POLITECNICO
MILANO 1863

TrackMe

Requirements Analysis and Specification Document

Luca Conterio - 920261
Ibrahim El Shemy - 920174

A.Y. 2018/2019 - Prof. Di Nitto Elisabetta

Contents

1 Introduction

1.1 Purpose

This document represents the `Requirement Analysis and Specification Document` (RASD) for TrackMe software. Main goals of this project are to specify a system that will be able to store and analyze users' health data and whereabouts, to grant third parties to access these data or to subscribe to new data of a specific individual or to retrieve them, and to offer elderly people a rapid assistance based on their health parameters, if needed. At the same time, this document aims at describing the system through functional and nonfunctional requirements, to analyze customers' needs, to show the limits of the software, indicating the typical use cases that can occur.

1.2 Scope

`TrackMe` is a company that wants to develop a software-based service allowing third parties to monitor the location and health status of individuals. Hence, the system has to be composed by two specific services:

- **Data4Help**

  This service supports the registration of individuals who agree that TrackMe acquires their data (through electronic devices such as smartwatches).
  In addition, it supports the registration of third parties that can request:

  - Access to the data of some specific individuals, who can accept/refuse it.
  - Access to anonymized data of groups of individuals. These requests are approved by TrackMe if it is able to properly anonymize the requested data. The request is rejected if it is way too specific.

  As soon as a request for some certain 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.

- **AutomatedSOS**

  This service is oriented to elderly people: monitoring their health status parameters, the system can send to the location of the customer an ambulance when some parameters are below certain thresholds, guaranteeing a reaction time of less than 5 seconds from the time the parameters get lower than the threshold.

1.2.1 Goals

- [**G1**]: Allow visitors to easily register in the system and later to login.
- [**G2**]: Allow users to simply share personal information/health parameters.
- [**G3**]: Allow third parties to access information shared by users.
  - [G3.1]: Allow third parties to access information of specific individuals (through an identifier).
  - [G3.2]: Allow third parties to access anonymized information of groups of individuals.
  - [G3.3]: Allow third parties to subscribe to new information of a specific individual and to receive it.
- [**G4**]: Guarantee users an assistance service when necessary.
  - [G4.1]: Guarantee the elderly users to receive an immediate assistance by an ambulance in case of high risk disease.
  - [G4.2]: Allow users to request on-demand ambulance assistance.
- [**G5**]: Guarantee the preservation of the privacy of the users.

1.3 Definitions, Acronyms and Abbreviations

1.3.1 Definitions

- `User Device`: any compatible device with the TrackMe application, either smartphone or smartwatch.
- `Personal Information`: the collection of infromation provided by a User during the registration process. It Incudes legal information, such as Social Security Number, name, surname, birth date, address, e-mail address, mobile number. Sometimes in the document this expression is abbreviated by `Data`, that also refers to clinical data.

- `Parameters`: everything that the system can monitor periodically through connected user devices. It can be location, heart rate, blood pressure, sleep and activity.
- `Assistance Request`: it is a request formulated by the system and sent to the Ambulance Dispatching System in order to guarantee users the necessary assistance.
- `Individual Access Request`: sometimes referred only as `individual request` or `access request`. It refers to a singular request sent by a Third Party to a User whenever it wants to have access to User's data. It needs the User's approval.
- `Sampling Search`: it refers to a search conducted by a Third Party according to some filters, returning anonymized results for a group of people. In this document it is often called only `sampling`.

1.3.2 Acronyms

- RASD: Requirements Analysis and Specification Document.
- API: Application Programming Interface.
- GPS: Global Positioning System.
- SMS: Short Message Service.
- ETA: Estimated Time Arrival.
- RAPS: Reliable Array of Partioned Service.
- SSN: Social Security Number.

1.3.3 Abbreviations

- [Gn]: n-goal.
- [Rn]: n-requirment.
- App: application.

1.4 Document Structure

This paper refers to the structure suggested by IEEE for RASD documents, with very slight modifications:

1. `Introduction`: the first section is a general description of the system's scope and its goals. It also includes the revision history of the document and its references. Definitions and abbreviations used along the paper are provided too.

2. `Overall Description`: this section includes shared phenomena, requirements and domain assumptions. It also clarifies users' needs.

3. `Specific Requirements`: this section includes all the requirments, both functional and non functional.

4. `Formal Analysis Using Alloy`: it includes the Alloy model of the described system.

5. `Effort Spent`: this section includes information about the hours spent to draft this document.

6. `References`: this section includes references about papers/documents used to support this document.

2 Overall Description

2.1 Product Perspective

The following is an high level UML Class Diagram. It the describes the structure of the system, including external services (such Google Maps and the Ambulance System), user devices and components devoted to specific software functionalities:

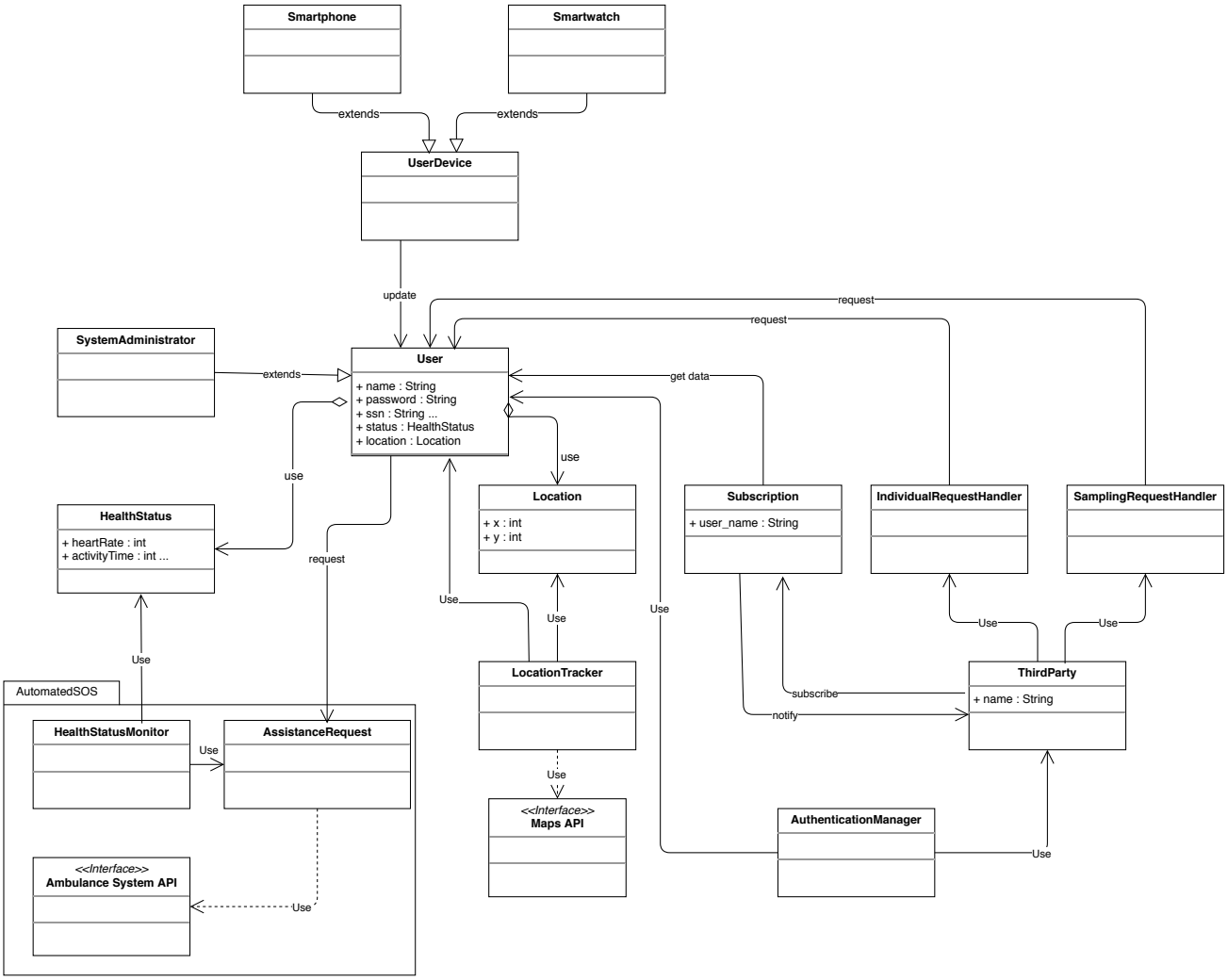


Figure 1: UML Class Diagram

Here below two UML Statechart Diagrams, one for the User and one for the Third Party, are provided to describe the state transition of the two actors according to external events:

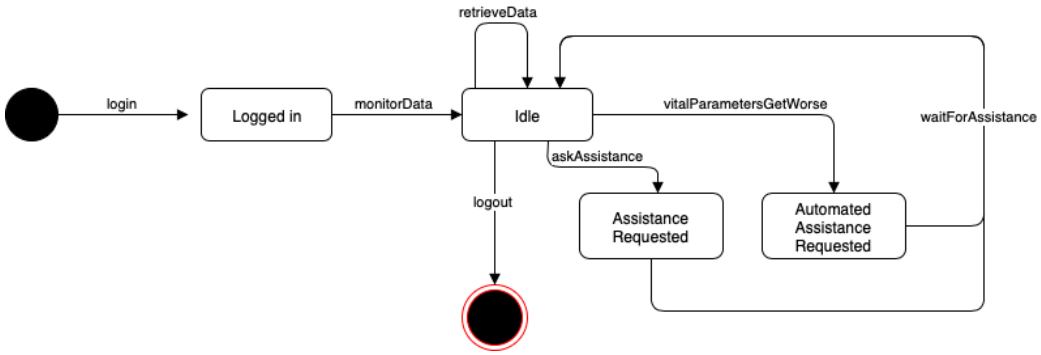


Figure 2: User Statechart Diagram

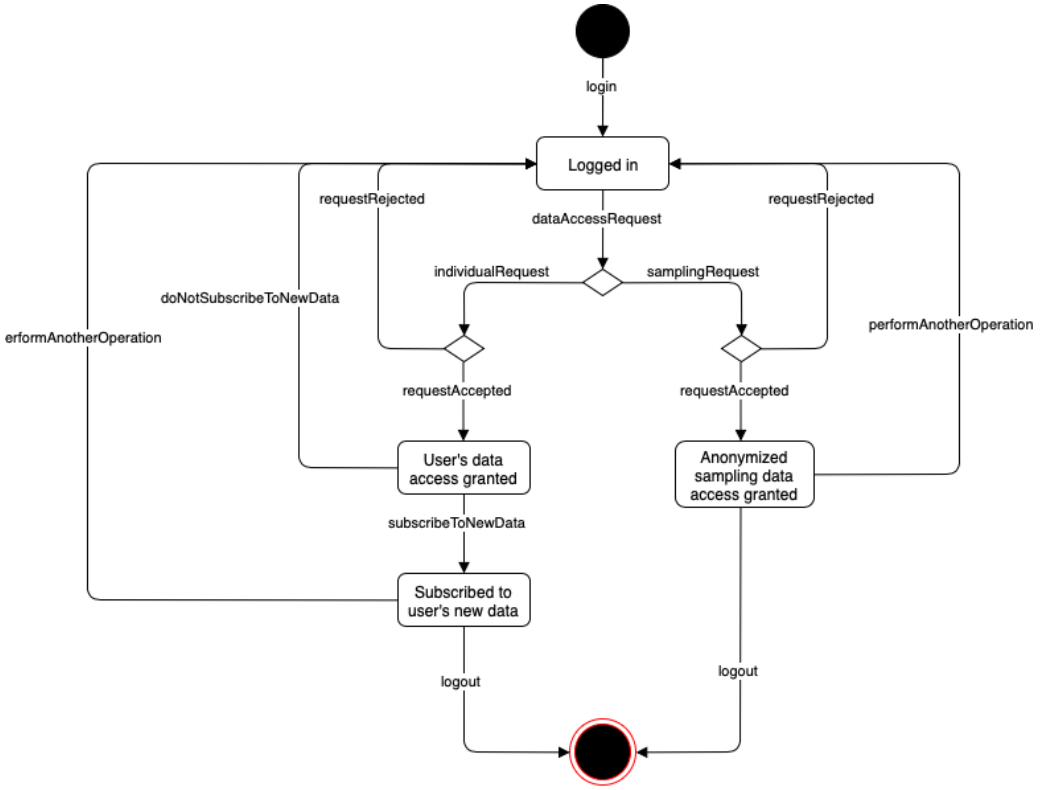


Figure 3: Third Party Statechart Diagram

2.2 Product Functions

According to the goals defined for the current system, we can highlight briefly the main functions of the Application, that can be divided in 4 groups which are the following:

- `Registration and Login`: This functionality lets a Visitor (either a Visitor or a Third Party) to register to TrackMe and therefore be able to Login.
- `Data Monitoring`: Data4Help periodically monitors Users' parameters.
- `Data Access and Update`: Users registered to TrackMe can either visualize their data or update it. Moreover, Third Parties can have access to some Users' data after their approval.
- `Automated Assistance`: This functionality is very crucial for the system. It allows registered Users to receive an assistance when it's necessary.

2.3 User Characteristics

- `Visitor`: a person/third party visiting TrackMe without being registered. He can only proceed to registration in oder to actually use system services, otherwise he can't have access to any service or data.
- `Registered User`: called simply `user` in this document. A person who registered himself to TrackMe, sharing his personal data. He can login to the system through provided credentials to exploit full services.
- `Third Party User`: called simply `third party` in this document. A company or invidual using the platform for some statistical goal or to offer assistance to registered users.
- `System Administrator`: doesn't require to register himself. Makes sure there are no issues in the interaction between users and third parties, guaranteeing a certain level of security. He can report a misuse of the system functionalities.
- `Ambulance Dispatcher`: called simply `dispatcher` in this document. An external individual to the system, whose role is to dispatch an ambulance to assist specific users.

2.4 Domain Assumptions

- [**D1**]: The Social Security Number is be unique.
- [**D2**]: Users are assumed to provide a valid Mobile Number and e-mail.
- [**D3**]: The verification message sent by SMS/e-mail will be certainly received by the User/Third Party.
- [**D4**]: Users' devices are up and running in order to retrieve and process vital parameters.
- [**D5**]: Parameters periodically received by Users' devices are assumed to have a good accuracy.
- [**D6**]: Personal information provided by Users' are assumed to be correct.
- [**D7**]: Users provide correct clinical data (such as blood group, allergies, etc.).
- [**D8**]: Users' devices support the GPS technology.

- [**D9**]: Users' devices support the Mobile Application.
- [**D10**]: Users' devices communicate through the Internet.
- [**D11**]: In case of emergency, all relevant data relative to a specific individual are correctly reported to the Ambulance Dispatching System.
- [**D12**]: The Ambulance Dispatching System is always running and available.
- [**D13**]: The time spent by an ambulance to reach a defined location is as low as possible.

3 Specific Requirements

3.1 External Interface Requirments

The following mockups represent a basic idea of how the Mobile Application and the Web Interface are supposed to look like.

Users can access to complete TrackMe functionalities through the smartphone application, while the smartwatch one will only support a subset of the system's services.

On the other side, TrackMe provides a Web Interface for Third Parties. Here they can exploit all the functionalities at their disposal, such as the request of a sampling according to some parameters.

3.1.1 User Interfaces (UI)

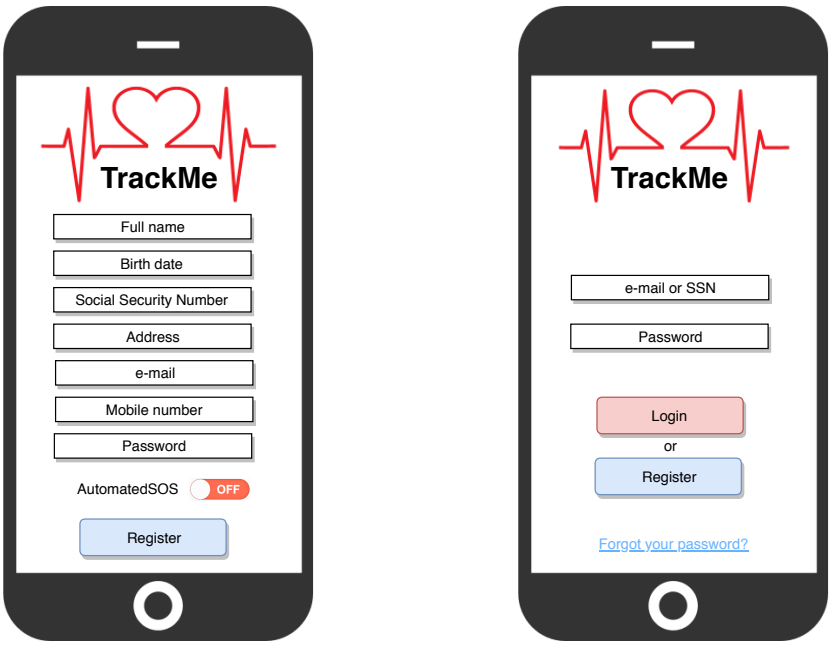


Figure 4: Mobile App registration form and login page.

Figure 5: Mobile App homepage with activity data and AutomatedSOS interface during an ambulance request.

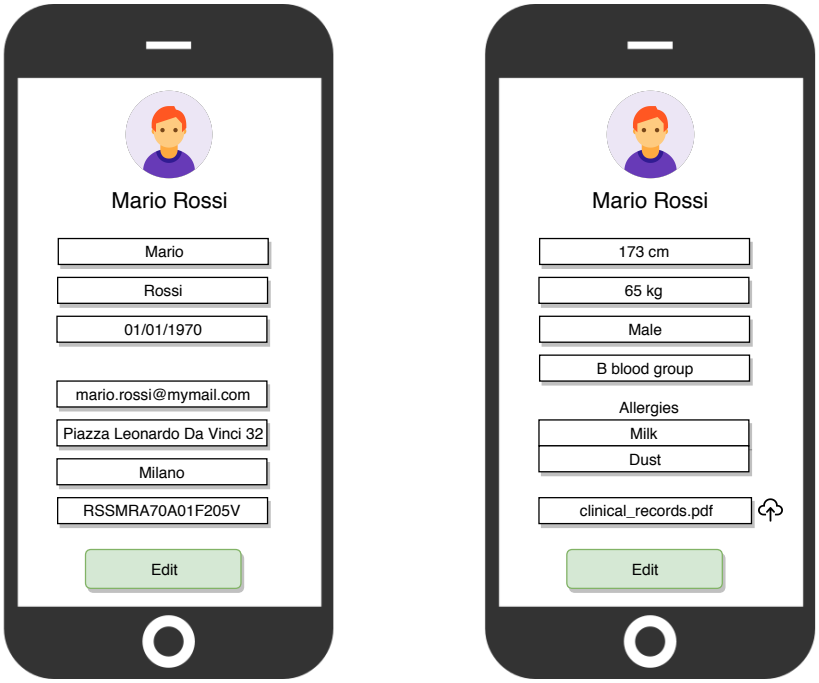


Figure 6: Personal information and clinical data pages.

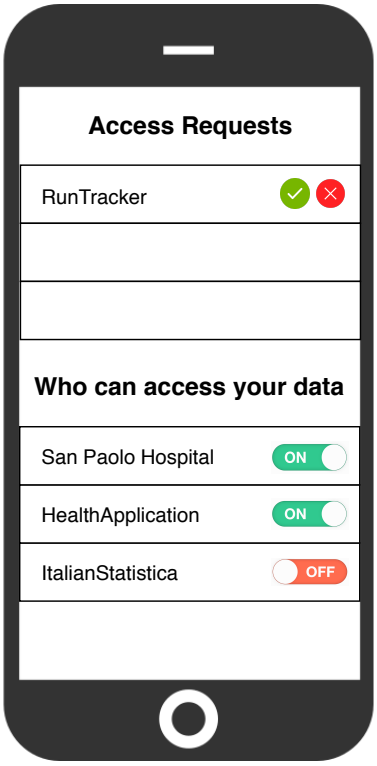


Figure 7: Data access management page.

Figure 8: Smartwatch UI homepage with activity data, heartbeat testing page and AutomatedSOS interface during an ambulance request.

3.1.2 Hardware Interfaces

Since the application must run over the internet, all the hardware shall require to connect network will be an hardware interface for the system, both server and client side.

- Server-side: e.g. Modem, WAN - LAN, Ethernet Cross-Cable.
- Client-side: e.g. Wi-Fi 802.11ac+ antenna, 3G/4G antenna. Moreover, Bluetooth antenna and Body Sensors will be needed to use all services provided by TrackMe.

3.1.3 Software Interfaces

TrackMe will make use of some Application Programming Interfaces to simplify the implementation, since these components are largely used and compatible with the majority of the devices currently on the market:

- `Google Maps API` to have a visual representation of user location and to get it in critical moments, such as AutomatedSOS requests to the Ambulance Dispatching System.
- `Google Fit Sensors API` to read raw sensor data in real time. It allows listing the available sensors on the device on which te application is running and registering a listener on specific sensors to retrieve data automatically.
- `Ambulance Dispatching System API` to perform assistance requests. Actually it could be an automatic phone call to the ambulance contact center, during which an algorithm communicates all relevant data to the interlocutor (e.g. name, age, location etc.).

3.1.4 Communication Interfaces

Since the TLS versions 1.0 and 1.1 will not be more supported, TrackMe must rely on the newer TLS version 1.3, released in 2018, to guarantee the best security possible, at least during the HTTPS connections involving messages carrying credentials or other sensible data.

For other types of connections, e.g. to retrieve periodical data from users' devices, TCP persistent connections can be avoided: even if a packet is lost another one will be generated within a slight amount of time, so UDP non-persistent connections can be used (it is a stream of data). This can also

provide a lower server load, as pointed out later in section 3.6, Performance Requirements.

Furthermore, Bluetooth connectivity is required in order to interface between themselves Users' devices. Indeed, the Mobile Application running on a smartphone needs to communicate with a smartwatch in order to fully exploit TrackMe's functionalities. The newest Bluetooth 5.0 will be preferred, even though older versions of the standard have to be supported, since the majority of devices don't support the latest one yet.

3.2 Scenarios

Scenario 1 - Registration

Marco saw the advertisement of TrackMe and decided to download the mobile application in order to exploit in a useful way his new smartwatch. After opening the new app, he is asked to fill a form with all his personal information, full name, date of birth, mobile number, SSN, etc. To proceed with his registration, after all fields have been filled, Marco clicks on the "Register" button and, as soon as he does, he receives an SMS on his mobile phone confirming the positive outcome of his operation. Marco is now a Registered User of TrackMe: he can login to update his clinical information and benefit of all functionalities of the application.

Scenario 2 - Automatic Ambulance Request

Maria is a 76 years old User of TrackMe, so she can benefit of the AutomatedSOS service. Thanks to this functionality, the system checks periodically her heart rate. During a relax time, Maria's heart rate get so low that her heart risks to stop. Luckily she always wears a smartwatch on her wrist, because she's aware of the utility and importance of TrackMe's monitoring. In fact, as soon as her heart rate gets lower than 30 bpm, AutomatedSOS sends an assistance request to the Ambulance Dispatching System, notifying Maria about the estimated time of arrival of the aid team.

Scenario 3 - On-Demand Ambulance Request

A sport addict user such as John always goes out during the morning to have a run in the park. In the last period he didn't have time to train himself so much as before and now he has more difficulties. Coming back to home, he starts feeling weak. He measure his pressure values and notices that they are lower than their normal level. To speed up the operation and to avoid calling the ambulance service, he logs in his TrackMe account and in the homepage he clicks on "Request assistance". The application confirms that the request has been successfully sent and notifies Jhon about the Estimated Time of Arrival of the ambulance.

Scenario 4 - Search for a User

San Paolo, a hospital located in Milan is looking for a blood donor, whose group is O- in order to help out a patient with a blood transufion. According to medicine, a recipient with blood type O- can only receive blood from

blood group O- only. Therefore, San Paolo, which is registered to TrackMe as a Third Party, is searching for Users whose blood group matches with the patient one. Franco, a registered User to TrackMe and a patient of San Paolo, receives a request in which there is a brief description about the research of a blood donor. Knowing about the importance of such fact and that he has blood group O- (which is the one requested for the blood transfusion), Franco accepts the request.

Scenario 5 - Anonymous Sampling

ItalianStatistica is a big company that performs statistical analysis on the italian territory studying the differences between some geographical areas. For a new analysis on Milan's population, ItalianStatistica decides to retrieve samples from the Data4Help database. After the registration the company can perform anonymous samplings on the people living in Milan. Since asking for how many people with blue eyes and younger than 15 live in the city center results in sampling than 1000 individuals, Data4Help refuses such a request. In order to produce a sampling, ItalianStatistica extends the search to the entire municipality. In this way, a group of more than 1000 individuals is found and the sample with all relevant data is made accessible to the company.

3.3 Functional Requirements

[G1]: Allow visitors to easily register in the system and later to login.

- [D1]: The Social Security Number is be unique.
- [D2] Users are assumed to provide a valid Mobile Number and e-mail.
- [D3]: The verification message sent by SMS/e-mail will be certainly received by the User/Third Party.

- [R1]: The system must allow the Visitor to provide credentials and personal data.
- [R2]: The system must verify the correspondance between the SSN provided by the Visitor and their personal information.
- [R3]: The system must let the Visitor to verify the account with an e-mail/SMS verification.
- [R4]: The system must verify that there are no other registered Users/Third Parties with the same e-mail/SSN.
- [R5]: In order to register successfully a Third Party, the system must oblige it to accept users data privacy conditions.

[G2]: Allow Users to share personal information/health parameters

- [D4]: Users' devices are up and running in order to retrieve and process vital parameters.
- [D5]: Parameters periodically received by Users' devices are assumed to have a good accuracy.
- [D6]: Personal information provided by Users' are assumed to be correct.
- [D7]: Users provide correct clinical data (such as blood group, allergies, etc.).
- [D8]: Users' devices support the GPS technology.
- [D9]: Users' devices support the Mobile Application
- [D10]: Users' devices communicate throught the Internet.

  - [R6]: Users locations must be retrieved by GPS.
  - [R7]: The system must allow the Users to update their personal data.
  - [R8]: The system must allow the Users to upload their medical records (such as blood group).

[G3]: Allow Third Parties to access information shared by the Users

- [G3.1]: Allow Third Parties to access information of specific individuals.
  - [R9]: The system must allow Third Parties to search Users through their SSN's.
  - [R10]: The system must allow Third Parties to send requests in order to access specific data.
  - [R11]: The system must allow Users either to accept or refuse Third Parties' requests.
- [G3.2]: Allow Third Parties to access anonymized information and parameters of groups of individuals.
  - [R12]:The system must allow Third Parties to perform samplings according to some parameters (such as Geographical ones).
  - [R13]: The system must anonymize sampling result data.
  - [R14]: The system must accept sampling if and only if results are related to more than 1000 Users.
- [G3.3]: Allow Third Parties to subscribe to new information of a specific individual and to receive it.
  - [R15]: If a User accepts a requests, the system must allow Third Parties to store and access the previously saved data.
  - [R16]: The system must show Third Parties new data whenever they are available.

[G4]: Guarantee users an assistance service when necessary.

- [G4.1]: Guarantee the elderly users to receive an immediate assistance by an ambulance in case of high risk disease.

  - [D5]: Data periodically received by Users' devices are assumed to have a good accuracy.
  - [D11]: In case of emergency, all relevant data relative to a specific individual are correctly reported to the Ambulance Dispatching System.
  - [D12]: The Ambulance Dispatching System is always running and reliable.
  - [D13]: The time spent by an ambulance to reach a defined location is as low as possible.

  - [R16]: The AutomatedSOS service is automatically activated during the registration process if the Visitor is over 60.
  - [R17]: The Health Status Monitoring System must notify the Ambulance Dispatching System as soon as possible (within 5 seconds).
  - [R18]: When the Ambulance Dispatching System is notified, the Health Status Monitoring System sends all the User's clinical parameters.
  - [R19]: Users' devices must support sensors for retrieving health parameters.

- [G4.2]: Allow users to request on-demand ambulance assistance.
  - [D5]: Parameters periodically received by Users' devices are assumed to have a good accuracy.
  - [D11]: In case of emergency, all relevant data relative to a specific individual are correctly reported to the Ambulance Dispatching System.
  - [D12]: The Ambulance Dispatching System is always running and reliable.
  - [D13]: The time spent by an ambulance to reach a defined location is as low as possible.

  - [R20]: The On-Demand Assistance Request System must notify the Ambulance Dispatching System as soon as possible (within 5 seconds).

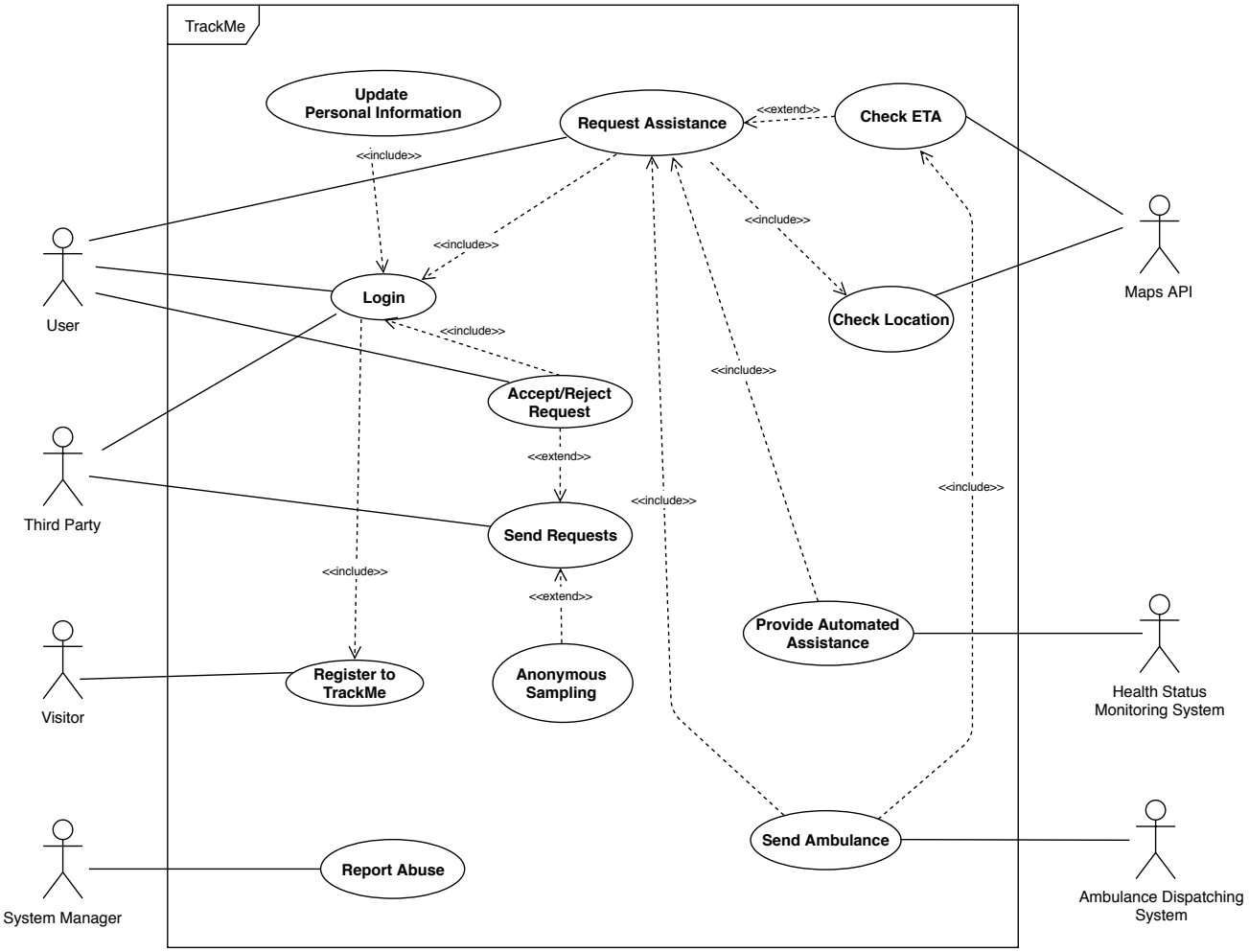
- [R21]: When the Ambulance Dispatching System is notified, the On-Demand Assistance Request System sends all the User's clinical parameters.
- [R22]: The system must avoid abuses of on-demand assistance requests functionality.

[G5]: Guarantee the preservation of the privacy of the Users.

- [R5]: In order to register successfully a Third Party, the system must oblige it to accept data privacy conditions.
- [R14]: The system must accept sampling if and only if results are related to more than 1000 Users.
- [R23]: A Third Party can access data and parameters of a specific User if and only if he/she accepts the request.
- [R24]: The system must allow Users to check the list of third parties accessing their data.
- [R25]: The system must allow the User to enable/disable data access by third parties.
- [R26]: The system must implement some form of accurate access control on databases.

3.4 Use Case Diagram

Here is represented a Use Case diagram representing all functionalities accessible by each actor:



3.5 Use cases

3.5.1 Visitor Registration

NAME	Register to TrackMe
ACTOR	Visitor
GOALS	[G1]
ENTRY CONDITIONS	The Visitor has installed the Application on his/her Mobile device
EVENTS FLOW	1. Fill all mandatory fields in the Registration Form 2. The visitor clicks on "Register" button 3. The system checks data validity 4. The system sends an SMS as confirmation 5. The system saves Users' data
EXIT CONDITIONS	The Vistor has successfully registered to TrackMe
EXCEPTIONS	1. The User is already registered 2. The e-mail is already registered 3. The Mobile Number is already registered 4. The SSN provided is invalid 5. Some mandatory fields are not filled

3.5.2 User Login

NAME	User Login
ACTOR	User
GOALS	[G1]
ENTRY CONDITIONS	The User is in the Login page
EVENTS FLOW	1. The User enters e-mail or SSN 2. The User enters the password 3. The User clicks on "Login" button 4. The system checks the correctness of the credentials
EXIT CONDITIONS	The User has successfully logged in
EXCEPTIONS	1. The User is not registered 2. The e-mail is wrong 3. The SSN is wrong 4. The password is wrong 5. Some mandatory fields are not filled

3.5.3 Third Party Login

NAME	Third Party Login
ACTOR	Third Party
GOALS	[G1]
ENTRY CONDITION	The Third Party is in the Login page
EVENTS FLOW	1. The Third Party enters e-mail or SSN 2. The Third Party enters the password 3. The Third Party clicks on "Login" button 4. The Third Party systems checks the correctness of the credentials
EXIT CONDITIONS	The Third Party has successfully logged in
EXCEPTIONS	1. The Third Party is not registered 2. The e-mail is wrong 3. The SSN is wrong 4. The password is wrong 5. Some mandatory fields are not filled

3.5.4 Update Personal Information

NAME	Update Personal Information
ACTOR	User
GOALS	[G2]
ENTRY CONDITION	The User is in the personal data page after Login
EVENTS FLOW	1. The User clicks on "Edit" button 2. The User changes his/her personal information 3. The User confirms the changes 4. The system checks that the inserted values are appropriate
EXIT CONDITIONS	The User has successfully edited his/her personal information
EXCEPTIONS	1. The User edited one or more fields in an inappropriate way (e.g. added a non-existing address)

3.5.5 Provide Automated Assistance

NAME	Provide Automated Assistance
ACTOR	Health Status Monitoring System
GOALS	[G4.1]
ENTRY CONDITION	Some User's parameters are below a certain threshold
EVENTS FLOW	1. The system detects that some parameters are below a threshold 2. The system requests an ambulance to assist the user 3. The system retrieves user's location and data and forwards them to the Ambulance Dispatching System 4. The Ambulance System confirms the availability 5. The Ambulance System communicates the ETA
EXIT CONDITIONS	An ambulance has been sent to the user's location
EXCEPTIONS	GPS is not active on User's device

3.5.6 On-Demand Assistance Request

NAME	Request Assistance
ACTOR	User
GOALS	[G4.2]
ENTRY CONDITION	The User is in the HomePage after Login
EVENTS FLOW	1. The User clicks on "Request Assistance" button 2. The User confirms the operation when it is asked 3. The system requests an ambulance to assist the user 4. The system retrieves user's location and data and forwards them to the Ambulance Dispatching System 5. The Ambulance System confirms the availability 6. The Ambulance System communicates the ETA
EXIT CONDITIONS	The User has successfully requested Assistance and an ambulance has been sent to the user's location
EXCEPTIONS	GPS is not active on User's device
SPECIAL REQ.	1. The request must be processed in less than 5 seconds.

3.5.7 Report Abuse

NAME	Report Abuse
ACTOR	System Manager
GOALS	[G4.2]
ENTRY CONDITION	The System Manager notices an abuse of the assistance request system
EVENTS FLOW	1. The System Manager signals the User that is misusing the service 2. The system notifies the User that he/she will no more be able to exploit the on-demand assistance request service 3. The system inserts the User in the list of the users that don't have access to such a service
EXIT CONDITIONS	The User is no more able to use the on-demand assistance request service
EXCEPTIONS	None

3.5.8 Accept Access Request

NAME	Accept Request
ACTOR	User
GOALS	[G3.1]
ENTRY CONDITION	The User has received an access request from a Third Party
EVENTS FLOW	1. The User receives a request from a Third Party 2. The User clicks on "Accept" button
EXIT CONDITIONS	The User has successfully accepted the request and the Third Party can access to his/her data
EXCEPTIONS	None

3.5.9 Reject Access Request

NAME	Reject Request
ACTOR	User
GOALS	[G3.1], [G5]
ENTRY CONDITION	The User has received an access request from a Third Party
EVENTS FLOW	1. The User receives a request from Third Party 2. The User clicks on "Reject" button
EXIT CONDITIONS	The User has successfully rejected the request and the Third Party can not access to his/her data
EXCEPTIONS	None

3.5.10 Send Access Request

NAME	Send Request
ACTOR	Third Party
GOALS	[G3.1]
ENTRY CONDITION	The Third Party must be in the Homepage after Login
EVENTS FLOW	1. Third party searches a User through his/her SSN 2. Third party sends a request to the User 3. Third party waits for User's response
EXIT CONDITIONS	The User accepted/rejected the request
EXCEPTIONS	1. The User's SSN is invalid 2. The person with the specified SSN is not registered

3.5.11 Subscription to User's Data

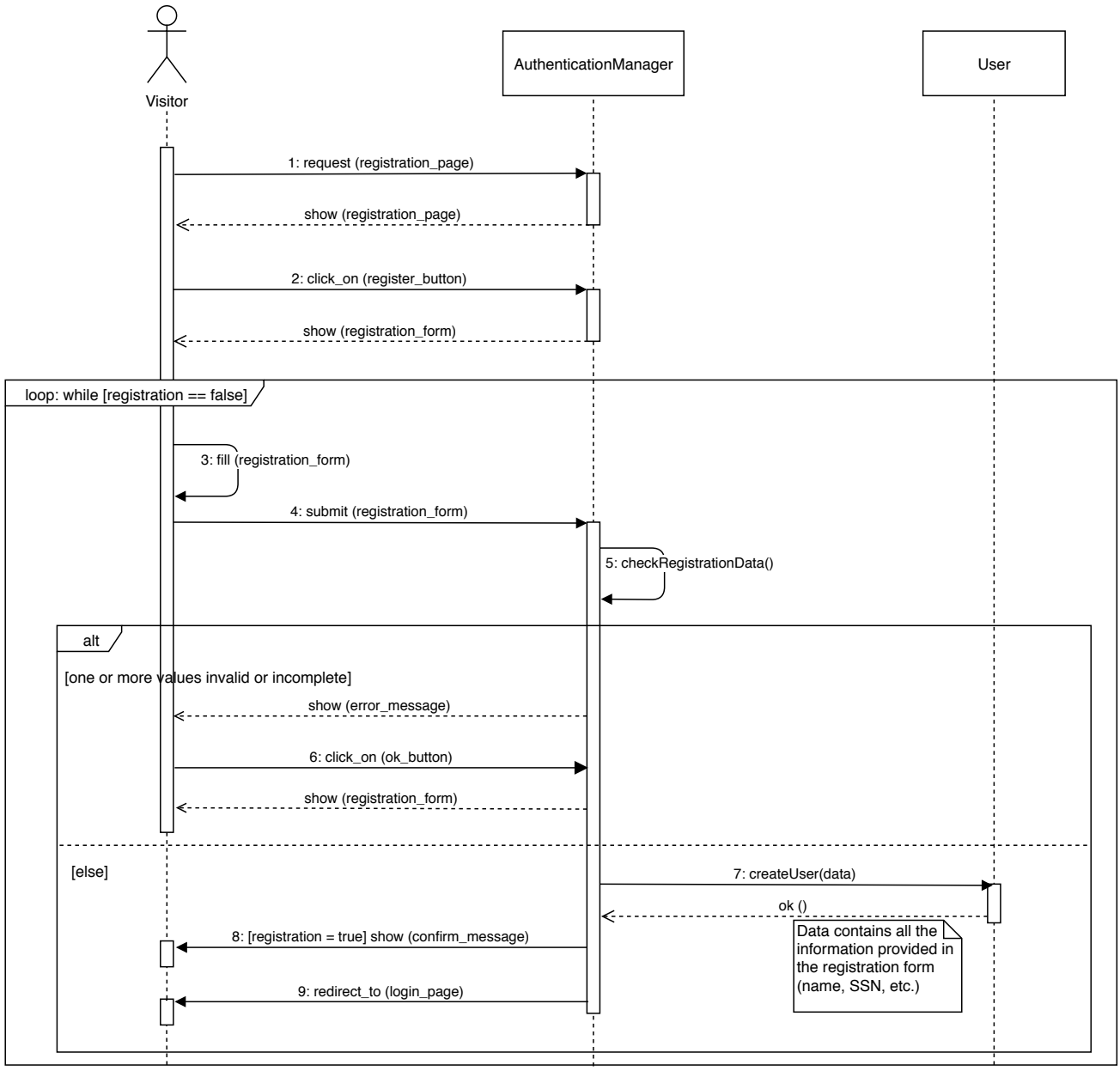
NAME	Subscription to User's Data
ACTOR	Third Party
GOALS	[G3.3]
ENTRY CONDITION	The User has accepted Third Party's access request
EVENTS FLOW	1. Third party clicks on "Subscribe to User's new data" button 2. Third party confirms the subscription
EXIT CONDITIONS	The Third Party has successfully subscripted to new User's data
EXCEPTIONS	None

3.5.12 Anonymous Sampling

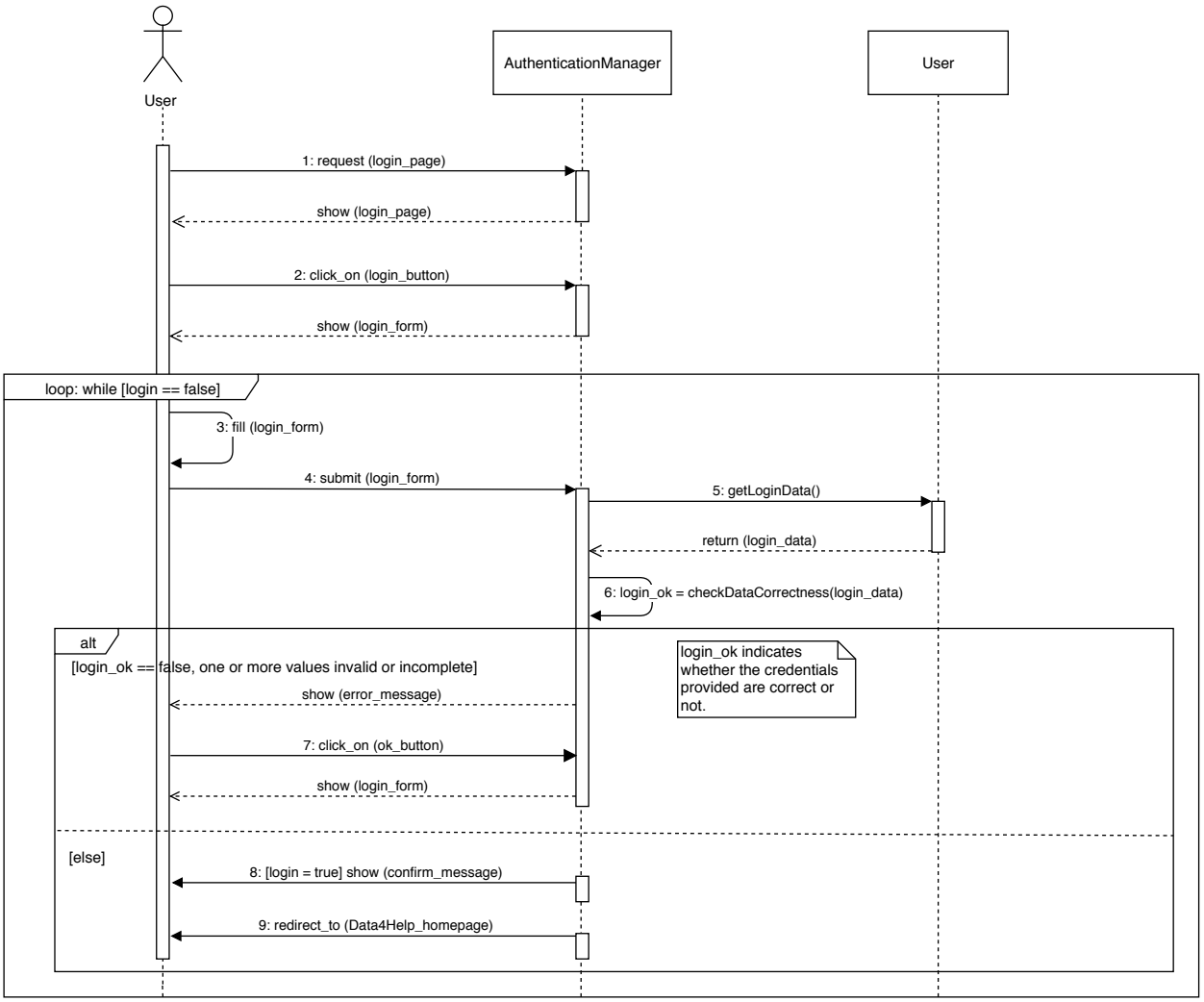
NAME	Anonymous Sampling
ACTOR	Third Party
GOALS	[G3.2], [G5]
ENTRY CONDITION	The Third Party wants to collect sample information and it is in the HomePage after Login
EVENTS FLOW	1. Third party conducts a sample search 2. Third party saves the collected information
EXIT CONDITIONS	The Third Party can access the sampling's data
EXCEPTIONS	The number of Users matching with the Third Party's sample search is lower than 1000.

3.6 Sequence Diagrams

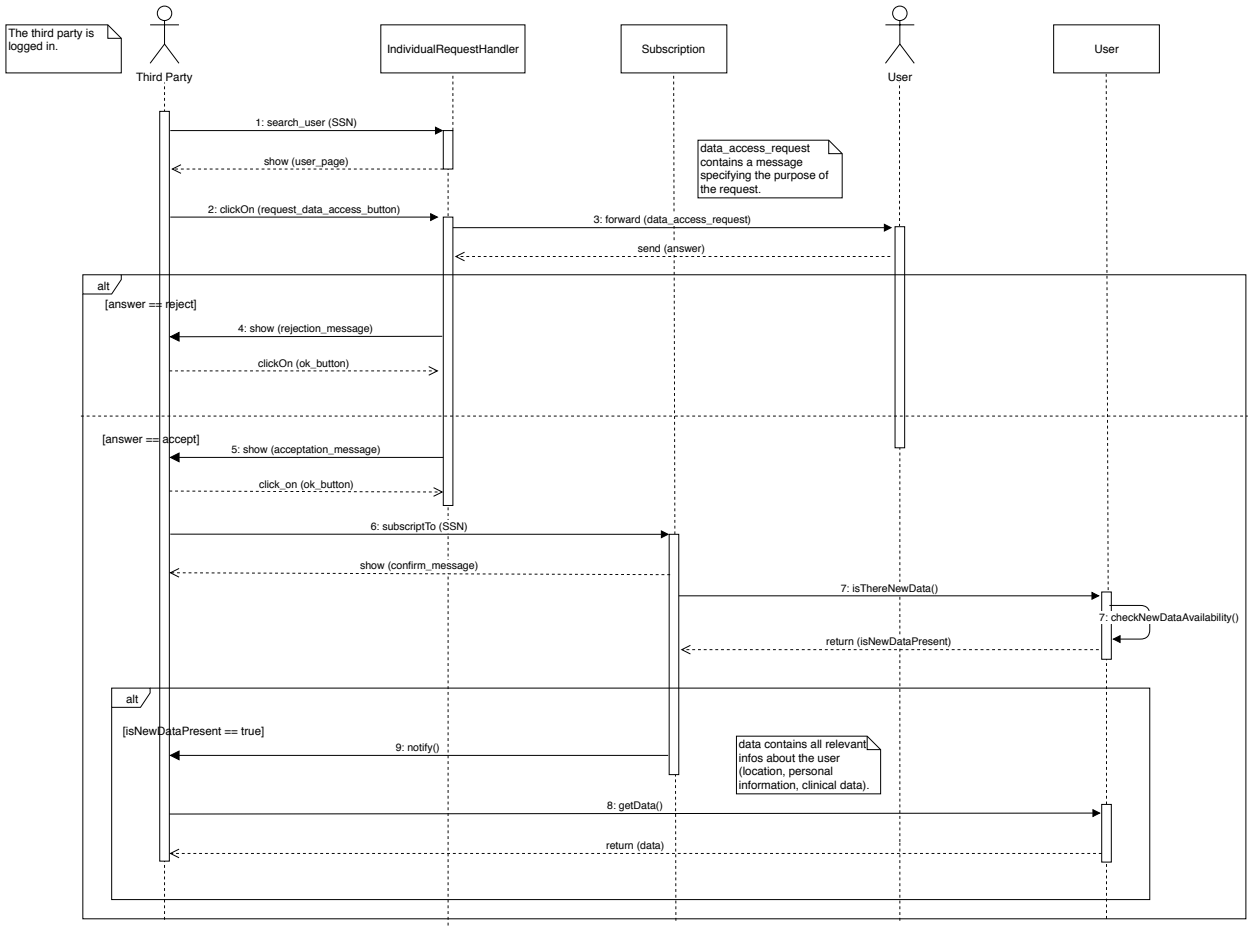
3.6.1 Visitor Registration



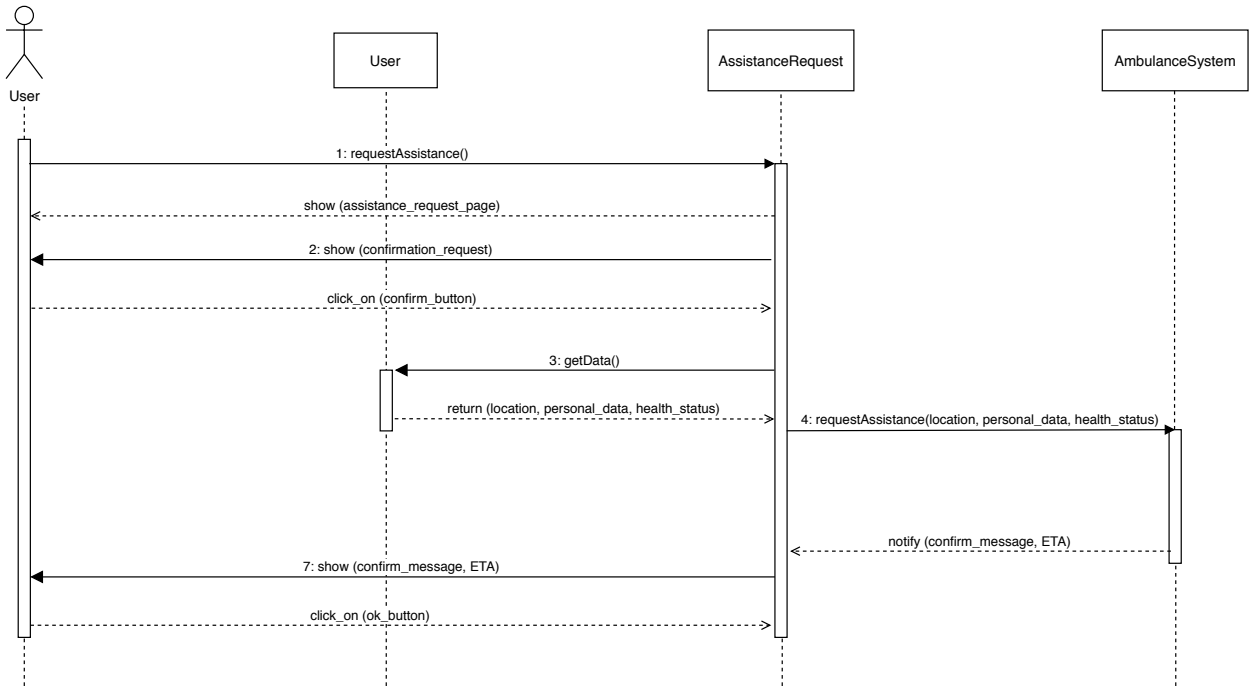
3.6.2 Login


User
AuthenticationManager
User
1: request (login_page)
show (login_page)
2: click_on (login_button)
show (login_form)
loop: while [login == false]
3: fill (login_form)
4: submit (login_form)
5: getLoginData()
return (login_data)
6: login_ok = checkDataCorrectness(login_data)
alt
[login_ok == false, one or more values invalid or incomplete]
login_ok indicates whether the credentials provided are correct or not.
show (error_message)
7: click_on (ok_button)
show (login_form)
[else]
8: [login = true] show (confirm_message)
9: redirect_to (Data4Help_homepage)

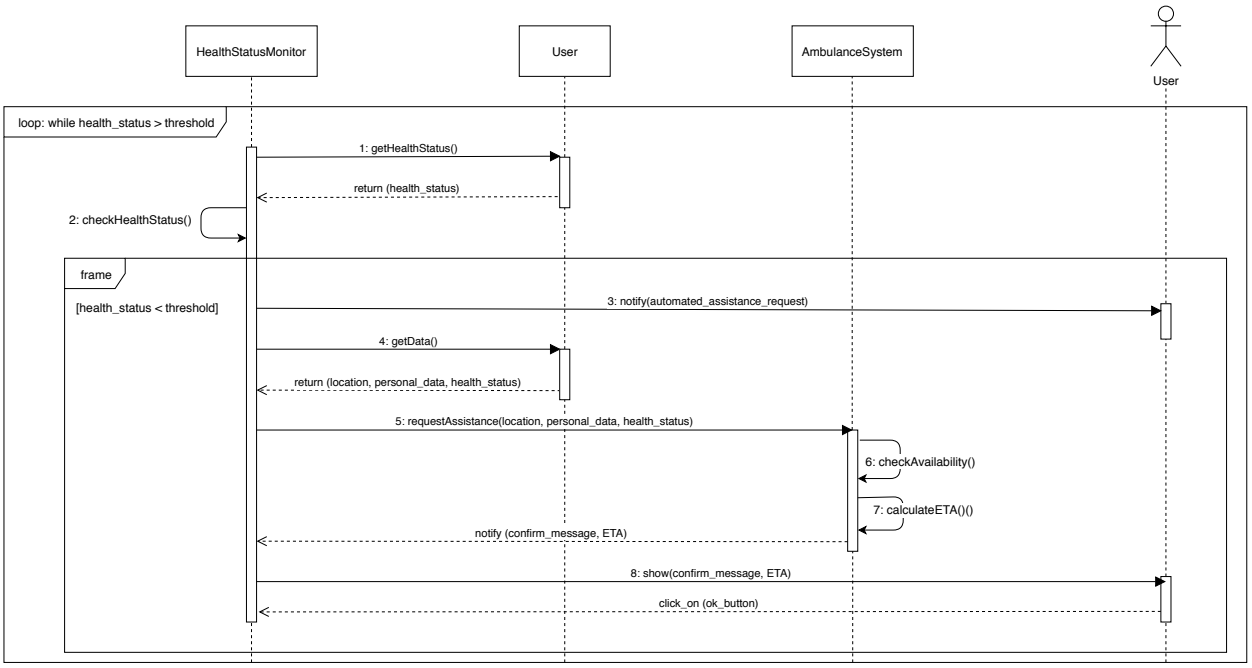
3.6.3 User Data Access Request and Subscription



3.6.4 On-Demand Assistance Request



3.6.5 Automated Assistance Request



3.7 Performance Requirments

The system must be able to handle a huge quantity of requests simultaneously throughout the day, responding to users' necessities in the shortest time possible. In order to improve the perfomance of the system, since data are received in a discrete way (e.g. monitored every 5 seconds), TrackMe relies on a UDP non-persistent connection.
For what concerns the AutomatedSOS service, it is required that each ambulance request is generated and sent to the Ambulance Dispatching System within 5 seconds from the moment in which the vital parameters of an elder user get below the fixed threshold.

3.8 Desgin Constraints

3.8.1 Hardware Limitations

- `Mobile App`: iOS/Android, Internet Connection, GPS.
- `watchOS/WearOS`: smartwatches linked to mobile devices or equipped with GPS and equipped with hearbeat/pressure sensors.

- `Web App`: browser (e.g. Google Chrome / Safari) able to retrieve users' location.

3.9 Software System Attributes

3.9.1 Reliability

The system must guarantee a 24/7 service. This requirment should not have any sort of deviation (especially concerning AutomatedSOS).

3.9.2 Availability

The system must fullfil all the requests whenever needed (e.g. get/update personal information, request for medical assistance). Only a small percentage of the total requests is admissible (less than 0.01% of the total amount of requests). The system relies on a RAPS architecture, to better guarantee availability. The whole system is partitioned in nodes, each one managing a single service, in which data is made redudant in different servers. In this way, the malfuction of a service will not cause a service breakdown, but only a decrease of performance.

3.9.3 Security

Despite the fact that personal information is stored, the system guarantees not to spread them outside, and third parties are obliged to be confirmed with a privacy policy.

3.9.4 Maintainability

Enforced by the usage of specific design patterns (e.g. third party subscription is notified through an Observer Pattern whenever new data is generated by the users) and the provided documentation.

3.9.5 Scalability

Relying on a RAPS architecture, it is easier to enlarge the structure of the system, since it is possible to invest only on those services that need to handle the higher amount of requests or the ones where the number of users is relevant.

4 Formal Analysis Using Alloy

4.1 Purpose of the Model

Through this model we want to formalize some principal aspects of the TrackMe system:

- All the users that receive an automated assistance are necessarily subscribed to the AutomatedSOS service. Furthermore, users cannot receive multiple assistances at the same time and an automatic assistance is generated whenever his/her health status gets below the critical threshold (here represented by 0). All the users with an age grater or equal to 60 (here represented by 6) are automatically registered to AutomatedSOS, but this doesn't preclude younger users to subscribe to the service too. If an automated assistance request is generated, then it must be sent to the Ambulance System within 5 seconds.

- All the third parties in order to have access to some users' data, have to send a request to the specific user, that can be either accepted or requested. Only if the user accepts the request, then the third party can access all his/her data. If the access is granted, then the third party has the possibility to subscribe to user's new data (but it is not necessary).

- Third Parties have the possibility to do sample searches according to some filters. TrackMe validates only those searches that result in more than 1000 users (here represented by 1).

4.2 Alloy Model

```
open util/integer
open util/boolean

---------------- SIGNATURES ----------------

-- The user signature: here are represented only the main attributes
-- (SSN, age, health status and the list of third parties that can access to user's data)
sig User {
    ssn: one Ssn,          -- unique identifier
    age: one Int,
    sos: one Bool,             -- 1 if the User is registered to AutomatedSOS, 0 otherwise
    healthStatus: one Int,    -- if status is negative, the User is automatically assisted by ambulance
    partyList: set ThirdParty -- the list of third parties that can access to user's data
}{
    age > 0                -- user's age is a positive number
}

sig ThirdParty {
    ssn: one Ssn,                       -- unique identifier
    accessibleUsers: set User,                  -- the list of the users accessible by the third party
    subbedTo: set User,                 -- the of the users to which the third party is subscribed to
    samplings: set AnonymousSampling   -- the list of sampling requested by the third party
}

sig Ssn {}      -- unique identifier for users and third parties

sig AssistanceRequest {      -- AutomatedSOS assistance request
    user: one User,            -- the user for which the assistance request is formulated
    elapsedTime: one Int,   -- the amount of time passed between the moment in which the request
                        -- is formulated and the moment in which the request is sent to the ambulance system
    requestSent: one Bool   -- boolean indicating whether the request has already been sent to the ambulance system
}{
    elapsedTime >= 0        -- the elapsed time is a positive integer
}

one sig AutomatedSOS {                -- the AutomatedSOS service entity
    monitoredPeople: set User,      -- the list of automatically assisted people
    assistances: set AssistanceRequest -- the list of all formulated assistance requests
}

sig IndividualAccessRequest { -- a request formulated by a third party to have access to a user's data
    user: User,             -- the user that a third party wants to have access to
    thirdParty: ThirdParty, -- the third party requesting access for user's data
    found: one Bool,           -- boolean indicating whether a user has been found or not
    accepted: one Bool        -- boolean indicating whether the user has accepted the ThirdParty's request or not
}

sig AnonymousSampling { -- an anonymous sampling request formulated by a third party according to some filters
    results: set User,      -- the list of users
    valid: one Bool            -- boolean indicating whether the samplig can be accomplished or not
}
```

```
---------- FACTS ----------

fact SSNAreUnique {
    -- all SSN are unique, both for users and third parties
    (no disjoint user1, user2: User | user1.ssn = user2.ssn) and
    (no u: User, tp: ThirdParty | u.ssn = tp.ssn)
}

fact thirdPartySubscribedToUser {
    all u: User, tp: ThirdParty |
        -- a third party can subscribe to a user of and only if that user is accessible to the third party
        ((u in tp.accessibleUsers) or (u in tp.accessibleUsers and u in tp.subbedTo)) and
         -- if the user is accessible to a third party, than the user has that third party in his third parties list
        (u in tp.accessibleUsers iff tp in u.partyList)
}


fact userAccessibleFromThirdPartyOnlyIfRequestAccepted {
    -- a user is accessible to a third party if and only if exists a access request
    -- from that third party to that user that has been accepted
    all u: User, tp: ThirdParty | u in tp.accessibleUsers iff
        (one r: IndividualAccessRequest | r.user.ssn = u.ssn and r.thirdParty.ssn = tp.ssn and r.accepted = True)
}

fact no2IndividualAccessRequests {
    -- for each pair (user;, third party) there's only one data access request
    no disjoint r1, r2: IndividualAccessRequest | r1.user = r2.user and r1.thirdParty = r2.thirdParty
}

fact noRequestsWithoutExistingThirdPArty {
    -- can't exist an access request from a non existing third party
    all r: IndividualAccessRequest |
        (one tp: ThirdParty | r.thirdParty.ssn = tp.ssn)
}

fact requestAcceptedIffUserExists {
    -- can't exists an accepted access request if the involved user doesn't exist
    all r: IndividualAccessRequest |
        ((r.found = False and r.accepted = False) or
        (r.found = True and  r.accepted = False) or
        (r.found = True and r.accepted = True))
        and
        (r.found = True iff (one u: User | u.ssn = r.user.ssn))
}

fact noSamplingWithoutThirdParty {
    all s: AnonymousSampling | s in ThirdParty.samplings
}

fact samplingRequestValidity {
    -- an anonymous sampling is valid (accepted by TrackMe if and only if
    -- it involves more than 1000 users (1000 is represented by 1 in this model)
    all s: AnonymousSampling |
        #s.results >= 1 implies s.valid = True
}
```

```
-- Automated SOS
fact automatedSOSMember {
	-- all people monitored by AutomatedSOS must be subscribed to it, hence the sos boolean must be set to True
	(all user: User |
		(user in AutomatedSOS.monitoredPeople iff user.sos = True) and
		-- if the user's age is greater or equal that 60 years (indicated in this model by value 6),
		-- then he/she is automatically subscribed to AutomatedSOS
		(user.age >= 6 implies user.sos = True))
}

fact automatedAssistanceRequest {
	-- a user is automatically assisted if and only he/she is subscribed to AutomatedSOS
	-- and his/her health status is negative (which means he/she is need of assistance)
	all u: User | (u.sos = True and u.healthStatus < 0) iff (one a : AssistanceRequest | a.user.ssn = u.ssn)
}

fact assistanceRequestSentWithin5Seconds {
	all a: AssistanceRequest |
		-- if the request of assistance has not been sent, then the elapsed time can not be greter that 5 seconds
		(a.requestSent = False implies a.elapsedTime < 5) and
		-- if the elapsed time is greater or equal to 5 seconds, then it means that a request has been sent
		(a.elapsedTime >= 5 iff a.requestSent = True)
}

fact noAssistanceForOKPeople {
	-- there is no assistance request sent for healthy people
	no a: AssistanceRequest | a.user.healthStatus >= 0
}

fact onlyOneAssistancePerUser {
		-- there can be only one exclusive assistance for each user
		no disjoint a1, a2: AssistanceRequest | a1.user = a2.user
}

fact allAssistancesAreInAutomatedSOS {
	 all a: AssistanceRequest | a in AutomatedSOS.assistances
}
```

```
---------- ASSERTIONS ----------

assert uniquenessOfAssistanceRequestInAutomatedSOS {
    all u: User |
        -- if a user's age is greater or equal than 60 and is subcribed to AutomatedSOS, then he/she is monitored by AutomatedSOS
        (u.age >= 6 and u.sos = True implies u in AutomatedSOS.monitoredPeople)  and
        -- a request is sent to a user if and only if his health status is negative and monitored by AutomatedSOS
        ((one a: AssistanceRequest | a.user.ssn = u.ssn) iff (u.healthStatus < 0 and u in AutomatedSOS.monitoredPeople))
}

check uniquenessOfAssistanceRequestInAutomatedSOS

assert userAccessibleOnlyIfRequestAccepted {
    all u: User, tp: ThirdParty |
        -- if a user is accessible by a third party, then there exists only one request sent by that third party and accepted by that user sent
        (u in tp.accessibleUsers implies (one r: IndividualAccessRequest | r.user = u and r.thirdParty = tp and r.accepted = True))
        and
        -- if a third party is subscribed to a user, then it implies also that that user is
        -- accessible and there exists only one request sent by that third party and accepted by that user sent
        (u in tp.subbedTo implies (u in tp.accessibleUsers and
            (one r: IndividualAccessRequest | r.user = u and r.thirdParty = tp and r.accepted = True)))
}
check userAccessibleOnlyIfRequestAccepted

assert noMoreThan5SecondsToSendAssistance {
    -- if the elapsed time is greater or equal that 5 seconds, then it means that a request has been sent
    all a: AssistanceRequest | a.elapsedTime >= 5 implies a.requestSent = True
}
check noMoreThan5SecondsToSendAssistance
```

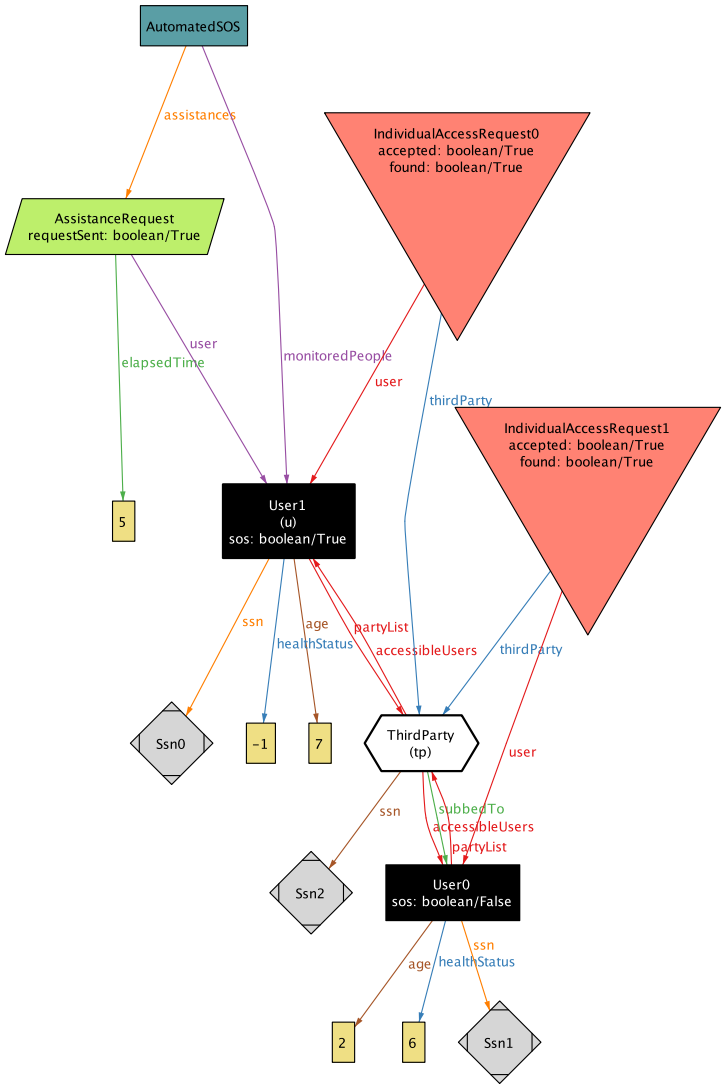
```
---------- PREDICATES ----------

pred createUser[u: User, s: Ssn, st: Int] {
    u.ssn = s
    u.healthStatus = st
}

pred show[tp: ThirdParty] {
    #User > 1
    #AutomatedSOS.monitoredPeople > 0
    some u: User | u.healthStatus < 0
    some u: User | u in tp.accessibleUsers
    some r: IndividualAccessRequest | r.thirdParty.ssn = tp.ssn
}

run show
```

```
Executing "Check uniquenessOfAssistanceRequestInAutomatedSOS"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  5162 vars. 291 primary vars. 14434 clauses. 43ms.
  No counterexample found. Assertion may be valid. 45ms.

Executing "Check userAccessibleOnlyIfRequestAccepted"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  4809 vars. 294 primary vars. 13054 clauses. 43ms.
  No counterexample found. Assertion may be valid. 31ms.

Executing "Check noMoreThan5SecondsToSendAssistance"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  4898 vars. 291 primary vars. 13469 clauses. 54ms.
  No counterexample found. Assertion may be valid. 43ms.

Executing "Run show"
  Solver=sat4j Bitwidth=4 MaxSeq=4 SkolemDepth=1 Symmetry=20
  5042 vars. 300 primary vars. 13774 clauses. 49ms.
  Instance found. Predicate is consistent. 55ms.

4 commands were executed. The results are:
  #1: No counterexample found. uniquenessOfAssistanceRequestInAutomatedSOS may be valid.
  #2: No counterexample found. userAccessibleOnlyIfRequestAccepted may be valid.
  #3: No counterexample found. noMoreThan5SecondsToSendAssistance may be valid.
  #4: Instance found. show is consistent.
```

4.3 Generated World

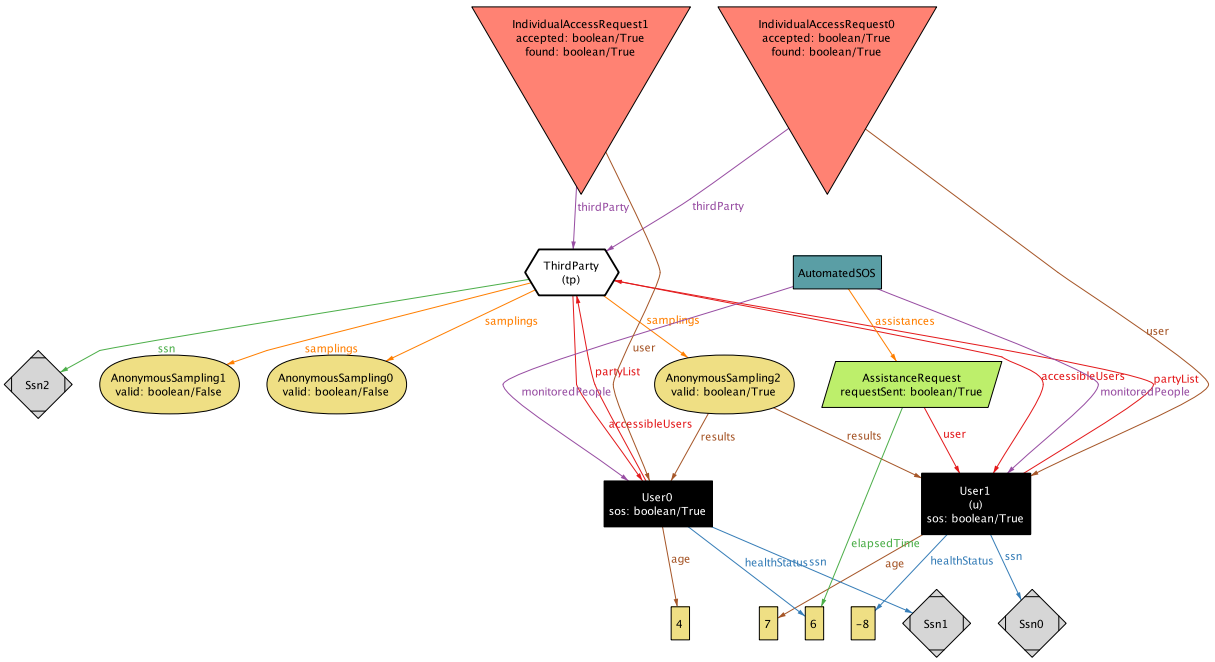
4.3.1 First Instance

As can be seen from this instance there are two different users, but only User1 is monitored by AutomatedSOS, since his age is equal to 7 (¿= 6). In addition his health status is negative and there exist an AssistanceRequest that has already been sent (the elapsed time equals to 5). Moreover, there's a Third Party that has access to both of the users' data, but it is subscribed only to User0. This situation is possible thanks to the two accepted IndividualAccessRequest sent by the Third Party to both the users.



4.3.2 Second Instance

The illustrated situation is similar to the previous one, even though both users are monitored by AutomatedSOS (one has age greater than 6, the other one doesn't). In addition three different sampling searches have been generated. AnonymousSampling2 is valid, since it contains in its results list both User0 and User1 (results ¿ 1). The other two samplings are not validated by TrackMe because they don't respect the privacy policies the system is based on (results = 0).



5 Effort Spent

- Luca Conterio

Day	Subject	Hours
15/10/2018	Purpose, Scope and goals	1
18/10/2018	Overall Description	1.5
19/10/2018	User Interface sketch and Domain Assumptioms	2
22/10/2018	UML and Non-Functional Requirements	3.5
25/10/2018	Functional Requirements	3
26/10/2018	Statechart Diagrams	2
28/10/2018	User Interface and Scenarios	1.5
29/10/2018	Interfaces and Scenarios	1.5
03/11/2018	Sequence Diagrams	2
07/11/2018	Sequence Diagrams	2
08/11/2018	Alloy Model	3
09/11/2018	Alloy Model	4
11/11/2018	Alloy Comment and General Revision	4
Total		31

- Ibrahim El Shemy

Day	Subject	Hours
15/10/2018	Purpose, Scope and goals	1
18/10/2018	Overall Description and Product Functions	2
19/10/2018	Domain Assumptions	2
22/10/2018	Assumptions and Non-Functional Requirements	3.5
25/10/2018	Functional Requirements	3
26/10/2018	Functional Requirements	1
27/10/2018	Use Case Diagram	3
28/10/2018	Use Case Diagram and Use Cases	1
02/11/2018	Scenarios Review and Use Cases	1.5
07/11/2018	Alloy Model	1.5
08/11/2018	Alloy Model	3
09/11/2018	Alloy Model	3
11/11/2018	Alloy Comment and General Revision	4
Total		29.5

6 References

- Specification Document "Mandatory Project Assignment A.Y. 2018/2019"
- 830-1993 - IEEE Recommended Practice for Software Requirements
- "Appunti di Sistemi Informativi per il Settore dell'Informazione" A.Y. 2017/2018
- Alloy Language Reference "http://alloytools.org/download/alloy-language-reference.pdf"