POLITECNICO
MILANO 1863

TrackMe project Giorgio Cozza and
Barbara Ferretti

Data4Help and AutomatedSOS

Requirements Analysis and Specification Document

Deliverable:	RASD
Title:	Data4Help and AutomatedSOS - Requirements Analysis and Specification Document
Authors:	Giorgio Cozza, Barbara Ferretti
Version:	1.0
Date:	11-November-2018
Download page:	https://github.com/GiorgioCozza/CozzaFerretti.git
Copyright:	Copyright © 2018, Giorgio Cozza and Barbara Ferretti – All rights reserved

Contents

List of Figures

1 Introduction

1.1 Purpose

TrackMe is a data management vendor whose business is mostly focused on healthcare. It wants to create a software system, called **Data4Help** to facilitate data acquisition by third parties of either a specific user or collective of users. Nowadays, in healthcare there would be tons of reasons in providing personal information: research centers may be interested in user data for carrying out large-scale studies or private clinics could monitor remotely their patients, likewise public institutions could manage such data for the public health assistance. Unfortunately, not all of them can afford software-based systems and related infrastructures to provide a service at this level: TrackMe and then Data4Help wants to cover an intermediary role in this situation.
The trend in this field is also heading to the developement of smarter systems able to ensure readness in emergency situations in which users need to be immediately assisted, rather than only be passively monitored. The support and availability of several public and private hospitals in providing resources required to implement the service has pushed TrackMe to build on top of Data4Help, another important service, able to achieve such purpose: **AutomatedSOS**.

This RASD (Requirement Analysis and Specification Document) aims to describe and analyze deeply the two problems, trying to define goals and requirements under the external environment conditions. It is followed, moreover, by a formal definition and consistency testing of critical aspects of both the services through Alloy modeling. It is intended as a guide for further phases of the design as well as a valid reference for possible legal agreements.

1.1.1 Goals

Data4Help

- **[G1]:** A user can register personal health monitoring devices in the system
- **[G2]:** TrackMe acquires periodically health parameters specifically related to a user
- **[G3]:** Third parties can access health data of specific users, if expressively authorized by the them
- **[G4]:** Third-parties can request anonymous information about groups of people

AutomatedSOS

- **[G5]:** The user has the possibility to specify the diseases he/she has, so the system can evaluate which health parameters to monitor
- **[G6]:** An ambulance may be sent within 5 seconds, if an emergency or anomaly condition is detected
- **[G7]:** A user designated as Supervisor of another one, is notified of both emergency and anomaly events occuring to the supervised user

1.2 Scope

Data4Help is a software system intended to facilitate the access to health data to third parties provided by users. The system covers an intermediary role between who wants to make personal information accessible to specific entities or specific purposes and who has an interest to access such information. Registered users can be monitored using personal sensor devices registered in the service and send information related to his/her health status to the system. Third parties, on the other side, can request to access to specific users' data only under authorization of the requested user, or acquire anonymous datasets of several Data4Help users whose identity is kept secret by the system. The service, of course must enforce a privacy protection policy to prevent third parties to trace back to user-protected information starting from an anonymous request.
AutomatedSOS instead, is built on top of Data4Help, it exploits real-time data provided by Data4Help users to provide immediate assistance when an emergency situation is detected. The service exists thanks to a network of Emergency Rescue Managers, a system provided by several hospital and clinics in order to manage and dispatch squads and vehicles when an emergency call is received.
Unfortunately, the actual sensing technology is premature in detecting unambiguously dangerous situations. The system will have to avoid the waste of medical resources in all those situations in which a detection may represent, with an appreciable probability, a false positive or a unproper usage of the service. In fact, a man who falls may be interpreted as a serious loss of consciousness or a minor injury. The system must ensure a response time to the user to check if everything is ok, otherwise alert an ambulance and all the people interested in the health status of the user.

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

User: both Data4Help and AutomatedSOS users.
Third Party: more frequentely are companies that looks for health information for their market research, but they could also be facilities like hospitals or other activities interested in the health field.
Anomaly condition: alteration of health monitored parameters that causes an alert to the user.
Emergency condition: alteration of the health monitored parameters that directly triggers the sent of an ambulance.
False Positive: an erroneous detection of the system.
Supervisor user: every user that has access to another user information and location.
Sensor/Monitoring Device: all the health sensor devices that can be registered in the Data4Help system.
Supervised User: an AutomatedSOS user who is monitored by the application and has one or more Supervisors.
ASOS Monitor Mode: flag that tells the system if the user must be monitored or not, it can be changed from client-side application.
Emergency Resource Manager: is a system already implemented in all the hospitals subscribed to the AutomatedSOS service, it handles emergency calls and AutomatedSOS notifications. For each case manages the resources required by the emergency.
RescueSquad: general term used to refer a registered group of rescuers provided with the equipment required for the specific type of emergency.
Clinical condition: a risk condition associated to one or more health indicators that must be monitored in order to prevent fatal effects.
Minimum required sensors: the minimum set of sensors in AutomatedSOS that allows the system to monitor at least one clinical condition.

1.3.2 Acronyms

ASOS: AutomatedSOS
D4H: Data4Help
API: Application Programming Interface
RASD: Requirement Analysis Specification Document
GPS: Global Positioning System
HR: Heart Rate
ECG: Electrocardiograph
BLE: Bluetooth Low Energy
TLS: Transport Layer Security
HTTPS: HTTP over TLS
ERM: Emergency Resource Manager
TP: Third Party

1.3.3 Abbreviations

[Gn]: n-th Goal
[Rn]: n-th Requirement
[Dn]: n-th Domain Assumption
24/7: 24 hours per day, for 7 days per week

1.4 Revision history

Up to now, there are no revision of this document as this is the first release.

1.5 Reference Documents

- Slides package: Requirement Engineerig part II
- Mandatory Project Assignement AY 2018-2019
- IEEE, *"IEEE Recommended Practice for Software Requirements Specifications"*

1.6 Document Structure

1.6.1 Chapter 1

In the first chapter is given a general presentation of the aim of this document, with the goals that the software has to satisfy. Morover are given other basic information in order to read easily the entire work, like the dictionary for example.

1.6.2 Chapter 2

Here is given a more detailed presentation of the software to be. In fact, in this chapter are presented the characteristics of the final users of the application, which will be the major functions of the system and the general interaction between the system and the user. Moreover are elencated all the constraints and the assumptions adopted in order to make the software work well.

1.6.3 Chapter 3

The third chapter is the most technical one. Here the requirements of the application are presented and is made clear the relation with them and the goals and the assuption of the previous chapters. Also various scenarios of a possible typical situation that needs the utilization of the software are listed. From them the use cases are created and so is possible to have, with the help of the UML diagrams, a more precise presentation of the interaction between the users and the system.

1.6.4 Chapter 4

This chapter is entirely dedicated to the analysis of the system, performed with Alloy. In the first subchapter the entire code is presented and, in the second one, the results of analysis and a representation of the generated worlds .

1.6.5 Chapter 5

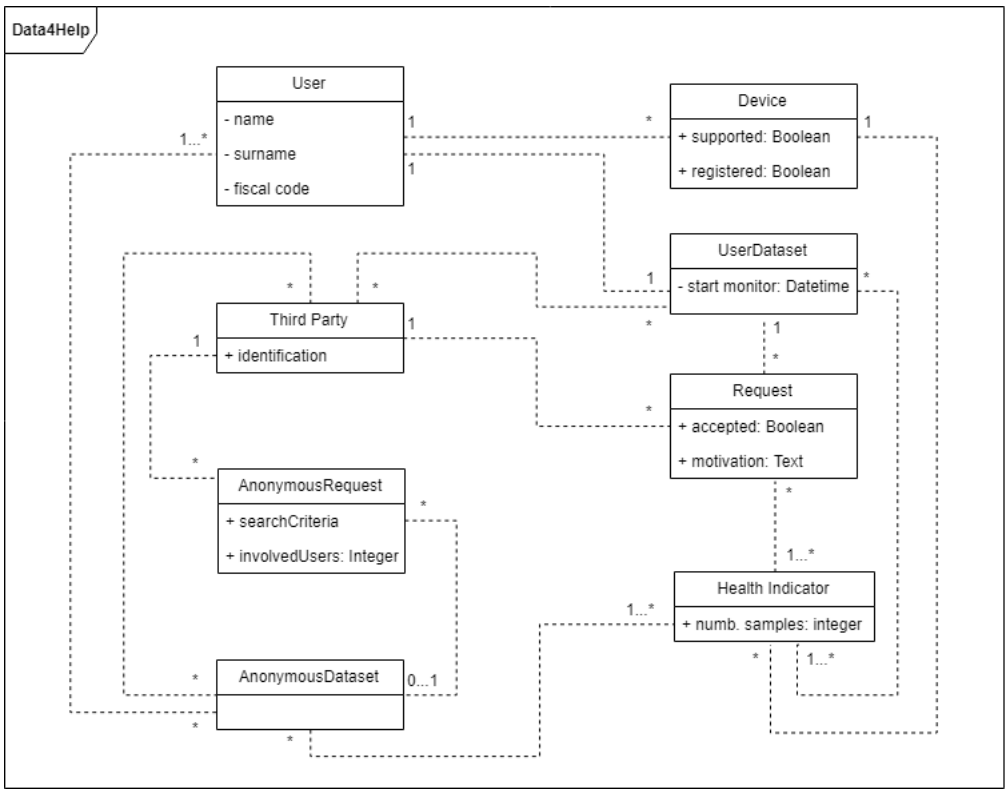
Chapter 5 presents the general amount of work required to complete this document and a list of references to the material we used to get the missing information.

2 Overall Description

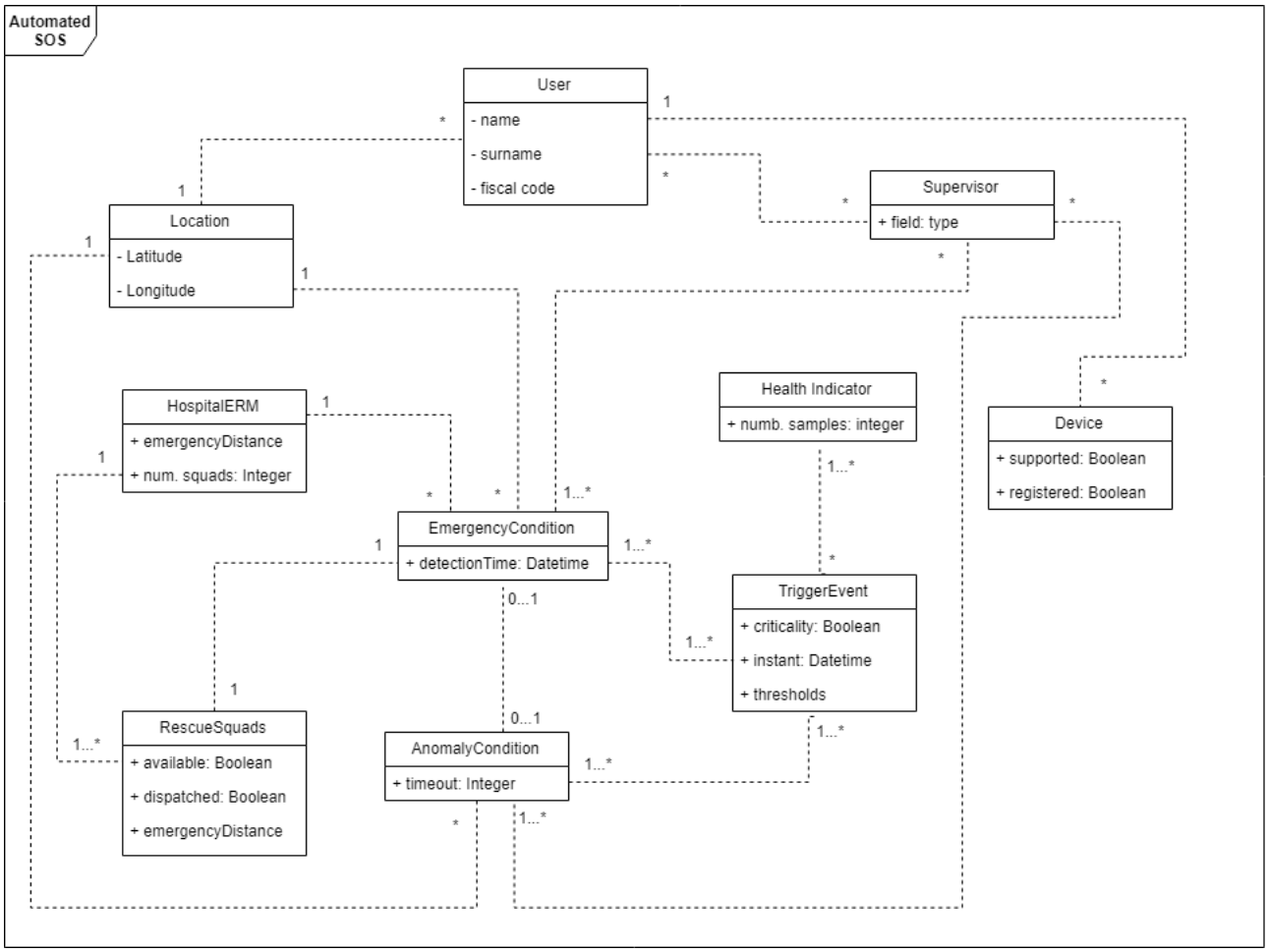
2.1 Product Perspective

Data4Help is a system that involves two type of entities: Users and Third Parties. In order to keep the difference between the two entities sharper, the User will be provided with an application, instead the Third Party will have to interact with the system with a website. Another reason for choosing this division is that for a company is easier to have a website to work on their researches instead of an application that have to be downloaded on a smartphone.
On the other hand, for AutomatedSOS an application is the best way to interact with the User.
The reason for dividing the Data4Help and AutomateSOS applications is that only the ones who wants to benefit from the advantages of AutomatedSOS will have to download the application on their smartphone. In this way the two applications remain as easy as possible, only providing the required services.
In any case, Data4Help and AutomatedSOS are not independent, they have common characteristics especially for what concerns the types of data provided by the user to the system and the mechanism to manage monitoring devices associated to both the services.

To provide an overview of the main system elements and what will be specifically explained in further sections for Data4Help, the following block diagram can be useful:



As in the previous case a diagram can clarify the AutomatedSOS perspective:



2.2 Product Functions

2.2.1 Data4Help

The main goal of Data4Help is to guarantee control on the health parameters of the users in order to give the possibility to third parties to obtain the data. Every user registered in Data4Help knows that his/her registered parameters could be used for market information, but also for helping researchers to discovers new treatments. This is made possible by providing two types of registration: the registration as a user and the registration as a third party.
There will be the possibility for the users to accept or deny personal requests from third parties, in this way there will be no privacy violations. On the other hand, the system will allow third parties to formulate anonymous requests. These are concepted to not disclose the identity of the user, providing information of an eteregenous set of users according to several requestor-specified search criteria. Unfortunately, it is trivial for a third party to trace back to sensitive information of a user providing restricting search parameters. After an attempt analysis, TrackMe evaluated a policy to prevent this possibility: in order to provide the result of an anonymous request to a third party, the involved users must be at least 1000.

2.2.2 AutomatedSOS

AutomatedSOS wants the users to feel safe. Provided the necessary sensor devices, the system guarantee a constant control on the health parameters. In order to minimize medical resource wasting, it is necessary for the system to discriminate carefully which conditions may lead to a fatality (e.g: heart attack) from those which can be simply false positive detections, uncareful behaviors of the person who is using the service, non-serious injuries or ambiguous observations. To this end, the system will be able to detect emergency and anomaly conditions. Obviously, in order to guarantee that the number of false positives remains as low as possible, the user is required to give correct health information when he/she registers to the service. Furthermore the system will not allow a user to choose arbitrarily which set of sensor types will be used to monitor his/her health conditions. For each specific desease selected, there will be a set of specific sensor devices required.
The system will also provide the possibility to accept supervisors. This particular characters are AutomatedSOS users concepted to be supported by the system for giving assistance to other ones designated as supervised. Each time an AutomatedSOS users is in danger the system will be able to provide to the desigated supervisor the necessary information. In this way, in case of emergency, there will be someone promptly informed.
As is known that it is not always possible to be monitored by the system, there is the possibility to notify the latter that it has to stop controlling the user parameters by turning off the service. The system also stops controlling them when the minimum required sensors are not avaiable (maybe because they run out of battery), sending a notification to the user.

2.2.3 Requirements

Data4Help

- **[R1]:** The system must allow a registered user to associate a sensor device to the service
- **[R2]:** Each user is uniquely identified by the system
- **[R3]:** The system can acquire health data by specific sensors connected to the user's main device
- **[R4]:**The system must provide the possibility to users to insert specific health information that cannot be acquired through other monitoring devices
- **[R5]:** The system must be able to identify and certificate the reliability of each organization that wants to request user data
- **[R6]:** Each registered organization that wants to access health data of specific users must be able to formulate a request providing information related to the purpose of the request
- **[R7]:** The system must notify each user of a third party request as soon as it is formulated, and allow him/her to accept or reject the request
- **[R8]:** For each third party request the system should provide to the user information related to the requestor and the purpose of the request
- **[R9]:** Once a third party request is accepted by the user, the third party must have full access to the entire collection of data of the user
- **[R10]:** Third parties must be able to request health data of groups of anonymous users, according to several criteria without being expressively authorized

- **[R11]:** The system must prevent third parties to trace back to specific user information through anonymous requests

AutomatedSOS

- **[R12]:** The system must give the possibility to the user to specify which diseases he/she has
- **[R13]:** The system must prevent the possibility to use the service, if the minimum required sensors are missing
- **[R14]:** The system must allow the user to turn on and off the service each time he/she wants
- **[R15]:** The system must stop monitoring when minimum required sensors are missing
- **[R16]:** The system must be able to distinguish and detect emergency or anomaly conditions occurring to a specific user
- **[R17]:** When an anomaly condition is detected, the system must send a notification to the user, asking if it is an emergency condition. If the user does not answer the notification within 30 seconds, the anomaly must become an emergency
- **[R18]:** When an emergency condition is detected, the nearest ambulance must be alerted, providing all the information about the situation
- **[R19]:** A user must have the possibility to request to become Supervisor of another one
- **[R20]:** Each supervised user must be able to accept or reject the possibility to have a Supervisor
- **[R21]:**The Supervisor must be notified by the system of all emergency and anomaly conditions occurring to the supervised user

2.3 User Characteristics

In Data4Help the main actor is who we already called User. He/She is the one that provides health information to TrackMe while is monitored after the registration to the service. Without this presence, the application does not have any reason to exist. He/She is uniquely recognizable by his/her fiscal code and his/her login information.
There is also the presence of another actor: the Third Party. It looks for the information provided by the users for its own interests (from business to healthcare). Third parties can access to the service if and only if they can provide a certification about who they are. Without a valid certification the system denies the registration.

For what concerns AutomatedSOS and the Users of the application, in a first hypothesis the idea was to offer AutomatedSOS services only to elderly people. A more deep consideration of the provided service led to the decision to permit to everyone that needs help to access to the application. A lot of young people need to be monitored as well as older one.

2.4 Assumptions, Dependencies and Constraints

2.4.1 Domain Assumptions

Data4Help

- **[D1]:** The device to be registered, is supported by the system
- **[D2]:** Data acquired by the connected sensors is intended to be accurate
- **[D3]:** Health information manually provided by the user is intended to be

AutomatedSOS

- **[D4]:** The user is registered to AutomatedSOS
- **[D5]:** The sensor devices monitoring the patient are providing reliable data
- **[D6]:** The position of both the ambulance and the user are accurately
- **[D7]:** The Supervisor is a Data4Help registered user
- **[D8]:** The Supervisor is available when the notification is sent

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 D4H: User Interfaces

The following figures (alse the ones presented in Subsection 3.1.2 and 3.1.3) are intended to give a general idea of the final result. The final application will not be necessarly as it is presented here.

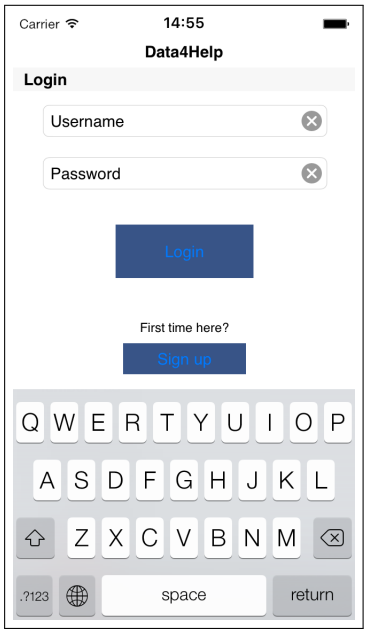


Figure 1: User Login

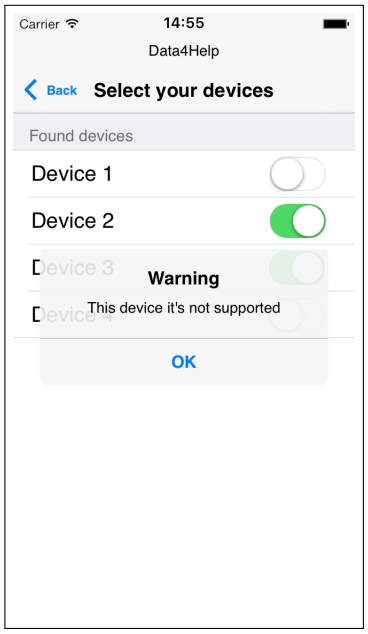


Figure 2: Device not supported

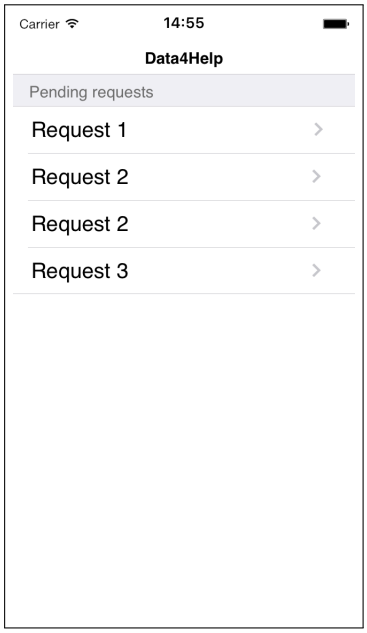


Figure 3: Pending TP requests

Figure 4: Evaluate Request

3.1.2 D4H: Third Party Interfaces

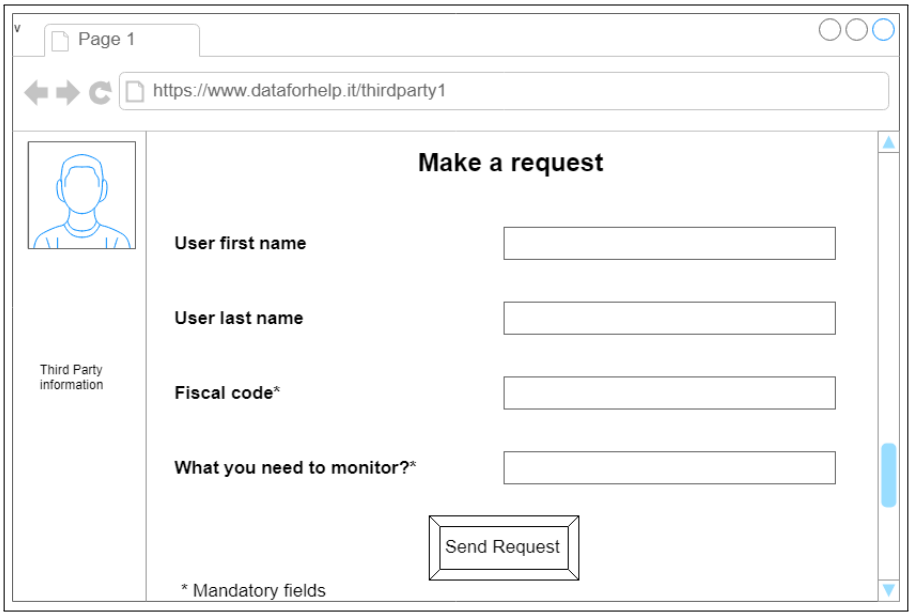


Figure 5: TP make a request

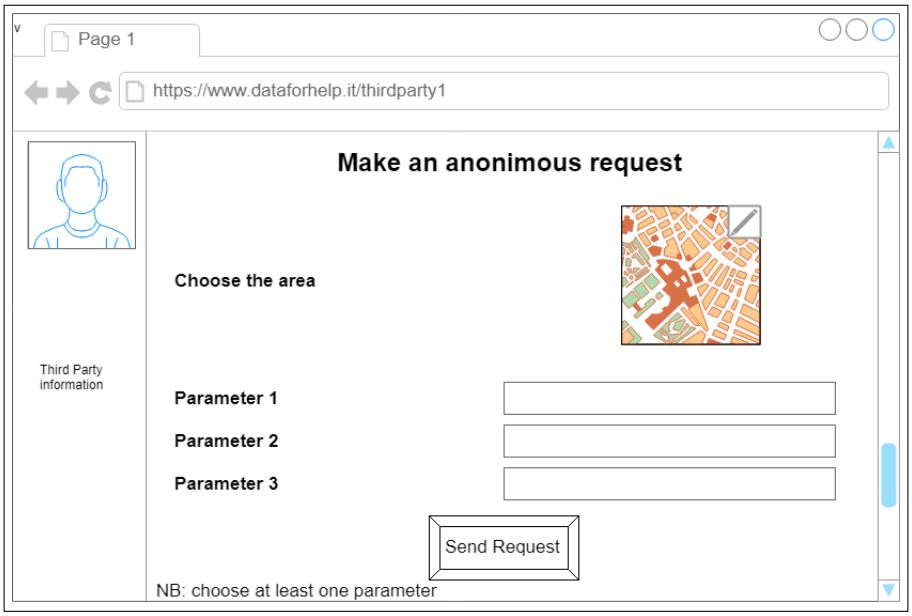


Figure 6: TP make an anonymous request

3.1.3 ASOS: User Interfaces

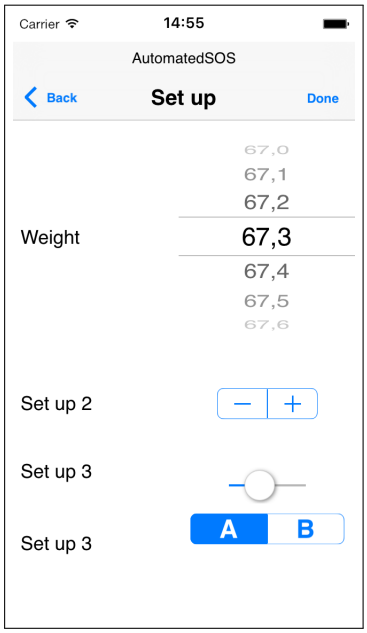


Figure 7: User Health Profile part 1

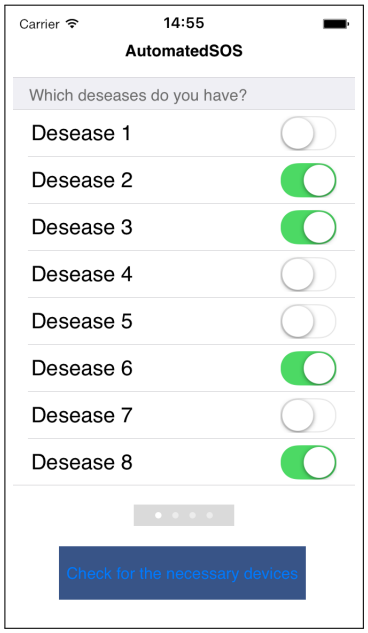


Figure 8: User Health Profile part 2

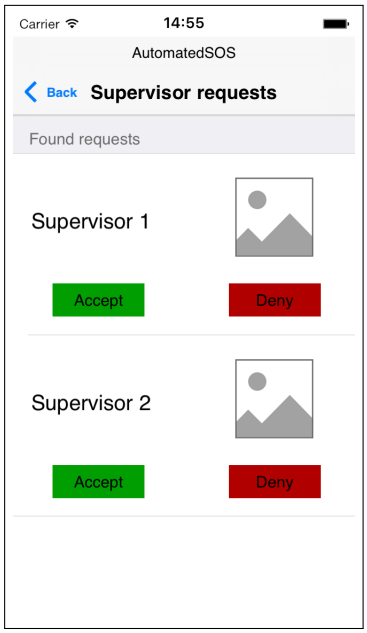


Figure 9: Supervisor Invitation

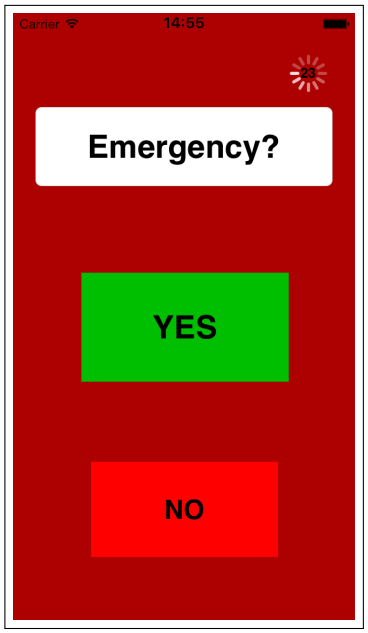


Figure 10: Anomaly Detection

3.1.4 Hardware Interfaces

In order to work well and give the possibility to take advantage of the services offered by the system, both Data4Help and AuomatedSOS are required to be installed in a smartphone that provides:

- GPS system

- Bluetooth (or BLE) interface

The first requirement is needed especially for ASOS users, as substitute of GPS bracelets or similar, in order to provide the position of the user each time the system asks for it.
The second requirement is important because most of the available sensor devices on the market can be connected to the smartphone using Bluetooth or BLE interfaces.
Sensor devices are not mandatory but strongly suggested to allow D4H to acquire health parameters (like blood pressure, HR, breath rate and so on), without them, the application does not provide consistent advantages to the users.

3.1.5 Software Interfaces

D4H and ASOS rely only on Google Maps API as external service. Both the computation and the storage part of the system are completely hosted and managed by TrackMe. Depending on more external services can affect negatively the reliability of the system.

3.1.6 Communication Interfaces

In order to ensure confidentiality, data integrity and server authenticity, HTTPS is likely to be used as network protocol to manage the encrypted communication between the client (user or third party) and TrackMe servers for security reasons. Since the protocol uses TCP at transport layer is probable to deal with problems related to delays in communication. A deep analysis of how much acceptable are such delays must be carried out.

3.2 Data4Help Scenarios

3.2.1 Scenario 1

Ross is a Dr. Herbert's patient. He discovers that the doctor is registered to Data4Help, the new service provided by TrackMe and decides to subscribe to it in order to facilitate the doctor to access his own health indicators. Ross downloads the app on his smartphone, runs it and fullfils the form for the registration. Once this procedure is completed, the service detects that a compatible smartwatch and a respiratory rate monitoring device are connected to the mobile phone through the Bluetooth interface. The application asks Ross which one he wants to register to the service. Ross selects both of them and allows the system to complete the operation. During the evening, Dr. Herbert receives a message from Ross who confirms his registration to Data4Help. The doctor logs in the service using the hospital credencials and inserts "Ross Gilbert" in the search form. Dr. Herbert, after the system founds the user's account, formulates the request including a motivation and sends it. Ross, the morning after, runs the application and finds a notification of the awaiting request, so he checks for the requestor and accepts it.

3.2.2 Scenario 2

Angelo has bought two smart devices: an elastic strip, equipped with a sensor able to measure blood pressure, and an electronic bracelet for monitoring HR. He discovers that they can be interfaced with his smartphone through Bluetooth interface, so requests the application to register the devices. The service starts to check if the detected sensors are compatible or not. Unfortunately, only the sensor strip is

completely supported by the system. So Angelo accepts to register only one device and the app, once performed the operation, update its status and return to collect data from all the connected devices.

3.2.3 Scenario 3

SoftGalaxy is a software company that is working on a new application to provide to registered users a modern service called Health Advisor. It can suggest changes in daily habits, in diet and so on, according to user's data provided to the system. Unfortunately, the company cannot afford a large-scale system to collect information in real-time. In order to save as much resources as possible, it decides to rely on Data4Help to achieve this task. Before releasing the application, SoftGalaxy connects to the system and registers itself as third party, fulfilling the proper form. In addition, Data4Help asks the company to provide its digital certification for security reasons.

3.2.4 Scenario 4

The company Virgin Active is considering opening a new gym near Saint Ambrogio church in Milan. In order to do so, it decides to make an anonimous requate un Data4Help to receive the information about all the overweight people that live in the area of Saint Ambrogio. It recieve a negative answer because there are only 593 people that suits with the requirements. Virgin Active has to change the request and so asks the same information but for a larger area, taking in also the area of Saint Agostino. This time there are enough people to let the request being forwarded and so Virgin Active can receive the information needed.

3.3 AutomatedSOS Scenarios

3.3.1 Scenario 1

Michelle has an elderly mother, Teresa, who suffers about a rare heart disease. She is constantly monitored by a portable ECG device due to the high risk of being hit by a heart attack. Unfortunately, Michelle works during the morning and in this period of the day no one takes care of Teresa. She suggests to the mother to rely on AutomatedSOS, a Data4Help-based service. Since the elderly woman has a smartphone with Data4Help installed and a personal account, she decides to download the application and logs in with the Data4Help credencials. The system notices that Teresa is using the service for the first time, so it asks some information about her health problems and through it the app creates an health profile and a list of required sensors. The ECG that is actually monitoring her, is already registered and used by Data4Help, but since she has specified that suffers about loss of consciousness the system warns her that a fall sensor should be registered to provide the proper assistance. Teresa has not the device yet, so ignores the warning and complete the profiling procedure. The app finally, is ready to monitor the woman's health status. The they after, when Michelle went to work, Teresa activated the ASOS Monitor Mode to start to be monitorated. As Michelle returned, she deactivated it.

3.3.2 Scenario 2

One month ago Marianna, a 60 years old woman, decided that it could be a good idea in order to live better and safer to join the AutomatedSOS application. As she was already a member of Data4Help she did not have to create a new profile but she simply logged in the new downloaded application as a Data4Help user. The system asked Marianna to fill a questionnaire about some basic information of

herself, select her deseases and register the necessary devices. Paolo, her husband, that was already an AutomatedSOS user, sends Marianna a Supervisor request that she promptly accepted. A few days after, she had dinner with friends and ordered a piece of cake without knowing that inside it there were almonds, a food that gives her anaphylactic shock. Immediately she started to breath bad, her heartbeat increased and also her pressure. Her phone rung, AutomatedSOS was asking her if everything was ok. As Marianna needed an ambulance, she answered negatively. Paolo received a notification from AutomatedSOS warning that Marianna was sick meanwhile an ambulance arrived at the restaurant, saving Marianna.

3.3.3 Scenario 3

Carol is a young student, she suffers of frequents loss of consciousness episodes due to medicines she has to take for a congenital disease. Carol is an AutomatedSOS user as well as her mother, Jenny, that decides to watch over the health conditions of her daughter when she is not home. So, Jenny in order to be notified of all the emergencies and provide support when she is not near her daughter, opens AutomatedSOS and sends to Carol an invitation for becoming her Supervisor. Carol, during the school break, receives the request and accepts it. Becoming a Supervisor, Jenny is allowed to be notified of the daughter health status when something goes wrong and to know the position of Carol.

3.4 Functional Requirements

3.4.1 Data4Help

[G1]: A user can register personal health monitoring devices in the system

- [R1]: The system must allow a registered user to associate a sensor device to the service.
  - [D1]: The device to be registered, is supported by the system.

[G2]: TrackMe acquires periodically health parameters specifically related to a user

- [R2]: Each user is uniquely identified by the system.
- [R3]: The system can acquire health data by specific sensors connected to the user's main device.
  - [D2]: Data acquired by the connected sensors is intended to be accurate.
- [R4]: The system must provide the possibility to users to insert specific health information that cannot be acquired through other monitoring devices.
  - [D3]: Health information manually provided by the user is intended to be correct.

[G3]: Third parties can access health data of specific users, if expressively authorized by the them

- [R5]: The system must be able to identify and certificate the reliability of each organization that wants to request user data.
- [R6]: Each registered organization that wants to access health data of specific users must be able to formulate a request providing information related to the purpose of the request.

- [R7]: The system must notify each user of a third party request as soon as it is formulated, and allow him/her to accept or reject the request.
- [R8]: For each third party request the system should provide to the user information related to the requestor and the purpose of the request.
- [R9]: Once a third party request is accepted by the user, the third party must have full access to the entire collection of data of the user.

[G4]: Third-parties can request anonymous information about groups of users

- [R10]: Third parties must be able to request health data of groups of anonymous users, according to several criteria without being expressively authorized.
- [R11]: The system must prevent third parties to trace back to specific user information through anonymous requests.

3.4.2 AutomatedSOS

[G5]: The user has the possibility to specify the diseases he/she has, so the system can evaluate which health parameters to monitor

- [R12]: The system must give the possibility to the user to specify which diseases he/she has.
  - [D4]: The user is registered to AutomatedSOS.
- [R13]: The system must prevent the possibility to use the service, if the minimum required sensors are missing.

[G6]: An ambulance may be sent within 5 seconds, if an emergency or anomaly condition is detected

- [R14]: The system must allow the user to turn on and off the service each time he/she wants.
- [R15]: The system must stop monitoring when minimum required sensors are missing.
- [R16]: The system must be able to distinguish and detect emergency or anomaly conditions occurring to a specific user.
- [R17]: When an anomaly condition is detected, the system must send a notification to the user, asking if it is an emergency condition. If the user does not answer the notification within 30 seconds, the anomaly must become an emergency
- [R18]: When an emergency condition is detected, the nearest ambulance must be alerted, providing all the information about the situation.
  - [D5]: The sensor devices monitoring the patient are providing reliable data.

  - [D6]: The position of both the ambulance and the user are accurately provided by the GPS system.

[G7]: A user designated as Supervisor of another one, is notified of both emergency and anomaly events occuring to the supervised user

- [R19]: A user must have the possibility to request to become Supervisor of another one.
  - [D7]: The Supervisor is a Data4Help registered user.
- [R20]: Each supervised user must be able to accept or reject the possibility to have a Supervisor.
- [R21]: The Supervisor must be notified by the system of all emergency and anomaly conditions occurring to the supervised user.
  - [D8]: The Supervisor is available when the notification is sent.

3.5 UML Modeling

3.5.1 Use Case Notation

In order to facilitate the comprehension of use cases, the "Exception" part of tables will contain, for each item, the corresponding step in the events flow at which the exception may be raised. Furthermore, in all the cases in which a use case is included in the description of another one, the relative part that mentions the included use case is represented underlined with a clear reference to the same one (e.g see: "Register Device"). In some tables for space and document organization needs, "Special Requirements" part has been removed, in all of these occurrences there are no special requirements.

3.5.2 Data4Help Use Cases

User Login	
Actors:	User
Entry Conditions:	The user is already registered in the service
Flow of Events:	1. The user inserts the access credencials 2. The user taps on "Log in"
Exit Conditions:	The system redirect the user to his/her personal account
Exceptions:	• 2) If the inserted credencials are not valid, the system sends a warning and asks for rewrite them
Special Requirement:	None

User Signs Up	
Actors:	User
Entry Conditions:	The user has already downloaded the application
Flow of Events:	1. The user clicks on "Signs up as User" 2. The user inserts all the required information 3. The user confirms the registration 4. The system asks the user if he/she wants to associate a sensor device to the service. 5. The user accepts to register the devices 6. The system redirect the user to the Device Registration space 7. The user registers the devices (see: "Register Device") connected to the smartphone 8. The system redirect the user to his/her personal account
Conditions:	The user is successfully signed up
Exceptions:	• 3) The user is already registered to the service, a notification will be sent, preventing him/her to register again. • 4) The user skips the device registration and the system redirect the user to his/her account

ThirdParty Sign Up	
Actors:	Third Party
Entry Conditions:	The third party has not registered to the service yet
Flow of Events:	1. The third party clicks on "Sign Up as Third Party" 2. The third party inserts the requested information 3. The third party provides a certification of its identity 4. The third party confirms the information provided 5. The system verifies the inserted data 6. The system confirms the registration and redirect the third party to its personal account.
Exit Conditions:	The third party is registered to Data4Help
Exceptions:	• 5) If the information provided is incorrect or the third party is already registered, the system shows a warning and denies the registration

Register Device	
Actors:	User
Entry Conditions:	The user is logged in Data4Help
Flow of Events:	1. The user taps on "Add Device" 2. The system provides the list of all the sensor devices connected to the smartphone 3. The user selects the devices that wants to associate to the service 4. The system registers the devices
Output Conditions:	A new monitoring device is added in the system
Exceptions:	• 4) If the registered sensor corresponds to a new monitored parameter the information is associated to the current user in the system • 4) If the device is not supported, the system sends a warning to the user and denies the device registration.
Special Requirement:	None

ThirdParty Logs in	
Actors:	Third Party
Entry Conditions:	The third party is already registered in the service
Flow of Events:	1. The third party clicks on "Log in as Third Party" 2. The third party inserts the access credencials
Exit Conditions:	The system redirects the third party to its personal account
Exceptions:	• 2) If the inserted credencials are not valid, the system sends a warning and asks for rewrite them
Special Requirements:	None

Make a Request	
Actors:	Third Party
Entry Conditions:	The third party has already signed up for Data4Help
Flow of Events:	1. The third party clicks on "Create Request" 2. The third party selects the user data types that wants to access and inserts the information required to find the user 3. The third party provides the motivation of the request. 4. The system checks if someone matches the search criteria and provide the result 5. The third party clicks on "Send Request" 6. The system notifies the third party that the request has been sent to the user
Exit Conditions:	The request is sent to the specific user
Exceptions:	• 4) If the specified search criteria do not provide a valid result, the system shows a warning and the request is not created.
Special Requirements:	The third party knows exactly the minimum search information required to find the specific user

Evaluate Request	
Actors:	User
Entry Conditions:	• The user is logged in Data4Help • The user is notified by the system that a third party request has been received
Flow of Events:	1. The user enters the pending requests 2. The user selects the last request 3. The user taps on the request 4. The system shows to the user identity and motivation of the requestor 5. The user accepts the request 6. The system notifies the third party about the decision of the user
Exit Conditions:	The system allows the third party to access the requested user's dataset
Exceptions:	• 3) If the request has been aborted by the third party the system notifies the user that the request is no longer acceptable or rejectable • 5) If the user rejects the request, the system notifies the rejection to the third party

Make an Anonymous Request	
Actors:	Third party
Entry Conditions:	The third party is logged in Data4Help
Flow of Events:	1. The third party clicks on "Create Anonymous Request" 2. The third party specifies the search criteria required to find the group of anonymous users 3. The system checks if the number of involved anonymous users are at least 1000, according the search criteria 4. The system notifies the requestor that a valid dataset has been found
Exit Conditions:	The third party obtains the anonymous dataset
Exceptions:	• 3) If the anonymous users involved in the request are less than 1000 or no one matches the search criteria, the system notifies the event and rejects the request.
Special Requirements:	None

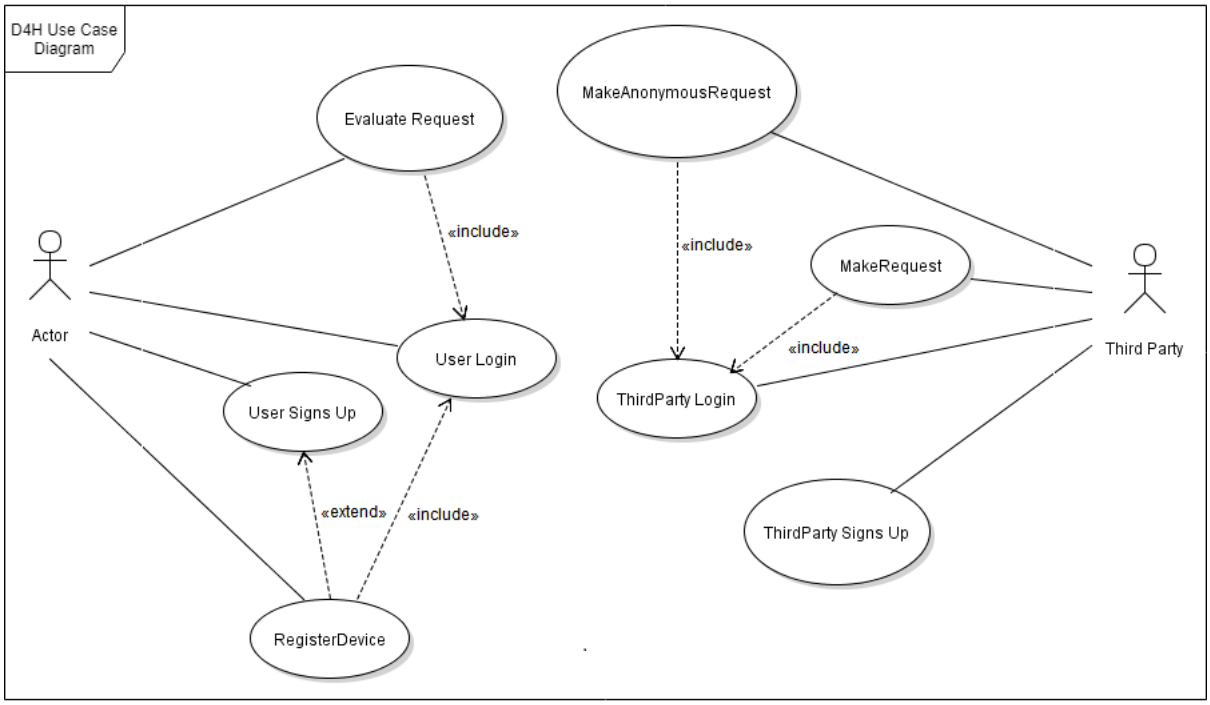


Figure 11: Data4Help Use Case Diagram

3.5.3 AutomatedSOS Use Cases

User Logs in ASOS	
Actors:	User
Entry Conditions:	The user is already registered in Data4Help
Flow of Events:	1. The user inserts the D4H access credencials 2. The user clicks on "Log in"
Exit Conditions:	The system redirects the user to his/her personal AutomatedSOS account
Exceptions:	• 2) If the inserted credencials are not valid, the system sends a warning and asks for rewrite them • 2) If the user has logged in for the first time, the system starts to profile the health conditions of the user (see:"Change User Health Profile")
Special Requirements:	None

Change Health Profile	
Actors:	User
Entry Conditions:	The user is logged in for the first time
Flow of Events:	1. The user clicks on "Manage Health Profile" 2. The user inserts personal information regarding age, daily habits, addictions, diagnosed diseases 3. The system evaluates and provides a list of clinical conditions that can be monitored according to the inserted information 4. The system checks if the sensors required for monitoring the given clinical conditions are registered to D4H and connected to the smartphone 5. The user confirms the result
Exit Conditions:	The health profile is updated and the user is monitored according to the new clinical conditions
Exceptions:	• 3) If none of the minimum required sensors are registered or connected, the system warns the user and denies the service. • 3) If only a part of the required sensors (at least the minimum required ones) are registered in the service, the user selects which of the available clinical conditions he/she wants to be monitored and confirms.
Special Requirements:	None

Send Supervisor Request	
Actors:	User
Entry Conditions:	The user is logged in AutomatedSOS
Flow of Events:	1. The user clicks on "Become Supervisor" 2. The system asks the identification of who should be supevised 3. The user inserts the fiscal code of the supervised user 4. The system provides the result of the research 5. The user sends the request
Exit Conditions:	The receiver is notified of the supervisor request
Exceptions:	• 4) If the research criteria do not match any existing AutomatedSOS user, the system sends a notification to the user
Special Requirements:	None

Anomaly Condition is Detected	
Actors:	User
Entry Conditions:	• The user is logged in AutomateddSOS • The AutomadesSOS Monitor Mode is switched on • AutomatedSOS system has detected an Anomaly Condition • The ASOS Monitor Mode is activated
Flow of Events:	1. The user is notified by the system that asks to him/her if an ambulance is needed 2. The user confirms that is an emergency 3. The system turns the anomaly into an emergency condition
Exit Conditions:	The system raises an emergency condition (see:"An Emergency Condition is Detected")
Exceptions:	• 2) If the user is fine, he/she notifies the system that an ambulance is not required • 2) If the user does not reply to the notification, the anomaly timeout expires and the system alerts the nearest hospital
Special Requirements:	The timeout must be fixed to 30 seconds

Evaluate Supervisor Request	
Actors:	User
Entry Conditions:	• The user is logged in AutomatedSOS • The user is notified that a Supervisor request has been received
Flow of Events:	1. The user enters the list of the pending supervisor requests 2. The user selects the request 3. The system provides the identity of the requestor 4. The user accepts the request
Exit Conditions:	The system notifies the new Supervisor and allow him/her to access the user's health status and position during emergencies
Exceptions:	• 4) If the user rejects the request the system notifies the requestor of the refuse.
Special Requirements:	None

An Emergeny Condition is Detected	
Actors:	EmergencyResourceManager
Entry Conditions:	• An emergency condition is detected. • An anomaly is turned into an emergency condition • The ASOS Monitor Mode is activated
Flow of Events:	1. The system sends an alert to the nearest hospital's ERM providing the identification and position of the user, the reason of the emergency, the time instant at which the event has been detected 2. The ERM alerts and dispatches that available RescueSquad which is the nearest to the user in danger
Exit Conditions:	The system is notified that a RescueSquad has been dispatched to the user
Exceptions:	• 2) If no RescueSquad for the alerted hospital is available, the system is informed about and sends the request to the ECM of the second nearest hospital, and so on if also the second attempt fails. If one of the alerted ECMs dispatches a RescueSquad for first, the system notifies the others of the dispatch. • 1) If the user is associated to one or more Supervisors, these ones are also alerted of the emergency and informed of the position of the user.
Special Requirements:	The time spent between the emergency detection and the RescueSquad dispatch must be at most 5 seconds

Turn on ASOS Monitor Mode	
Actors:	User
Entry Conditions:	• The user is logged in asos • The ASOS Monitor Mode is off
Flow of Events:	1. The user taps on turn on asos monitor mode 2. The system checks if there are the minimum required sensors
Exit Conditions:	The user starts to be monitored by the system
Exceptions:	• 2) If one or more of the minimum required sensors turn off, a notification is sent to the user and the asos monitor mode turns off
Special Requirements:	None

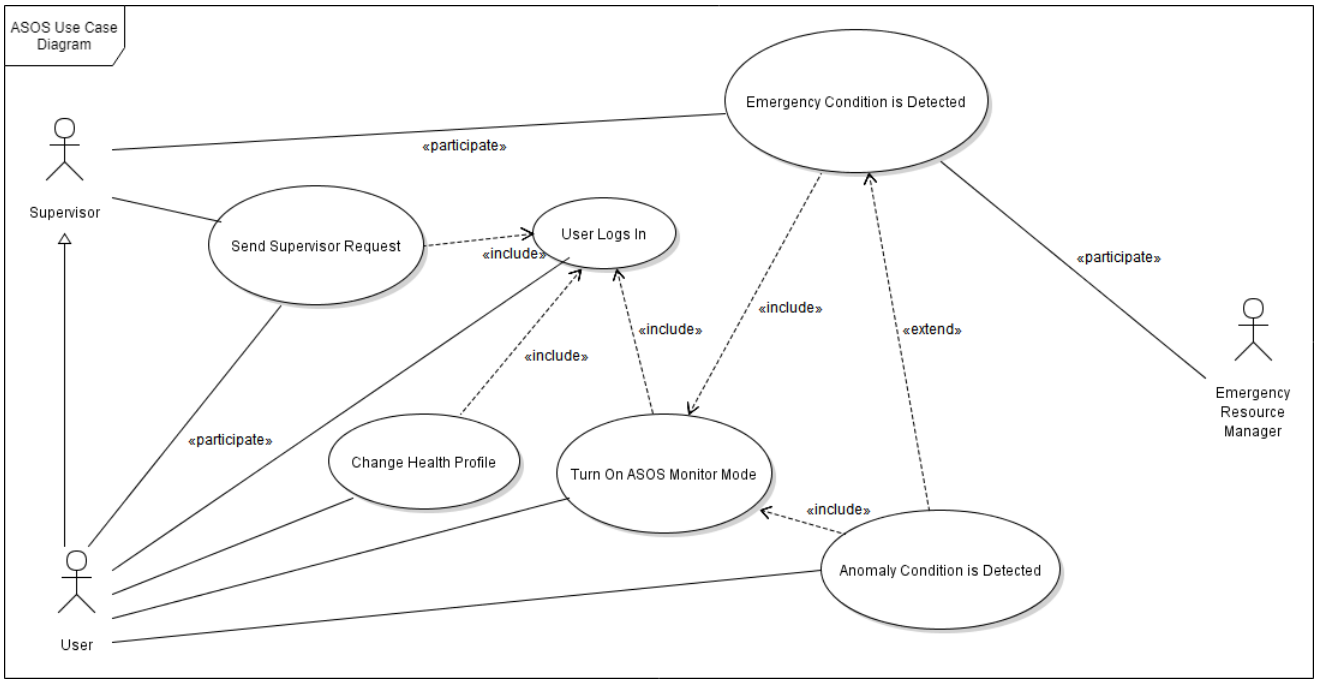


Figure 12: AutomatedSOS Use Case Diagram

3.5.4 Data4Help Sequence Diagrams

Register Device

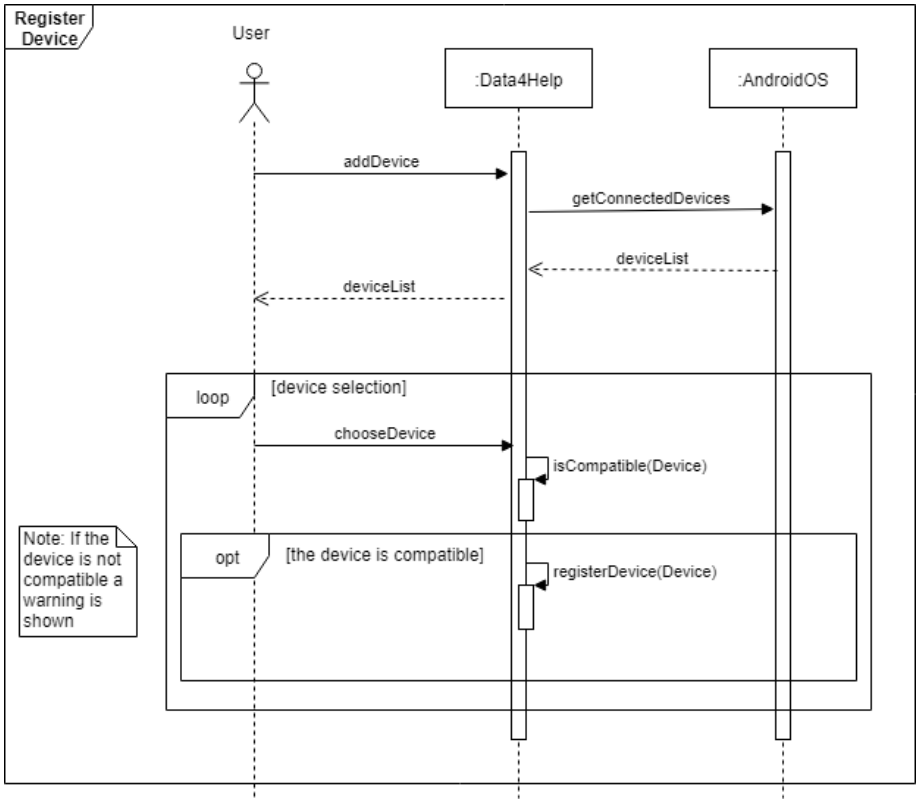


Figure 13: Register Device Event Sequence

Make Request

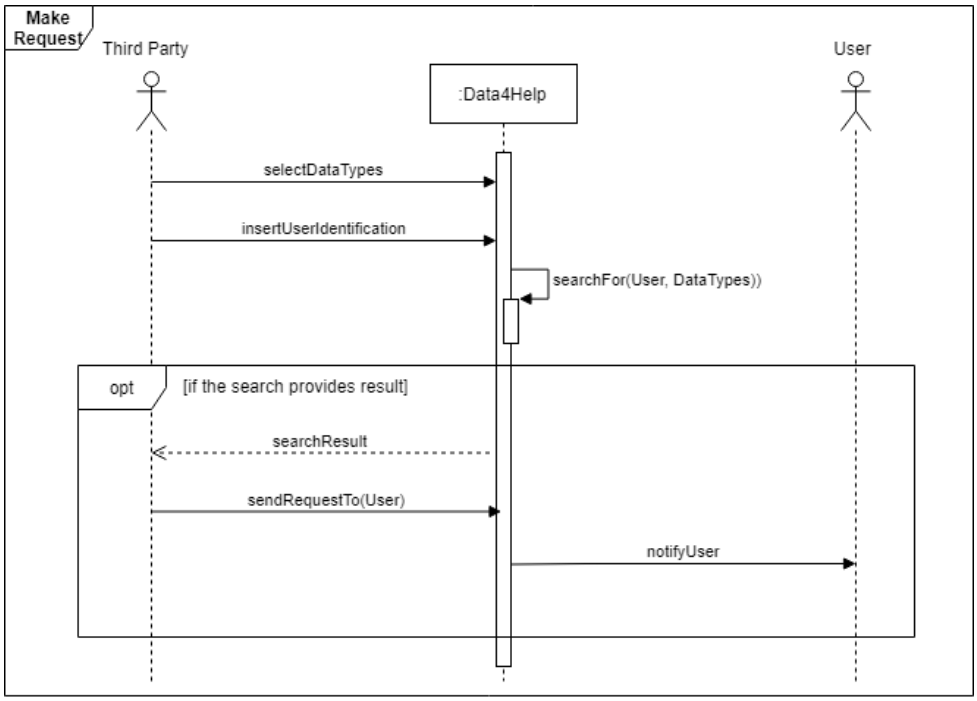


Figure 14: Make Request Event Sequence

Evaluate Request

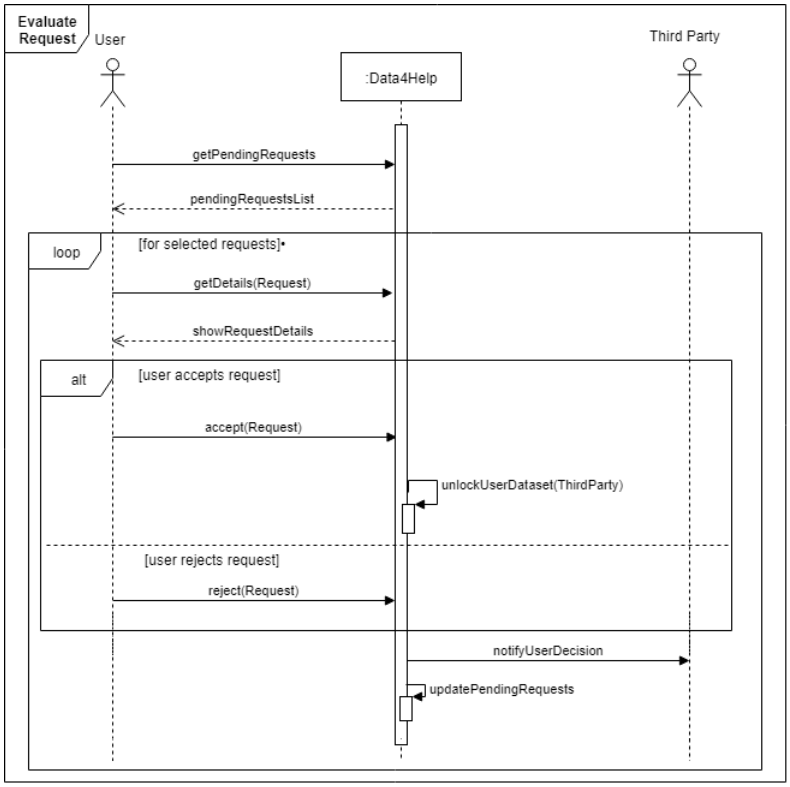


Figure 15: Evaluate Request Event Sequence

3.5.5 AutomatedSOS Sequence Diagrams

Emergency Condition is Detected

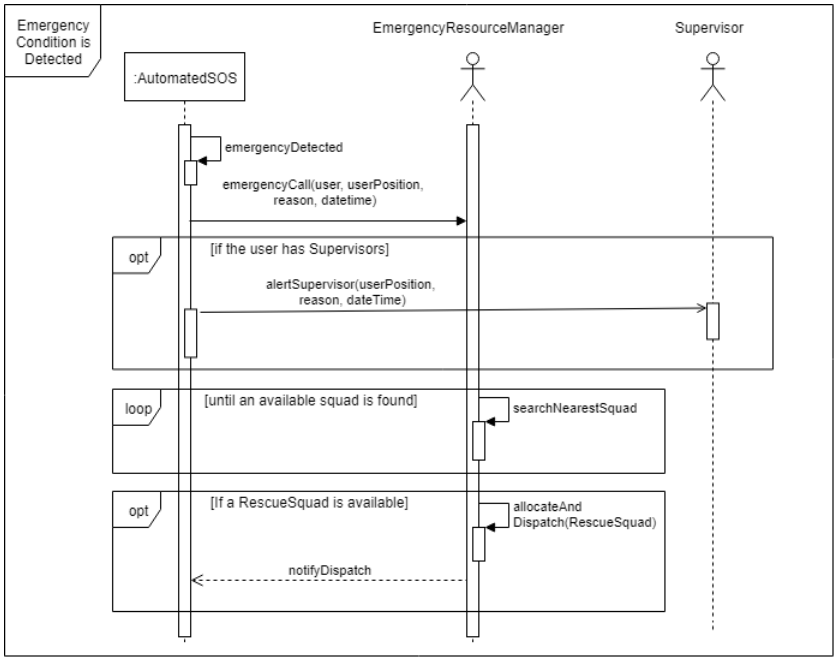


Figure 16: Emergency Condition Detection Event Sequence

3.6 Performance Requirements

Both Data4Help and AutomatedSOS aim to help the highest number of people. AutomatedSOS must guarantee that the user's information is sent in real time to the application so the emergency can be solved as soon as possible. A delay higher than 3 seconds can't be tolerated. Data4Help have to make avaiable the information of each user during an entire month.

3.7 Design Constraints

Unfortunately one of the biggest limitation in the software to be is due to the large variety of monitoring devices available on the healthcare market. Companies in some cases are reluctant to provide details about the APIs of the mentioned products, since they are also involved in the development of applications able to interface with such devices in other contexts. Without a full availability of the proper libraries, it is not trivial to provide a full support to some healthcare sensors.
Another problem is related to connectivity. Outside the mobile and WiFi networks, it is not possible to guarantee any service. This is a critical aspect of the system. In further versions of the app, a mechanism to measure network coverage will be used to alert the user when he/she is going to be disconnected from the system when the service is turned on.
AutomatedSOS, in its first version, will depend strictly by the availability of the hospital ERMs. As mentioned before, this type of resource has allowed the developement of AutomatedSOS. Without it, there is no possibility to alert rescue squads providing information on each raised emergency.

3.8 Software System Attributes

3.8.1 Reliability

Data4Help presents a critical factor related to the data processing on server side. As already motivated, this huge amount of information related to each connected user should be processed in real-time especially when supporting AutomatedSOS users, any failure of both the services could potentially lead to the death of a person. A domino effect, in which a generic fail may cause the blackout of the entire service is unacceptable and must be avoided.

3.8.2 Availability

The software has to be online 24/7, in particular the AutomatedSOS part because its chore is to guarantee a constant control of the users. In this case the system should be kept down for the least time possible when performing maintenance. For minimizing the risk of having interruption of service for patients who need assistance, an availability of at least 99.999% has to be granted. It is an exigent but required value. Since AutomatedSOS relies on Data4Help for collecting data, both the services should have the same availability. Unfortunately this requirement is too expensive for all the users of D4H. It is sufficient to provide such level of availability only for all the users who rely on AutomatedSOS and a value of 99.99% for the remaining ones. This solution can reduce appreciably the costs of maintenance.

3.8.3 Security

Companies and organizations that want to rely on the service must certificate their identity. For the first version of the application, a third party has to provide to the service a valid digital certification, released by a well-known and reliable Certification Authority. This represent an essential security requirement that also provide a strong encrypted communication between the third party and TrackMe. Since TrackMe is also provided with a digital certification, user data sent to the server are encrypted using TLS (Transport Layer Security).

Passwords of both users and third parties, must be hashed and salted on server side.

3.8.4 Maintainability

Especially for Data4Help, the system will deal with an enormous amount of data coming from a wide variety of sensors. Maintainability challenges will concern mainly with the support of new sensor devices appearing in the market that must be supported by the system. The architecture should facilitate the introduction of new products to speed up updated versions of the system.

3.8.5 Portability

The application have to be developed in order to support all the Android devices with an API level higher or equal to 15 (Android 4.0, Ice Cream Sandwich).

4 Formal Analysis Using Alloy

The target of this part is to provide a formal model of the software to be, that can be analyzed and tested. Such study is carried out in the Alloy envinronment. Defining all the aspects of the system is not so trivial in this case, the two services, as said are not independent. ASOS uses a subset of data collected through Data4Help in order to evaluate the user condition, such data are provided by devices registered to D4H, to just name a few. So in order to not complicate too much the model, only the following cases are highlighted and tested to facilitate the comprehension:

Data4Help

- A third party can access a specific user dataset, only after the same user accepts such request.
- A third party is provided with an anonymous dataset (Anonymous Request) if the number of anonymous users involved in an anonymous request is at least 1000.
- A user dataset is composed only with data provided by registered sensor devices

AutomatedSOS

- If an emergency condition is detected the nearest ERM is alerted
- Anomaly conditions can turn into emergencies only if expressed by the user or after the prevention timeout expires. The reverse conversion is not allowed. Emergency conditions can be directly triggered by critical trigger events (criticality = true).
- A user can have a supervisor which is notified each time an anomaly or emergency condition is detected by the system

5 Alloy Code

5.1 Signatures

```
-- SYSTEM ALLOY MODEL

open util/integer
open util/boolean

-- DATA4HELP

sig Time{
        seconds: one Int
}{
        seconds > 0
}

sig Timeout{
        seconds: one Int
}{
        -- the timeout fixed for example to 30 seconds
        seconds ≥ 0 and seconds ≤ 30
}

sig Location {
        --Latitude
        lat: one Int,
        --Longitude
```



```
        long: one Int
}{
        long ≥ -90 and long ≤ 90 and lat ≥ -180 and lat ≤ 180
}

sig User{
        dataset: lone UserDataset,
        pendingList: lone PendingRequests,
        devices: set Device
}

sig Device{
        user: one User,
        healthParameters: set HealthIndicator,
        location: Location lone -> Time,
        supported: Bool one -> Time,
        registered: Bool one -> Time,
        connected: Bool one -> Time
}

sig FiscalCode{
        identify: lone User
}

sig HealthIndicator{}

abstract sig Dataset{}

sig UserDataset extends Dataset{
        subject: one User,
        // device registered to the service may change along the timeline
        sensors: Device some -> Time,
        // health data provided by previous registered devices must be included in the dataset
        healthStatus: HealthIndicator some -> Time,
        userPosition: set Location
}{
        --All the devices associated to the dataset are registered
        all d: Device, t: Time | d in sensors.t iff d.registered.t = True
        --No health data is provided to the system at time t, if the corresponding device is
            ↪ not registered
        all h: HealthIndicator, d: Device, t: Time | ((h in healthStatus.t) and (h ≠ none) and
            ↪  (h in d.healthParameters) and (d in sensors.t)) iff d.registered.t = True
}

sig AnonymousDataset extends Dataset{
        involvedUsers: set User,
        healthStatus: set HealthIndicator
}

sig ThirdParty{
        requests: set Request,
        accessTo: set UserDataset
}

sig Request{
        accepted: Bool one -> Time,
        requestor: one ThirdParty,
        requestObject: one UserDataset
}

sig PendingRequests{
        requests: set Request
}

sig AnonymousRequest{
        accepted: Bool one -> Time,
        involvedUser: one Int,
        requestor: lone ThirdParty,
        requestObject: lone AnonymousDataset,
}
```



```
-- AUTOMATED SOS


sig ERM{
        userDistance: one Int,
        squads: some RescueSquad,
        managedEmergencies: set EmergencyCondition
}{
        #squads ≥ #managedEmergencies
        userDistance > 0
}

sig RescueSquad{
        userDistance: one Int,
        erm: one ERM,
        //it determines if the rescue squad is dispatched for an emergency
        currentEmergency: EmergencyCondition lone -> Time,
        //a rescue squad could be unavailable for other reasons as well as an emergency
        available: Bool one -> Time
}{
        -- a rescue squad is dispatched to an emergency only if available
        all t: Time | (currentEmergency.t ≠ none implies available.t = False) and (available.t
            ↪  = True implies currentEmergency.t = none)
        userDistance > 0
}

sig Supervisor extends User{
        supervised: set User,

}

sig SupervisorRequest{
        receiver: one User,
        accepted: Bool one -> Time
}

-- Notification sent to the Supervisor each time something anomalous happens to the supervised
    ↪  user
sig SupervisorNotification{
        supervisor: one Supervisor,
        reason: one Condition
}


-- An event that triggers an anomaly or emergency condition
sig TriggerEvent{
        indicators: some HealthIndicator,
        constraints: some Int,
        criticality: Bool one -> Time
}

abstract sig Condition{
        user: one User,
        emPosition: lone Location,
        triggerEvents: some TriggerEvent,
}

sig AnomalyCondition extends Condition{
        timeout: one Timeout,
        //it is triggered by the user and determines if the detection is a false positive (
            ↪ false)
        // or an emergency (true)
        isCritical: Bool one -> Time
}{
        -- Anomaly conditions are all those conditions in which the trigger event is NOT
            ↪ critical
        all te: TriggerEvent, t: Time | (te in triggerEvents) and (te.criticality.t = False)
}

sig EmergencyCondition extends Condition{

}{
```



```
        -- Emergency conditions are all those Anomaly conditions in which the trigger event is
            ↪  critical
        all te: TriggerEvent, t: Time | (te in triggerEvents) and (te.criticality.t = True)
}
```

5.2 Facts and Asserts

```
-- DATA4HELP FACTS AND ASSERTS

-- Devices of the same user cannot provide data of the same health indicator
fact noSameIndicator{
        all disj d1, d2: Device | d1.user = d2.user implies d1.healthParameters ≠ d2.
            ↪ healthParameters
}

-- The device can be registered only if connected and supported by the system
fact registeredIfSupported{
        all d: Device | all t: Time | d.registered.t = True implies d.supported.t = True and d
            ↪ .connected.t = True
}

-- One UserDataset cannot be associated to two different users
fact oneUserToOneDataset{
        all disj ds1, ds2: UserDataset | ds1.subject ≠ ds2.subject
}

-- If the anonymous request does not involve at least 1000 users
fact only1000AnonUser{
        all ar: AnonymousRequest | all t: Time | ar.accepted.t = True iff #(ar.involvedUser) ≥
            ↪  1000
}

-- A specific-user request has a unique subject
fact oneRequestToOneUser{
        all r: Request | all disj u1, u2: User | r.requestObject.subject = u1 implies r.
            ↪ requestObject.subject ≠ u2
}

-- a third party accesses to a UserDataset only if authorized
fact onlyAuthorizedAccess{
        all tp: ThirdParty, ds: UserDataset, rq: Request, t: Time |
         (ds in tp.accessTo and ds.subject = rq.requestObject.subject) iff rq.accepted.t =
             ↪ True
}

-- a user dataset is empty of health data if no device has never been registered
assert userDatasetIsEmpty{

        all ud: UserDataset, t,t': Time | (t'.seconds > t.seconds) and (ud.sensors.t' = none)
            ↪ implies (ud.healthStatus.t = none)
}


-- AUTOMATED SOS FACTS AND ASSERTS

-- the nearest ERM to the user position is always alerted
fact theNearestERMisAlerted{

        no erm: ERM | one erm': ERM | all ec: EmergencyCondition | ec in erm'.
            ↪ managedEmergencies iff ((erm' ≠ erm) and (erm'.userDistance > erm.userDistance)
            ↪ )
}

 -- For safety reasons could never happen that an EmergencyCondition is turned into an
     ↪ AnomalyCondition
fact emergencyNotTurnIntoAnomaly{

        all t, t': Time | no c: Condition | (t'.seconds > t.seconds and c.triggerEvents.
            ↪ criticality.t = True) and c.triggerEvents.criticality.t' = False
}
```



```
-- if an anomaly is detected this is turned into an emergency either if the user triggers the
    ↪ emergency or the timeout expires
fact emergencyTriggeredByAnomaly{

        all t: Time, c: Condition, ac: AnomalyCondition, ec: EmergencyCondition | c = ec iff
            ↪ ((ac.timeout.seconds = 0) or (ac.isCritical.t = True))
}

-- no Supervisor can be Supervisor of him/herself
fact notMyOwnSupervisor{
        all sv: Supervisor | sv not in sv.supervised
}

-- each supervisor is notified if an emergency or anomaly occurs to the supervised user
fact notifySupervisor{

        all sv: Supervisor, u: User, c: Condition, sn: SupervisorNotification | (sn.supervisor
            ↪  = sv and sn.reason = c) iff ((u in sv.supervised) and (sn.reason = c))
}

-- An emergency is raised by a critical trigger event, or directly by the user (isCritical
    ↪ true), or when the timeout expires
assert emergencyTriggers{

        all c: Condition | all t,t': Time | all ac: AnomalyCondition | all ec:
            ↪ EmergencyCondition | c = ec iff (t'.seconds > t.seconds) and ((c.triggerEvents.
            ↪ criticality.t = False and c.triggerEvents.criticality.t' = True) and (ac.
            ↪ timeout.seconds = 0) and (ac.isCritical.t' = True))
}
```

5.3 Predicates

```
-- DATA4HELP PREDICATES

-- the device is supported by the system
pred isDeviceSupported[d: Device, t: Time]{
        d.supported.t = True
}


-- the device is connected to the master device
pred isDeviceConnected[d: Device, t: Time]{
        d.connected.t = True
}

-- register a device in the system
pred registerDevice[us: one User, d: Device, h: HealthIndicator, t: Time]{
        isDeviceSupported[d,t]
        isDeviceConnected[d,t]
        d.user =  us
        d.healthParameters = h
        d.registered.t = True
}

-- create a third party request
pred makeRequest[u: User, tp: ThirdParty, r:Request, t: Time, pr, pr': PendingRequests]{
        r.requestor = tp
        r.requestObject.subject = u
        pr'.requests = pr.requests + r
}

-- user accept request and the request is removed from the pending list
pred acceptRequest[u, u': User, r: Request, ds: UserDataset, t: Time, tp, tp': ThirdParty]{
        r.requestObject.subject = u
        r.accepted.t = True
        r.requestObject = ds
        tp'.accessTo = tp.accessTo + ds
        u'.pendingList.requests = u.pendingList.requests - r
}
```



```
-- user rejects request and the request is removed from the pending list
pred rejectRequest[u, u': User, tp: ThirdParty, r: Request, ds: UserDataset, t: Time]{
        r.requestObject.subject = u
        r.accepted.t = False
        r.requestObject = ds
        u'.pendingList.requests = u.pendingList.requests - r
}


-- AUTOMATED SOS PREDICATES

-- the system detects an emergency condition
pred isEmergencyCondition[u: User, te: set TriggerEvent, c: Condition, ec: EmergencyCondition,
    ↪  t: Time]{

        c.user = u
        c.triggerEvents = te
        te.criticality.t = True
        c = ec
}

-- the RescueSquad is available if an intervention is requested
pred isRescueSquadAvailable[erm: ERM, rs: RescueSquad, t: Time]{

        rs in erm.squads
        rs.available.t = True
        rs.currentEmergency.t = none
}

-- the alerted ERM is the nearest
pred isTheNearestERM[nerm: ERM]{

        all ferm: ERM | nerm ≠ ferm and nerm.userDistance < ferm.userDistance
}


-- As soon as an emergency condition is detected the ERM is alerted and a RescueSquad is
    ↪ dispatched
pred ifEmergencyAlertERMAndDispatch[u: User, te: TriggerEvent, c: Condition, ec:
    ↪ EmergencyCondition, nerm, nerm': ERM, t: Time, rs: RescueSquad]{

        isEmergencyCondition[u,te,c,ec,t]

        nerm'.userDistance = nerm.userDistance
        isTheNearestERM[nerm] iff nerm'.managedEmergencies = nerm.managedEmergencies + ec

        isRescueSquadAvailable[nerm',rs,t]
        rs.currentEmergency.t = ec implies rs.available.t = False
}

-- a supervised user accept the request of a Supervisor
pred addSupervisedUser[sv, sv': Supervisor, u: User]{

        sv'.supervised = sv.supervised + u
}

-- Notify the supervisor
pred notifySupervisor[sv: Supervisor, u: User, sn: SupervisorNotification, c: Condition]{

        //precondition
        u in sv.supervised
        sn.reason = c
        c.user = u
        sn.supervisor = sv
}
```

5.4 Testing

```
-- D4H Testing

run isDeviceSupported for 5 but 3 Device, 3 Time, 1 User, 5 HealthIndicator, 1 Location, 3 Int
```



```
run isDeviceConnected for 5 but 3 Device, 3 Time, 1 User, 5 HealthIndicator, 1 Location, 3 Int
run makeRequest for 2 but 2 User, 2 ThirdParty, 2 Request, 2 Time
run acceptRequest for 2 but 1 User, 1 ThirdParty, 2 Request, 2 UserDataset, 2 Time
run rejectRequest for 2 but 1 User, 2 ThirdParty, 2 Request, 1 UserDataset, 2 Time
check userDatasetIsEmpty


-- ASOS Testing
run isEmergencyCondition for 3 but 3 User, 3 TriggerEvent, 3 Condition, 3 Time, 3 Int
run isRescueSquadAvailable for 3 but 1 ERM, 3 RescueSquad, 6 Time, 3 Int
run isTheNearestERM for 2 but 3 ERM
run isEmergencyCondition for 2 but 3 User, 6 TriggerEvent, 3 Condition, 3 EmergencyCondition
run ifEmergencyAlertERMAndDispatch for 2 but 3 User, 3 EmergencyCondition, 1 ERM, 3
    ↪ RescueSquad, 3 Time, 3 Int
run addSupervisedUser for 2 but 3 User, 2 Supervisor
run notifySupervisor for 2 but 1 Supervisor, 2 User, 2 Condition, 2 SupervisorNotification
check emergencyTriggers
```

```
6 commands were executed. The results are:
   #1: Instance found. isDeviceSupported is consistent.
   #2: Instance found. isDeviceConnected is consistent.
   #3: Instance found. makeRequest is consistent.
   #4: Instance found. acceptRequest is consistent.
   #5: Instance found. rejectRequest is consistent.
   #6: No counterexample found. userDatasetIsEmpty may be valid.
```

Figure 17: Consistency Test of D4H model

```
7 commands were executed. The results are:
   #1: Instance found. isEmergencyCondition is consistent.
   #2: Instance found. isRescueSquadAvailable is consistent.
   #3: Instance found. isTheNearestERM is consistent.
   #4: Instance found. ifEmergencyAlertERMAndDispatch is consistent.
   #5: Instance found. addSupervisedUser is consistent.
   #6: Instance found. notifySupervisor is consistent.
   #7: No counterexample found. emergencyTriggers may be valid.
```

Figure 18: Consistency Test of ASOS model

5.5 Data4Help World

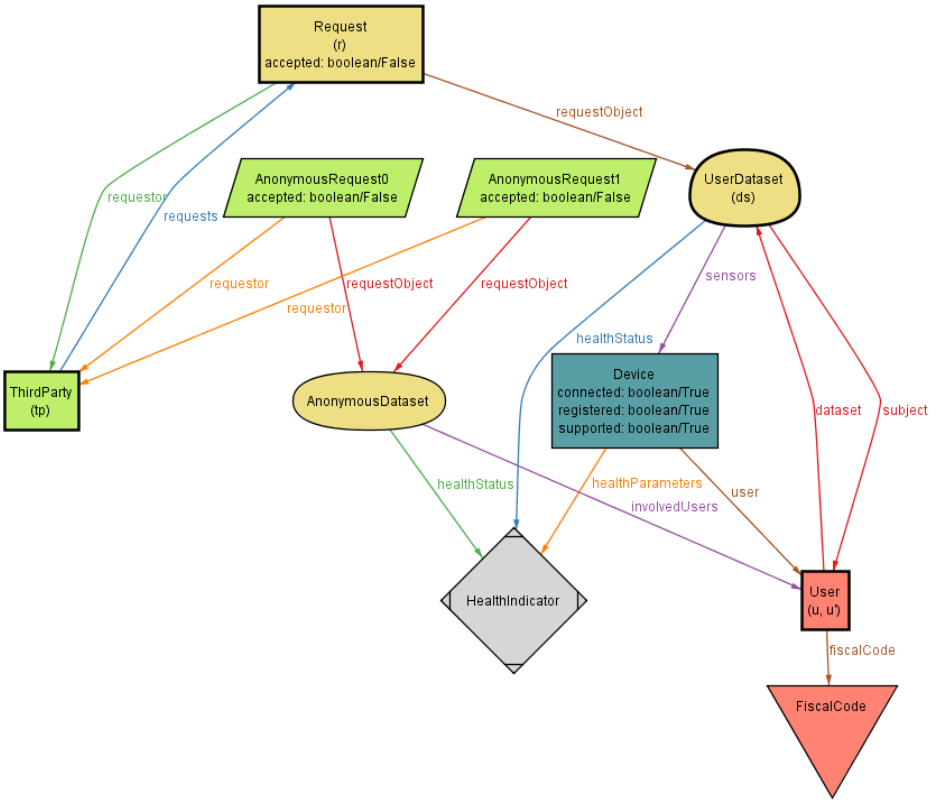


Figure 19: Data4Help world generated after testing

5.6 AutomatedSOS World

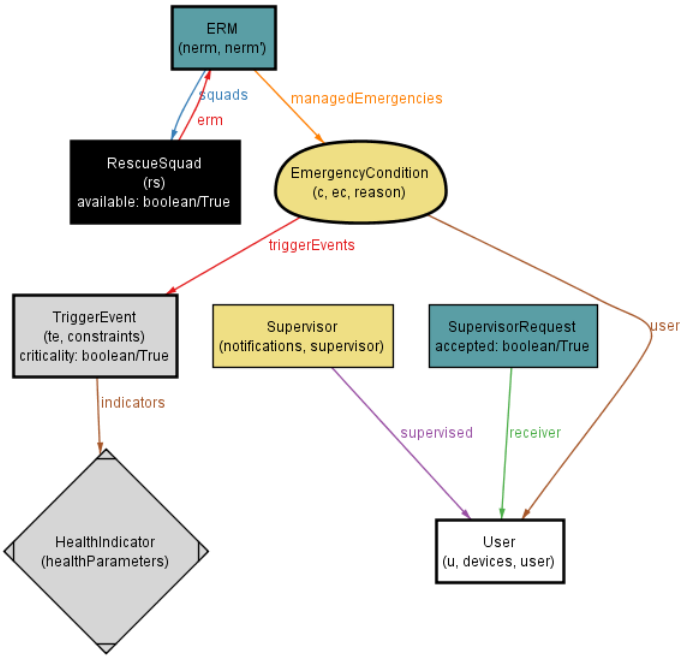


Figure 20: AutomatedSOS world generated after testing

6 Effort Spent

Here is the effort spent by each group member in working at this document.

Barbara Ferretti	**Hours**
Goals	4
Requirements	6
Scenarios and Use Cases	10
Overall Description	3
Mockups	4
Software System Attributes	1

Giorgio Cozza	**Hours**
Goals	4
Requirements	6
Scenarios and Use Cases	8
Overall Description	4
Software System Attributes	2
Alloy Modeling	8

7 Software Tools

- draw.io: UML Modeling
- violet-umleditor-2.1.0 UML Modeling
- GIMP 2
- BluePrint: User Interfaces
- Alloy Analyzer 4.2

8 References

- Kuan Zhang, Xuemin Shen, *"Security and Privacy for Mobile Healthcare Networks", Introduction*
- Android, *"Bluetooh Overview"*, https://developer.android.com/guide/topics/connectivity/bluetooth
- Android, *"Distribution Dashboard"*, https://developer.android.com/about/dashboards/
- Maggi, Zanero, *"Web Application Security"*