user.</li></ol> <p>Exception 1 redirects the Visitor to the wrong page: since this exception cannot be effectively handled, the Visitor will have to return back from the 'Log In' page.</p> <p>All other exceptions are handled by looping the insert request until correct data is provided (returning to point 3) and by notifying the Visitor.</p>

### 3.2.3.2 User login

ACTORS	User
GOALS	[G1]
INPUT CONDITIONS	There are no entry conditions.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. On the welcome page, the Visitor inserts his e-mail and presses the 'Enter' button.</li><li>2. The system recognised an email associated to a user and redirects the User to the 'Log In' page.</li><li>3. The User insert his password in the correct field.</li><li>4. The User clicks on the 'Log In' button.</li></ol>
OUTPUT CONDITIONS	The User has successfully logged in.
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The User inserts an invalid email.</li><li>2. The User inserts an invalid password.</li></ol> <p>Exception 1 redirects the User to the wrong page: since this exception cannot be effectively handled, the User will have to return back from the 'Sign Up' page.</p> <p>Exception 2 is handled by looping the insert request until the correct password is provided (returning to point 3) and by notifying the User.</p>

### 3.2.3.3 Registration of a third-party company

ACTORS	Visiting company
GOALS	[G2]
INPUT CONDITIONS	There are no entry conditions.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. On the welcome page, the Visiting Company inserts its e-mail and presses the 'Enter' button.</li><li>2. The system recognised an email not associated to any RU or RTP and redirects the Visiting Company to the 'Sign Up' page.</li><li>3. The Visiting Company fills every mandatory field in this page providing its information and specifying it wants to become a RTP.</li><li>4. The Visiting Company clicks on the 'Confirm' button.</li><li>5. The systems saves the data and sends a confirmation email to the Visiting Company.</li><li>6. The Visiting Company clicks on the link in the confirmation email and completes its registration.</li></ol>
OUTPUT CONDITIONS	The Visiting Company has completed the registration and can now log in to the service.
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The Visiting Company has already registered himself.</li><li>2. The Visiting Company inserts invalid data in some mandatory fields.</li><li>3. The visiting Company chooses a VAT already associated to another RTP.</li></ol> <p>Exception 1 redirects the Visiting Company to the wrong page: since this exception cannot be effectively handled, the Visiting Company will have to return back from the 'Log In' page.</p> <p>All other exceptions are handled by looping the insert request until correct data is provided (returning to point 3) and by notifying the Visiting Company.</p>



#### 3.2.3.4 RTP Login

ACTORS	Registered Third Party
GOALS	[G2]
INPUT CONDITIONS	There are no entry conditions.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. On the welcome page, the RTP inserts its e-mail and presses the 'Enter' button.</li><li>2. The system recognised an email associated to a RTP and redirects the RTP to the 'Log In' page.</li><li>3. The RTP insert his password in the correct field.</li><li>4. The RTP clicks on the 'Log In' button.</li></ol>
OUTPUT CONDITIONS	The RTP has successfully logged in.
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The RTP inserts an invalid email.</li><li>2. The RTP inserts an invalid password.</li></ol> <p>Exception 1 redirects the RTP to the wrong page: since this exception cannot be effectively handled, the RTP will have to return back from the 'Sign Up' page.</p> <p>Exception 2 is handled by looping the insert request until the correct password is provided (returning to point 3) and by notifying the RTP.</p>

#### 3.2.3.5 Sending a request for a specific user's data

ACTORS	RTP
GOALS	[G3]
INPUT CONDITIONS	The RTP has to be logged in to the system.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The RTP starts the request by clicking on the "Request" button.</li><li>2. The RTP inserts the fiscal code of the specific RU it wants to track.</li><li>3. The RTP clicks on the 'Send Request' button.</li><li>4. The system processes the request.</li></ol>
OUTPUT CONDITIONS	The RTP has sent a request for a specific user's data.
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The RTP inserts an invalid fiscal code.</li><li>2. The RTP inserts the fiscal code of a RU to whom it had already issued a request.</li></ol> <p>All exceptions are handled by looping the insertion of the fiscal code (returning to point 2) and by notifying the RTP.</p>

### 3.2.3.6 *Accepting a request for a specific user's data*

ACTORS	Registered User, RTP
GOALS	[G3]
INPUT CONDITIONS	The RU has to be logged in to the system. A RTP has sent a request to the RU for his private data.
EVENTS FLOW	1. The system notifies the RU about the request, asking him if he wants to share his data or not. 2. The RU accepts that request. 3. The system notifies the RTP that the request has been accepted.
OUTPUT CONDITIONS	The RTP can now access a copy of the specific user's data.
EXCEPTIONS	

### 3.2.3.7 *Declining a request for a specific user's data*

ACTORS	Registered User, RTP
GOALS	[G3]
INPUT CONDITIONS	The RU has to be logged in to the system. A RTP has sent a request to the RU for his private data.
EVENTS FLOW	1. The system notifies the RU about the request, asking him if he wants to share his data or not. 2. The RU declines that request. 3. The system notifies the RTP that the request has been declined.
OUTPUT CONDITIONS	The RTP cannot access the specific user's data.
EXCEPTIONS	

### 3.2.3.8 *Sending a successful general request for users' data*

ACTORS	RTP
GOALS	[G4]
INPUT CONDITIONS	The RTP has to be logged in to the system.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The RTP starts the request by clicking on the “Request” button.</li><li>2. The RTP inserts the selects the parameters used to discriminate users and insert the corresponding desired values.</li><li>3. The RTP clicks on the “Send Request” button.</li><li>4. The system processes the request, verifying if the number of users satisfying the request is at least 1000.</li><li>5. Since that condition is verified, the system grants the RTP access to anonymized data of the RU that satisfy the request.</li></ol>
OUTPUT CONDITIONS	The RTP has access to a copy of the wanted anonymized users' data
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The RTP inserts invalid values in the parameter value fields.</li><li>2. The RTP inserts the same values that it inserted in a previous general request.</li></ol> <p>The exception is handled by looping the insertion of the values (returning to point 2) and by opportunely notifying the RTP.</p>

### 3.2.3.9 Sending an unsuccessful general request for users' data

ACTORS	RTP
GOALS	[G4]
INPUT CONDITIONS	The RTP has to be logged in to the system.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The RTP starts the request by clicking on the “Request” button.</li><li>2. The RTP inserts the selects the parameters used to discriminate users and insert the corresponding desired values.</li><li>3. The RTP clicks on the “Send Request” button.</li><li>4. The system processes the request, verifying if the number of users satisfying the request is at least 1000.</li><li>5. Since that condition is not verified, the system doesn't grant the RTP access to anonymized data of the RU that satisfy the request.</li></ol>
OUTPUT CONDITIONS	The RTP has not access to the wanted anonymized users' data
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The RTP inserts invalid values in the parameter value fields.</li><li>2. The RTP inserts the same values that it inserted in a previous general request.</li></ol> <p>The exception is handled by looping the insertion of the values (returning to point 2) and by notifying the RTP.</p>

### 3.2.3.10 Registering to new data

ACTORS	RTP
GOALS	[G4]
INPUT CONDITIONS	The RTP has to be logged in to the system.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The RTP starts the process by clicking on the “New Data” button.</li><li>2. The system asks for a confirmation that the RTP really wants to subscribe to new data.</li><li>3. The RTP clicks on the 'Continue' button.</li><li>4. The system sends a notification to the RTP, displaying the successful subscription to new data.</li></ol>
OUTPUT CONDITIONS	The RTP has subscribed to new data.
EXCEPTIONS	

### 3.2.3.11 Reception of new data

ACTORS	RTP
GOALS	[G4]
INPUT CONDITIONS	The RTP has to be logged in to the system. The RTP must have subscribed to new data. A new version of the RTP's data must be available.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The system sends a notification to the RTP, showing that a new version of its data is now available.</li><li>2. The system grants the RTP access to the new data.</li></ol>
OUTPUT CONDITIONS	The RTP has access to a copy of the new data.
EXCEPTIONS	

### 3.2.3.12 *Becoming an AutomatedSOS User*

ACTORS	RU
GOALS	[G6]
INPUT CONDITIONS	The RU has to be logged in to the system.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The RU starts the process by clicking on the “AutomatedSOS” button.</li><li>2. The system asks again the user if he wants to complete the upgrade.</li><li>3. The user clicks on the 'Continue' button.</li><li>4. The system saves the thresholds associated to the individual, respecting TrackMe's policies.</li><li>5. The system sends a notification to the RU, displaying the successful upgrade to the AutomatedSOS service.</li></ol>
OUTPUT CONDITIONS	The RU is now a RAU and can therefore utilise the AutomatedSOS service.
EXCEPTIONS	<ol style="list-style-type: none"><li>1. The RU is already a RAY</li></ol> <p>The exception is handled by aborting the process and notifying the user.</p>

### 3.2.3.13 *AutomatedSOS Emergency*

ACTORS	RU
GOALS	[G7]
INPUT CONDITIONS	The RAU has to be logged in to the system. The RAU's health parameters drop below their safety threshold.
EVENTS FLOW	<ol style="list-style-type: none"><li>1. The system sends a message to the ambulance system, containing the information regarding the RAU's health status and current position.</li><li>2. An ambulance is dispatched towards the RAU's position.</li></ol>
OUTPUT CONDITIONS	The ambulance is now going towards the RAU's position.
EXCEPTIONS	

### 3.2.4 Requirements specification

#### 3.2.4.1 [G1]

Allow an individual to become a Registered User, by agreeing to the TrackMe privacy policy and by providing credentials.

- [R1] A visitor must be able to register him/herself, by starting the registration process.
  - [R1.1] The system will ask to the user all the needed information during the registration process.
  - [R1.2] The system will ask the user to accept TrackMe's privacy policy.
- [R2] The system must check the unicity of the inserted email.
- [D3]: The users' unique form of identification, here considered to be the individual's fiscal code, is actually unique for each person.

#### 3.2.4.2 [G2]

Allow a third-party company to become a Registered Third Party by providing credentials.

- [R3] A third party company must be able to register itself, by starting the registration process.
  - [R3.1] The system will ask to the company all the needed information during the registration process.
  - [R3.2] The system will ask the company to accept TrackMe's privacy policy.
- [R2] The system must check the unicity of the inserted email.
- [D4] The company's unique form of identification, here considered to be its VAT number, is actually unique for each company.

#### 3.2.4.3 [G3]

Allow a Registered Third Party to have access to a specific, properly identified Registered User's data if and only if the said User gives its consent to that action.

- [R4] A Registered Third Party must be able to log in to the system using its credentials.
- [R5] A Registered User must be able to log in to the system using his/her credentials.
- [R6] A Registered Third Party must be able to send a request to a Registered User, identified by its unique form of identification

- [R7] A Registered User must be able to view every request sent to him/her by Registered Third Parties.
  - [R7.1] A Registered User must be able to accept or decline the said request.
- [R8] If and only if the Registered User accepted the request from a Registered Third Party, then the same Registered Third Party must be able to access the Registered User's data.

#### 3.2.4.4 [G4]

Allow a Registered Third Party to access anonymized data of groups of individuals, if and only if the said group comprehends at least 1000 individuals.

- [R4] A Registered Third Party must be able to log in to the system using its credentials.
- [R5] A Registered User must be able to log in to the system using his/her credentials.
- [R8] A Registered Third Party must be able to send a request to the system for access to anonymized data of groups of individuals.
  - [R8.1] The said groups must be identified by some common characteristics, that can be their age, location, genre, etc. by the system.
- [R9] The system must be able to assess if the group identified by the request sent by the Registered Third Party is composed by at least 1000 individuals.
- [R10] If and only if the group identified by the request sent by the Registered Third Party is composed by at least 1000 individuals, then the same Registered Third Party must be able to access the said group's anonymized data.

#### 3.2.4.5 [G5]

Allow a Registered Third Party to subscribe to new data, in order to receive the latest data as soon it is produced.

- [R4] A Registered Third Party must be able to log in to the system using its credentials.
- [R11] A Registered Third Party must be able to subscribe to the new data.
- [R12] The system must send the new data to every Registered Third Party subscribed to the said service, as soon as the new data is produced.



#### 3.2.4.6 [G6]

Allow a Registered User to become a Registered AutomatedSOS User

- [R4] A Registered User must be able to log in to the system using his/hers credentials.
- [R13] A Registered User must be able to subscribe to the AutomatedSOS service, by starting the subscription process.
- [R14] The system must recognise the user is a RAU.

#### 3.2.4.7 [G7]

Send an ambulance to a Registered AutomatedSOS User's location, if and only if his/hers health parameters drop below a certain threshold.

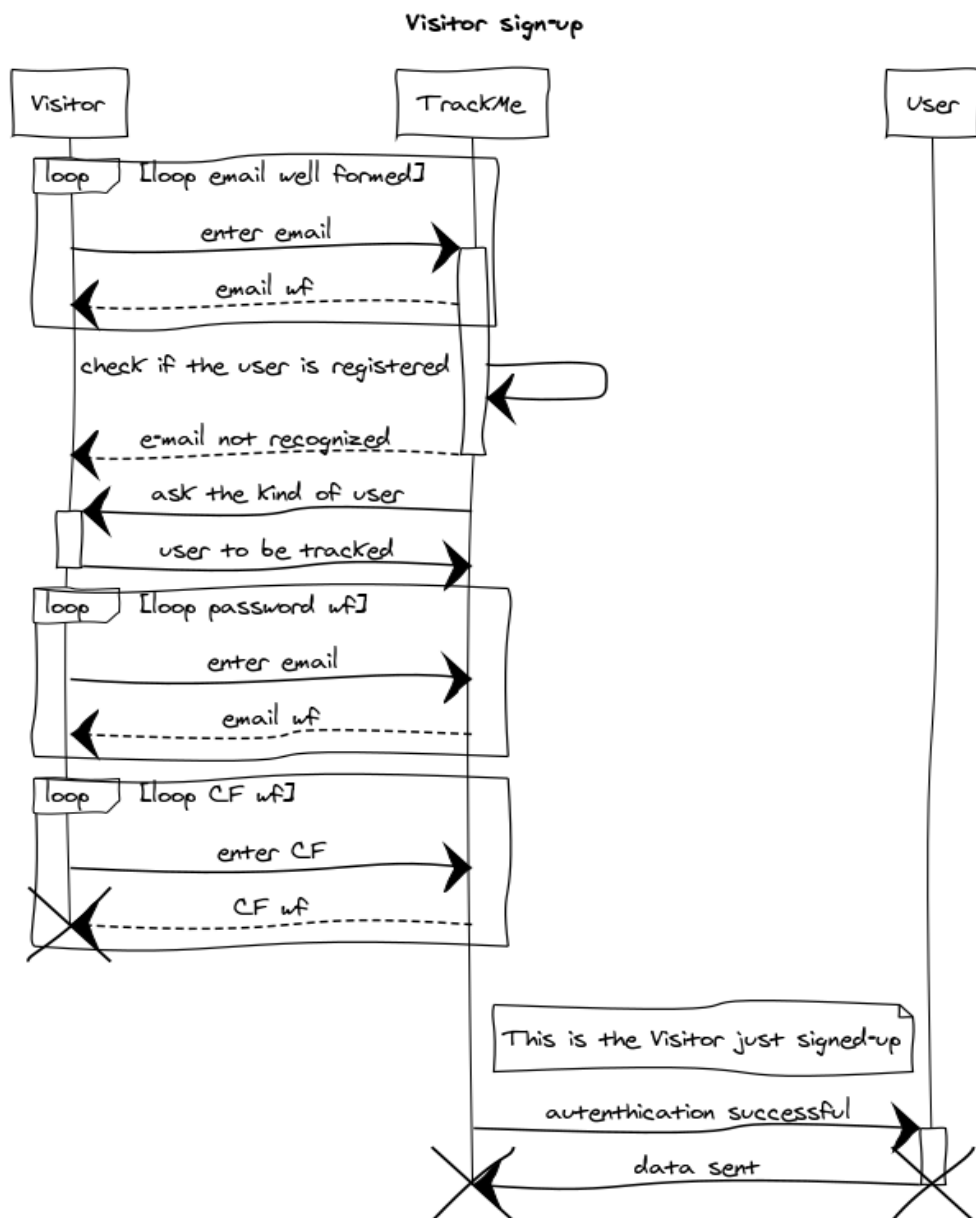
- [R5] A Registered User must be able to log in to the system using his/her credentials.
- [R14] The system must recognise the user is a RAU.
- [R15] The system must be able to detect when the RAU is in danger.
- [D2] Users' health parameters retrieved by smartwatches or similar devices are the actual health parameters of the user.
- [D8] The threshold parameters are always correctly tuned, so that there will never be a false alarm.
- [R16] The system must be able to notify an ambulance when there is an emergency.
- [R17] The system must be able to transmit to the ambulance the RAU's current position.
- [D1] Users' locations are correctly retrieved by GPS.
- [D5] An ambulance will always be available to assist any RAU, at any time.
- [D6] The user's current location will always be reachable by the ambulance.
- [D7] When the order to dispatch an ambulance is correctly sent, then the said ambulance will surely arrive to the target location.

### 3.2.5 Sequence diagram

In this section are reported some sequence diagrams, regarding some iconic and important processes by both individuals and companies.

#### 3.2.5.1 Visitor sign up

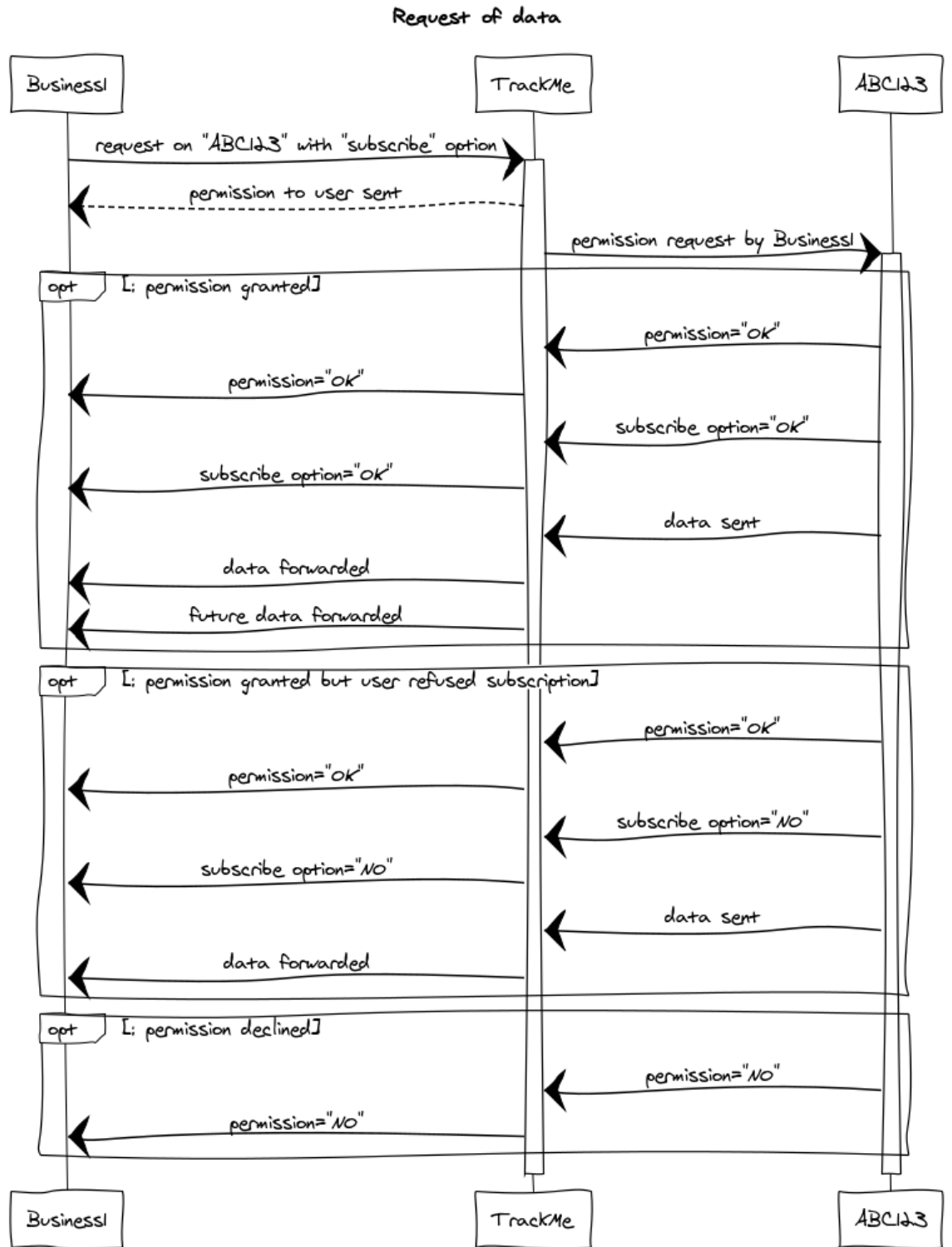
A new visitor decides to sign up into the system. With “TrackMe” is here intended the entire system, Data4Help and AutomatedSOS. Here the focus is only on the correctness of the information provided by the visitor.



www.websequencediagrams.com

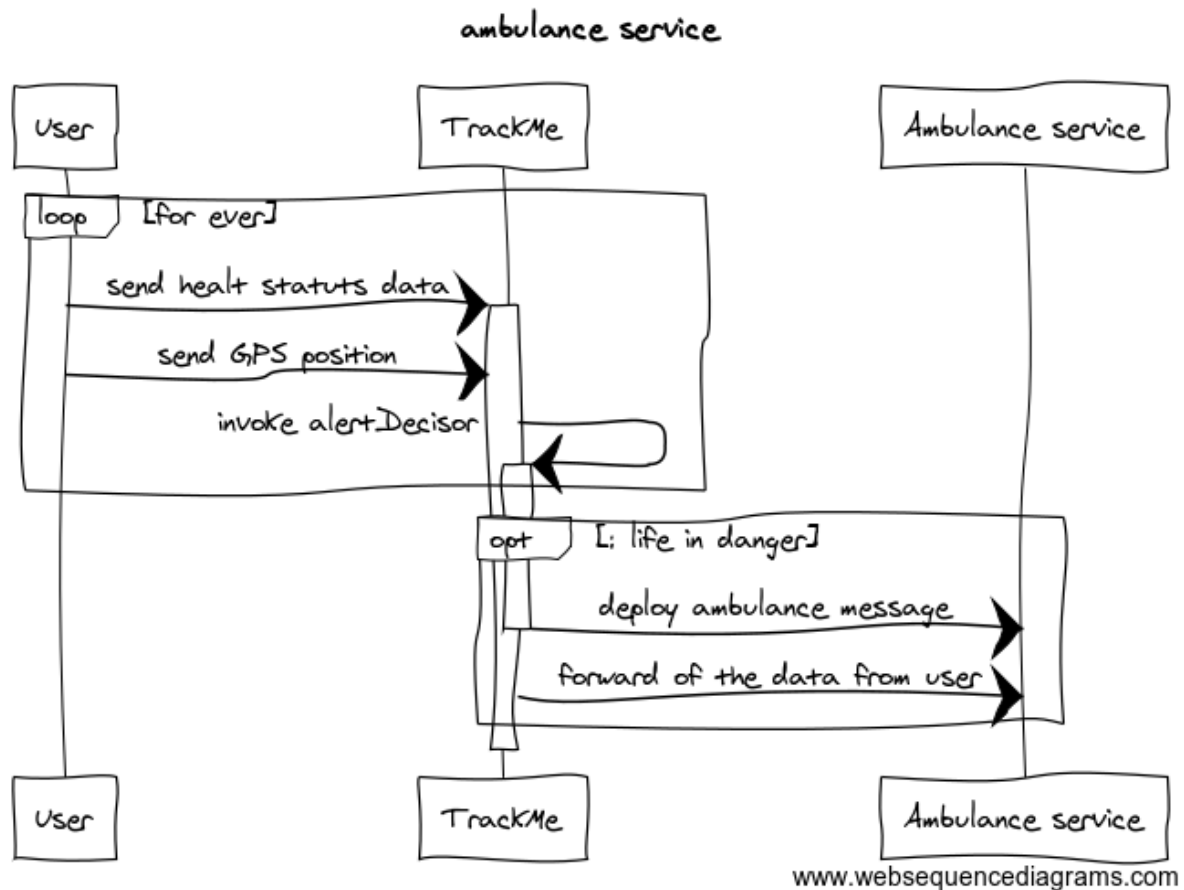
### 3.2.5.2 Request for a user's data.

A RTP files a request for a specific user's data. With "TrackMe" is here intended the entire system, Data4Help and AutomatedSOS, and with "Business" is intended a RTP , while "ABC123" is the fiscal code of a user, used here as an identifier. The focus is here on the request of a user's data by a RTP.



### 3.2.5.3 AutomatedSOS Service

A new visitor decides to sign up into the system. With “TrackMe” is here intended the entire system, Data4Help and AutomatedSOS. The focus is here on the constant monitoring of a user's health parameters, displayed in the loop, and the dispatching of an ambulance when needed.



### 3.3 Performance Requirements

At the first release, Data4Help is required to manage at least 20.000 Registered Users and 1.000 Registered Third Party. AutomatedSOS instead must have the capability to manage at least 1.000 Registered AutomatedSOS users.

The time between the detection that a user is in danger and the sending of an emergency message to the ambulance must be less than 5 seconds.

### 3.4 Software system attributes

#### 3.4.1 *Standards compliance*

The system must be able to communicate through Android and iOS devices, therefore it must respect all standards compliances these two environment require.

#### 3.4.2 *Hardware limitations*

- Android or iOS smartphone
- Any smartwatch-compatible device that can scan and transmit the individual's health status
- GPS
- 2G/3G/4G connection

#### 3.4.3 *Any other constraint*

- The time between the detection that a user is in danger and the sending of an emergency message to the ambulance must be less than 5 seconds.

## 3.5 Software system attributes

### 3.5.1 *Reliability*

The AutomatedSOS service must be available 24/7, with no exceptions, since people's health depends on that.

The Data4Help service, however, can be less strict and tolerate discrete concessions on the previous condition.

### 3.5.2 *Availability*

The AutomatedSOS service is expected to be available 99.99% of the time.

The Data4Help service, once again, can tolerate discrete concessions on the previous condition, descending to 99% of the time.

### 3.5.3 *Security*

For data-sensitive based services like the ones considered in this document, security is highly important. All sensitive information contained in the system, like users' data and RU and RTP's passwords must be confidentially stored and encrypted with high-security protection.

### 3.5.4 *Maintainability*

The application should be as easier as possible to maintain, in order to possibly add new features. This clarity is to be achieved by using common design patterns and a detailed documentation.

### 3.5.5 *Portability*

Since in this document it is considered a portable application, portability and compatibility with multiple devices are very important aspects. In particular, the Android version must be able to run on a very different devices.

## 4. Alloy Model

### 4.1 Introduction

In this section it's reported the formal model of the system, created and formalised in Alloy.

Hereby are listed and modelled only some sections of the system itself, in order to better specify some delicate details. Particular focus is on the following points:

- specific requests when accepted by the user;
- specific requests when declined by the user;
- general requests for groups of users;
- detection of health issues in RAUs and sending of an ambulance.

### 4.2 Simplifications

In order to simplify the model, but without losing any meaning, the following simplifications were adopted while developing the model itself:

- The only characteristic reported for the users, other than their location and health parameters, is their age. That said age is considered to be an integer ranging only from 0 to 2. Other factors, like their gender, domicile, etc. were omitted, but their behaviour is exactly the same as their age's.
- The users' position is composed by its geographical coordinates, latitude and longitude; anyway, these details are hereby overlooked, in order to simplify the
- Instead of 1000 individual, the threshold of anonymity is set to 2, in order to simplify the model without depriving it from any meaning. As previously mentioned in the first point, the only parameter considered, as an example, to define groups of users is their age.
- Every entity regarding RTP and RU's data is considered to belong in a central TrackMe's database: that database is then not further reported in the model and relations between RTP's datasets and RUs and RUs' data are directly reported without passing through the central database itself. It was chosen to represent the model this way in order to avoid heavy and continuous repetitions in these relationships, without depriving it from any meaning.
- As a consequence of the point before, it was chosen not to explicitly represent the “subscription to new data” mechanic. Because of its inherent simplicity, the focus was shifted towards more complex and interesting concepts, such as the ones reported in section 4.1.
- The only anonymous parameter that RTPs can request is the position.

## 8.3 Signatures

```
open util/boolean
open util/integer
-- A Registered User
sig RegUser{
    fiscalCode: one FiscalCode,
    age: one Int,
    position: one Position,
    heartbeat: one Int
}
-- Boundaries for age and heartbeat.
{ age >= 0 and age <= 4 and heartbeat > 0 }
-- A Registered Third Party .
sig RegThirdParty{
    vat: one VAT,
    requests: set Request,
    dataset: one Dataset
}
-- A RTP's dataset, containing info about the specific users (users) or anonymized data (anon).
sig Dataset{
    users: set RegUser,
    anon: set Position
}
-- In order to simplify the model, these entities are not further specified.
sig FiscalCode, VAT, Position{}

sig Ambulance{
    patient: one SOSUser,
    destination: one Position
}

abstract sig Request{}
-- A request for anonymized data, selecting users basing on their age.
sig GroupRequest extends Request{
    ageSelector: one Int
}
-- Boundaries for the age discriminator.
{ ageSelector >= 0 and ageSelector <= 2 }
-- A request for a specific user's data.
sig SingleRequest extends Request{
    userFC: one FiscalCode,
    accepted: one Bool
}
-- A Registered AutomatedSOS User.
sig SOSUser extends RegUser{
    threshold: one Int
}
{ threshold > 0 }
-- A collection of all the RAUs considered to be in danger.
lone sig InDanger{
    users: set SOSUser
}
```



## 8.4 Facts

```
-- A function that supports the inDatasetIffAnon fact. It returns the set of RU having age equal to a.
fun isItAnonymous [a: Int] : set RegUser{
    { u: RegUser | u.age = a}
}

-- Two different users cannot have the same fiscal code.
fact fiscalCodeIsUnique{
    no disjoint u1, u2: RegUser | u1.fiscalCode = u2.fiscalCode
}

-- Each fiscal code must be associated to a user.
fact noLoneFiscalCode{
    all f1: FiscalCode | some u1: RegUser | u1.fiscalCode = f1
}

-- Two different Registered Third Party cannot have the same VAT.
fact thirdPartyIdIsUnique{
    no disjoint t1, t2: RegThirdParty | t1.vat = t2.vat
}

-- Each VAT must be associated to a RTP.
fact noLoneVAT{
    all v1: VAT | some t1: RegThirdParty | t1.vat = v1
}

-- Each position must be associated to a user.
fact noLonePosition{
    all p1: Position | some u1: RegUser | u1.position = p1
}

-- A RTP cannot send more than one request to the same user.
fact oneSingleRequestPerUserPerRTP{
    (all t1: RegThirdParty |
        no disjoint r1, r2: SingleRequest |
            r1 in t1.requests and r2 in t1.requests and r1.userFC = r2.userFC) and
    (no disjoint t1, t2: RegThirdParty |
        some r: Request |
            r in t1.requests and r in t2.requests)
}

-- Two different RTP cannot have the same dataset.
fact datasetIsUnique {
    no disjoint t1, t2: RegThirdParty |
        t1.dataset = t2.dataset
}

-- Each dataset must be associated to a RTP.
fact noLoneDataset {
    all d1: Dataset | some t1: RegThirdParty | t1.dataset = d1
}
```

```

-- Each request must be associated to a RTP.
fact noLoneRequest {
    all r: Request | some t: RegThirdParty | r in t.requests
}

-- There cannot be two exactly same group requests from the same RTP.
fact noDoubleGroupRequests {
    all t: RegThirdParty |
        no disjoint gr1, gr2: GroupRequest |
            gr1 in t.requests and gr2 in t.requests and
            gr1.ageSelector = gr2.ageSelector
}

-- A user is present in a RTP's dataset if and only if the said RTP has requested that individual
-- for his/hers data and he/she has given consent.
fact inDatasetIffConsenting{
    all u: RegUser, t: RegThirdParty |
        u in t.dataset.users iff (some sr: SingleRequest |
            sr in t.requests and sr.userFC = u.fiscalCode and sr.accepted = True)
}

-- Anonymized users' data is present in a RTP's dataset if and only if the said RTP has
-- issued a group request that grants users' anonymity.
fact inDatasetIffAnon{
    all t: RegThirdParty, u: RegUser |
        u.position in t.dataset.anon iff some gr : GroupRequest |
            (gr in t.requests and gr.ageSelector = u.age
            and #isItAnonymous[gr.ageSelector] >= 1)
}

--A SOSUser is in danger if and only if his/hers heartbeat is below the threshold.
fact inDangerIffBelowThreshold{
    all s: SOSUser | (s.heartbeat < s.threshold iff s in InDanger.users)
}

--Each SOSUser in danger must have an ambulance sent to his/hers position.
fact ambulanceInDanger{
    all u: SOSUser |
        u in InDanger.users implies some a: Ambulance | a.patient = u
}

--No ambulance is dispatched without an emergency.
fact ambulanceInDanger{
    all a: Ambulance | a.patient in InDanger.users
}

--Only one ambulance per RAU in danger must be dispatched.
fact noDoubledAmbulances{
    no disjoint a1, a2: Ambulance | a1.patient = a2.patient
}

--The destination of the ambulance must be the position of the user in danger.
fact ambulanceCorrectDestination{
    all a: Ambulance, u:SOSUser | a.patient = u implies a.destination = u.position
}

```

## 8.5 Assertions and predicates

```
-- If and only if a SOSUser's health parameters drop below his/hers threshold
-- an ambulance is dispatched.
assert ambulanceAlwaysWhenNeeded{
  (all u: SOSUser |
    u.heartbeat < u.threshold implies one a: Ambulance |
      (a.patient = u and a.destination = u.position)) and
  (all a: Ambulance | a.patient.heartbeat < a.patient.threshold)
}

-- If a request is correctly accepted the corresponding RTP has access to the user's data.
assert singleRequestAcceptedCase{
  all u: RegUser, t: RegThirdParty, sr: SingleRequest |
    sr in t.requests and sr.userFC = u.fiscalCode and sr.accepted = True implies
      u in t.dataset.users
}

-- If a request is declined the corresponding RTP has not access to the user's data.
assert singleRequestDeniedCase{
  all u: RegUser, t: RegThirdParty, sr: SingleRequest |
    sr in t.requests and sr.userFC = u.fiscalCode and sr.accepted = False implies
      u not in t.dataset.users
}

-- If a group request is issued and approved, then the corresponding RTP must have access to the
-- anonymized users' data.
pred groupRequest (gr: GroupRequest, t: RegThirdParty, u1, u2: RegUser, p1, p2: Position, a: Int){
  u1.age = a
  u2.age = a
  gr.ageSelector = a
  gr in t.requests
  u1.position = p1
  u2.position = p2

  p1 in t.dataset.anon and p2 in t.dataset.anon
}

pred show {
  #RegUser = 3
  #SOSUser = 2
  #InDanger.users = 2
}
```

## 8.6 Results

### 8.6.1 Executed commands

```
check ambulanceAlwaysWhenNeeded
```

```
check singleRequestAcceptedCase
```

```
check singleRequestDeniedCase
```

```
run groupRequest for 7 but 4 Int
```

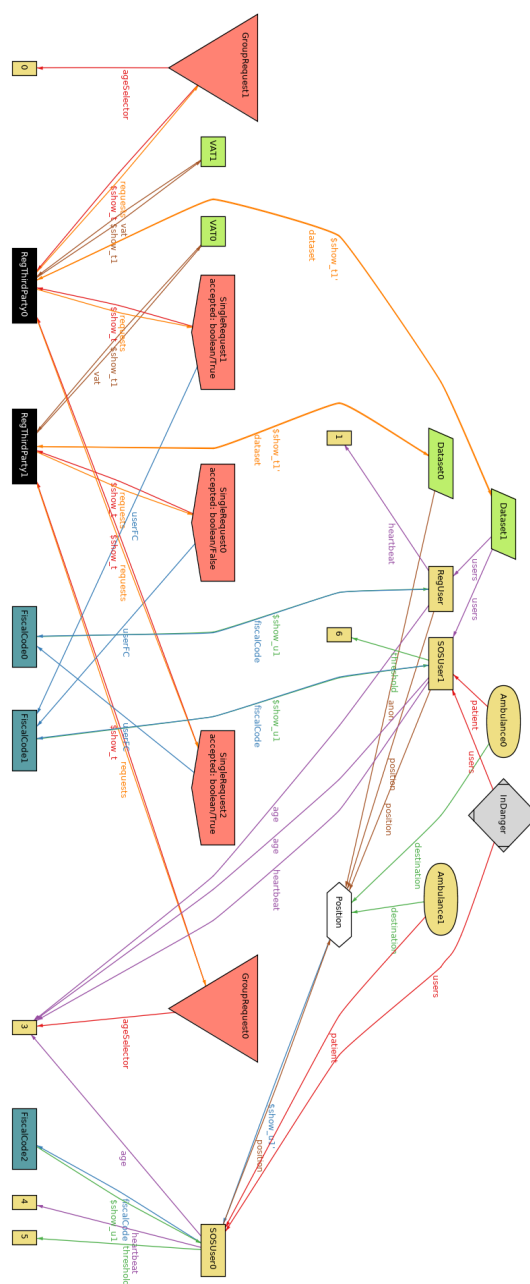
```
run show for 8 but exactly 3 SingleRequest, exactly 2 GroupRequest, exactly 2 RegThirdParty
```

*Note:* the Alloy Analyzer tool was run on Kubuntu 18.04, where some vertical purple lines randomly appear in the screen during the execution of the Analyzer itself; therefore, in the reported screenshot the same purple lines can be found.

5 commands were executed. The results are:

- #1: No counterexample found. ambulanceAlwaysWhenNeeded may be valid.
- #2: No counterexample found. singleRequestAcceptedCase may be valid.
- #3: No counterexample found. singleRequestDeniedCase may be valid.
- #4: **Instance found.** groupRequest is consistent.
- #5: **Instance found.** show is consistent.

One of the possible world generated with the model presented.



## 7. Appendix

### 7.1 Used tools

In the redaction of this document the following software tools were utilised:

- OpenOffice Writer
- Microsoft Word
- StarUML
- Alloy Analyzer 4.2\_2015-02-22
- Sequencediagrams.com
- Creately.com
- Github.com
- <https://www.draw.io>

### 7.2 Hours of work

In this section are reported the hours of work spent by the single group members in order to redact this document, with a brief description of the activities done during each session.

#### 7.2.1 Alberto Tiraboschi

<i><b>DATE</b></i>	<i><b>TASK</b></i>	<i><b>HOURS</b></i>
21/10/2018	Outline of the chapter	2
28/10/18	Use case diagrams produced, outline of the goals	6
30/10/2018	Chapter 3.2	4
31/10/2018	Chapter 3.3	5
3/11/2018	Revision	2
5/11/2018	End of Chapter 3	6
10/11/2018	Merge of the work done	5

### 7.2.2 Giorgio Polla

<b>DATE</b>	<b>TASK</b>	<b>HOURS</b>
15/10	Specification analysis	1
18/10	Specification analysis Requirements analysis	1
21/10	Domain assumptions Goals	2
23/10	Initial RASD layout	1
30/10	Goals RASD layout	3
02/11	Functional requirements	3
05/11	Functional requirements Non-functional requirements	2
06/11	Functional requirements Alloy	3
07/11	Alloy model	2
09/11	Alloy model	3
10/11	Merge of the work done RASD layout	6
11/11	Final review Complete rework of chapter 3 Correction of some UML diagrams	9