Software Engineering 2: Data4Help

Requirement Analysis and Specification Document

Mattioli Daniele, Romano Leonardo, Tonelli Alessio

Politecnico di Milano

November 11, 2018

Version 1.0.2

Contents

1. Introduction

1.1. Purpose

This document is the Requirements Analysis Specification Document (RASD) for TrackMe's systems Data4Help, AutomatedSOS and Track4Run. Its purpose is to provide a detailed description of these 3 systems. It is addressed to be read by the customers, users, systems analysts, requirements analysts, developers, programmers, testers and project managers.

Data4Help is designed as a software application used to allow third parties to monitor the location and the health status of individuals. Once registered, third parties can request data both of specific users and of groups of individuals. In the former case, an individual can accept or refuse the request; in the latter case, data will be provided only if they can be properly anonymized by the application. If the request is approved, Data4Help allows third parties to subscribe to new data and to receive them as soon as they are produced.

AutomatedSOS is a service designed to allow elderly people to receive help if their pressure or heartbeat, monitored by Data4Help, go below certain thresholds.

Finally, Track4Run is designed to allow third parties to create runs, individuals to both enroll them and see the position of all runners on a map.

1.1.1. Goals

Data4Help:

- [G1] - Allows individuals to make available their position
- [G2] - Allows individuals to make available their health status
- [G3] - Allows third parties to request data of some specific individuals
- [G4] - Allows third parties to request access to anonymized data of groups of individuals
- [G5] - Allows individuals to choose whether to accept or not the request for sharing data required by third parties
- [G6] - Allows third parties to be able to see saved data as soon as a request is approved by the individual.
- [G7] - Allows third parties to have access to new data as soon as they are produced.
- [G8] - Allows third parties to be notified with the user's response

AutomatedSOS:

- [G9] - Allows individuals to receive help if their health parameters go below certain thresholds

Track4Run:

- [G10] - Allows third parties to create athletic runs

- [G11] - Allows individuals to enroll to a run
- [G12] - Allows individuals to see the position of the runners on a map during a run
- [G13] -The user can be recognized by providing a form of identification

1.2. Scope

Data4Help is a useful application to monitor individuals' data. These data are acquired by a smartwatch on which the application is installed. Using AutomatedSOS functionalities, one can be sure of receiving help if his/her health parameters go below certain thresholds. Finally, Track4Run provides interesting features for all those interested in a run. Focusing on the relevant phenomena for the system to be developed, it is possible to distinguish the world ones and the shared ones.

1.2.1. World phenomena and shared phenomena
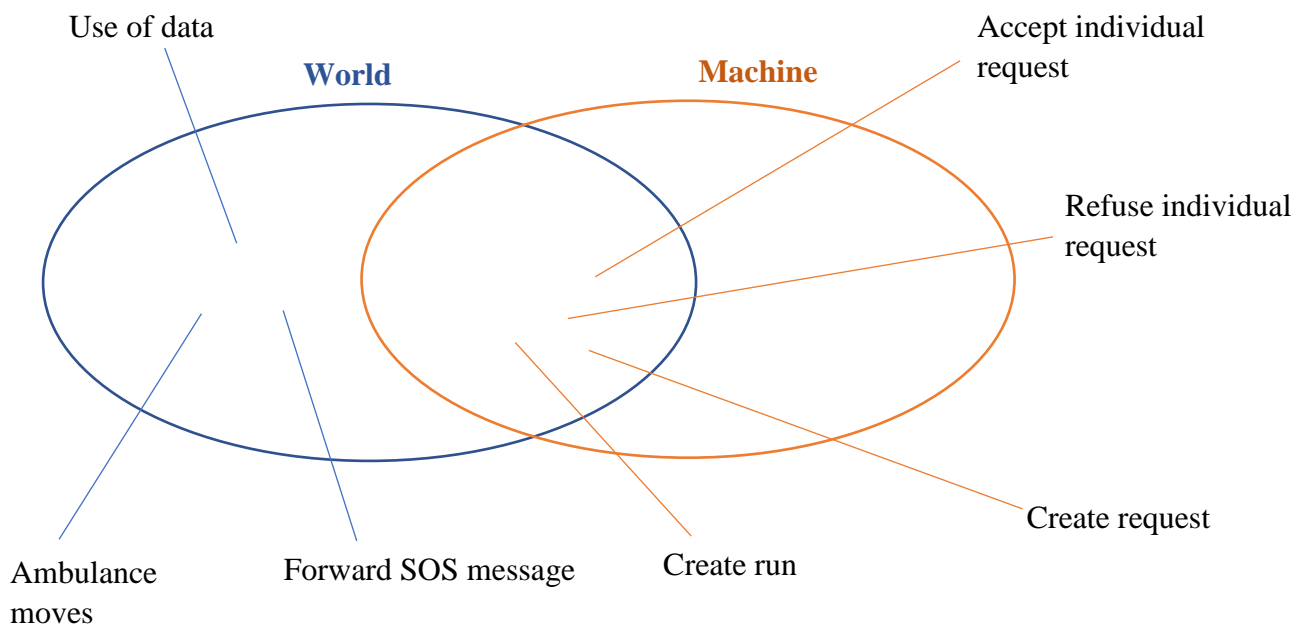
The machine, that is the portion of the system to be developed, cannot observe the use of the data acquired by third parties, the forwarding of the SOS message received by the external partner to the nearest hospital and the actual ambulance moves. The creation of both a request and a run and also the response to the individual request are shared phenomena controlled by the world and observed by the machine.

1.3. Definitions, acronyms, abbreviations

1.3.1. Definitions

- Users: people registered on Data4Help

- Individuals: people registered on Data4Help that allow the application to acquire their data through a smartwatch

- Group: set of individuals

- Third parties: people registered on Data4Help that can request data

- Participants: individuals enrolling a run

- Spectators: individuals that see a run

- Organizers: third parties that organize runs

- Data: information about health parameters and location of an individual or a group of individuals

- Help: ambulance

- Health parameters: pressure, heartbeat and quality of sleep

- Devices: electronic device on which Data4Help can be installed

- System: the whole software system to be developed, comprehensive of all its parts and modules

- Node: place from which a run passes

- External partners: switchboard to forward the message to the nearest hospital

1.3.2. Acronyms

- RASD: Requirements Analysis Specification Document

- GPS: Global Positioning System

- HTTPS: Hypertext Transfer Protocol Secure

- TLS: Transport Layer Security

- API : Application Programming Interface

1.3.3. Abbreviations

- [Gn]: n-th goal

- [Dn]: n-th domain assumption

- [Rn]: n-th functional requirement

1.4. Revision History

Version 1.0.0 – Initial version

1.5. Document Structure

The document is divided in six parts. The first one includes introductory information to give a view of what this document is about. Indeed, it describes the purpose of the system to be developed by

listing its goals and it describes also the scope of the application, by listing the world and the machine phenomena. The second part contains an overall description of the problem, including a class diagram to describe the domain model used, a state diagram to analyze the creation of a run, and the requests that a third party can send, the users' characteristics and a specification of the most important requirements with respect to the goals. The third part contains specific requirements, including external interface requirements, functional requirements, defined by scenarios, use cases and sequence and activity diagrams and non-functional requirements, defined by performance requirements, design constraints and software system attributes. The fourth part contains the alloy model and some generated worlds. The fifth part includes information about the number of hours each group member has worked for this document. The sixth part includes the reference documents.

2. Overall Description

2.1. Product perspective

Data4Help is a system designed to provide the functionalities described in section 2.2 Product function. AutomatedSOS uses data provided by Data4Help and relies on external partners to send ambulances when necessary. Also, Track4Run uses data provided by Data4Help. To better represent the structure of the system, look at the class diagram shown in Figure 1.

It is possible to observe that the system has two types of users: individuals and third parties. The former are associated to a location and a health status, because Data4Help has permission to acquire these data. Furthermore, individuals can also enroll to a specific run and retrieve information about participants during it, thus becoming spectators. The latter can retrieve data of both individuals and groups of individuals through a specific request and can organize runs, thus becoming organizers.
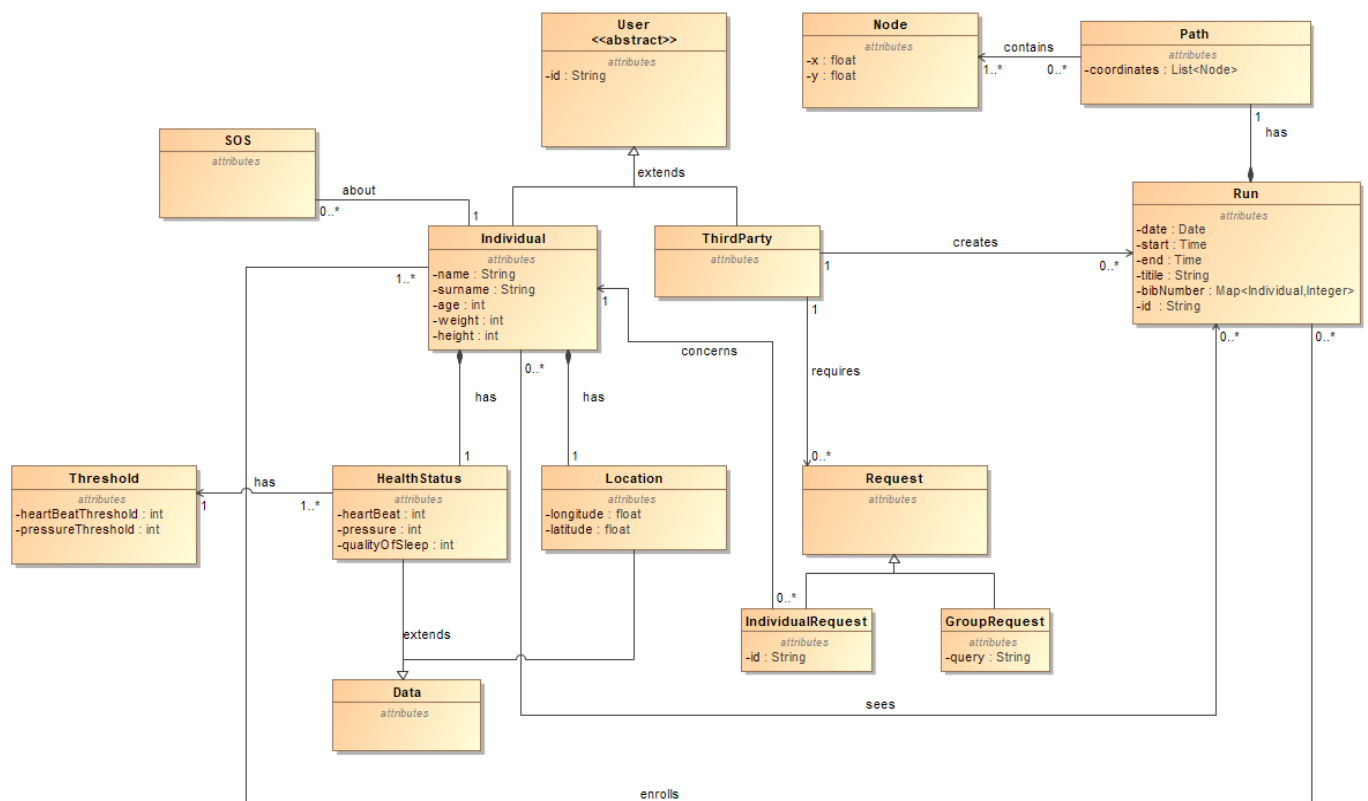
Figure 1: Class diagram

The process of creating an individual request is shown in the state diagram in Figure 2. It is possible to observe that if the id entered by the third parties is wrong, either because its format is incorrect or because there is no individual with that id, they're requested to enter the id again. The request is created when the id is correct and is sent to the specified individual.

Figure 2: State diagram for individual requests

The process of creating a group request is shown in the state diagram below.

Figure 3: State diagram for group requests

The processes of creating a new run is shown in the state diagram below.

Figure 4: State diagram for creating a run

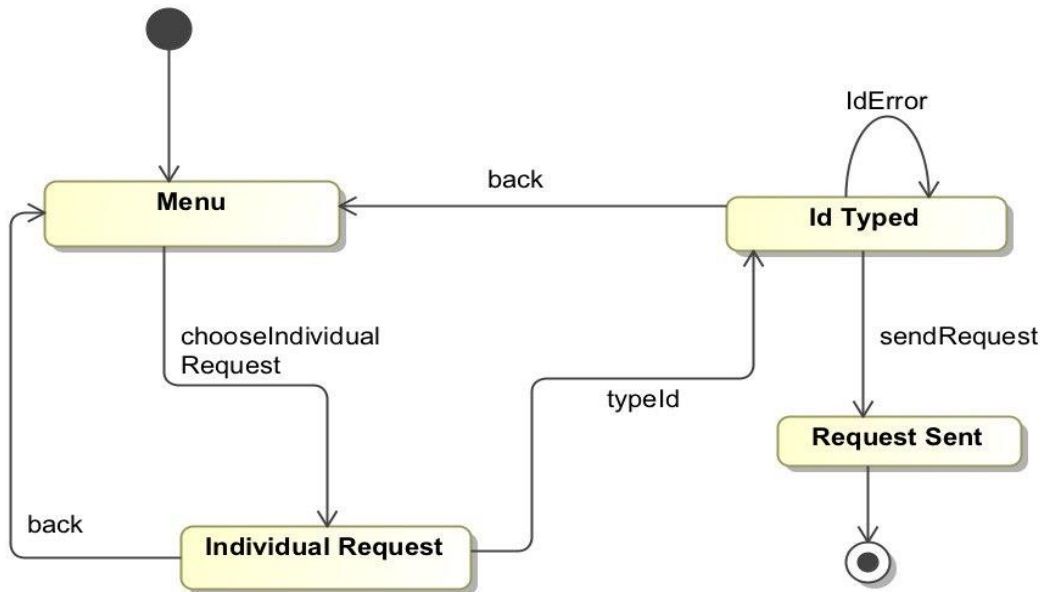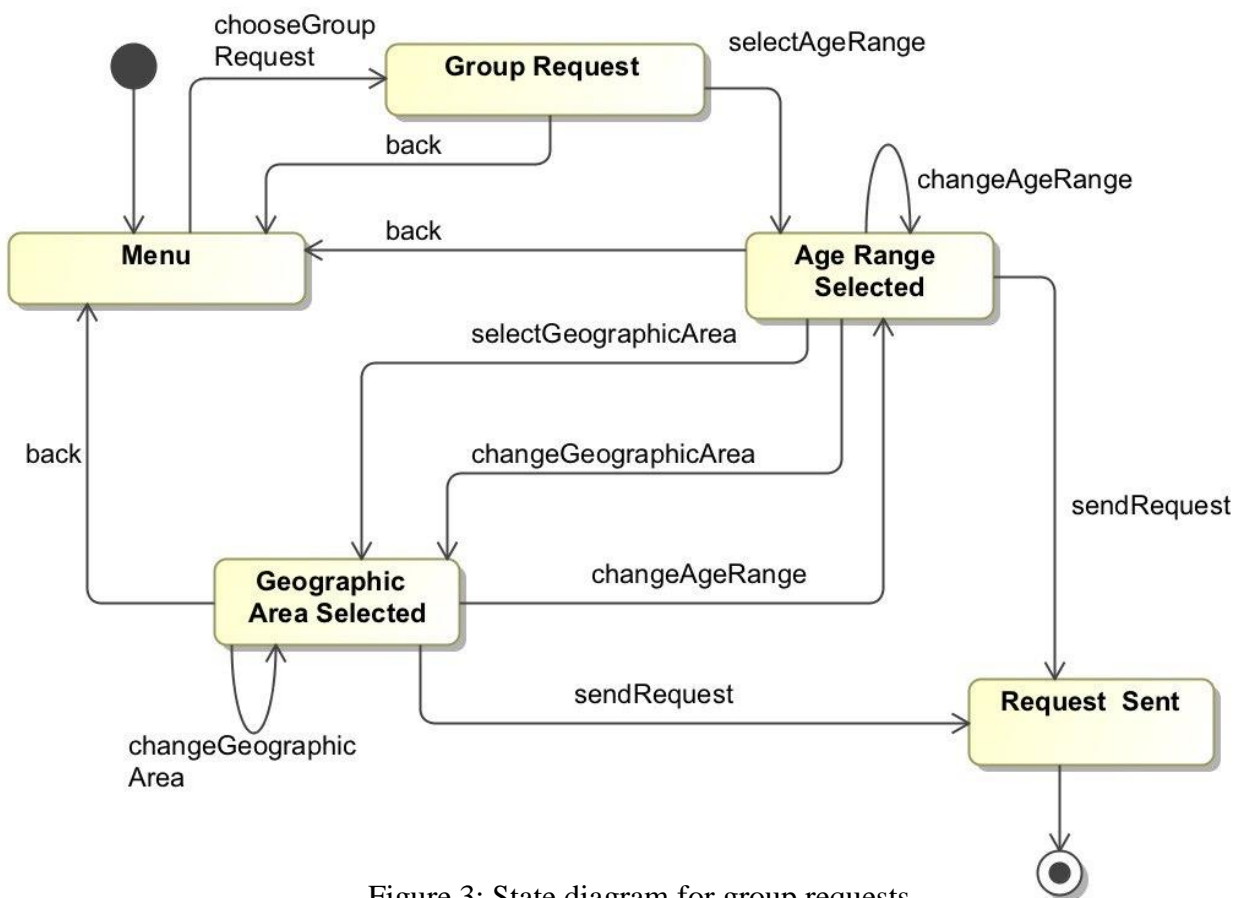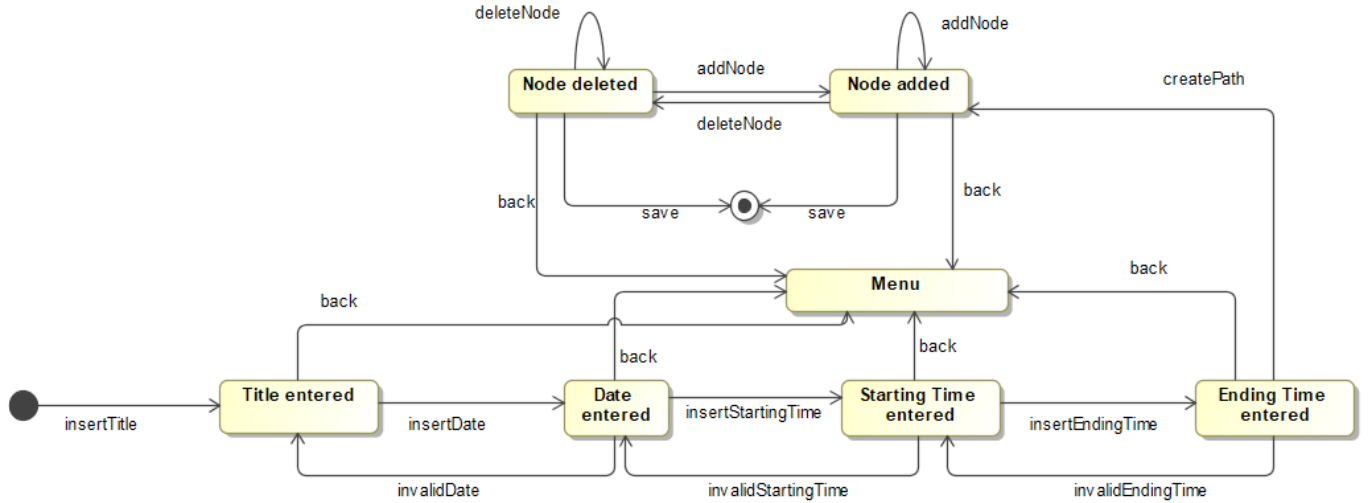It is possible to observe that creating a run implies the definition of the title, the date, the starting time, the ending time and the path. A path is made of nodes and different nodes can be added or deleted during the creation of it. If a data inserted is not valid, the third parties are asked to insert it again. If the third parties click the "Back" button during the creation of a run, they are shown the menu and their run is not saved. Otherwise, when they finish inserting the nodes, the run is correctly created and saved.

2.2. Product functions

This section includes the most important requirements with respect to the already mentioned goals of the system.

Request for data:

Third parties can request for data of both individuals and groups of individuals. In the former case, it is necessary for the third parties to know the unique ID of the person they want to send the request to. Indeed, third parties are asked to enter the ID in order to send the request. The individual receives the request through a notification and can choose whether to accept it or not. Since Data4Help is able to retrieve data directly from an individual 's smartwatch, if the request is accepted, third parties are shown requested data on their devices. If the request is refused, third parties receive a message in

which they are warned that the individual has chosen not to share his/her data. In the latter case, instead, third parties are asked to enter specific information in order to send the request. Indeed, third parties could be interested in data about people of a specific age range or living in a specific geographical area. Once the form is filled, the request is directly handled by the system: if the number of individuals whose data satisfy the request is higher than 1000, the request is accepted. In this case, indeed, the system is able to properly anonymize the requested data that are shown to third parties on their devices. Furthermore, Data4Help sends them a message asking if they want to subscribe to the data they've asked for, in order to receive new values as soon as they're produced. This request can be either accepted or refused. If the constraint is not satisfied, third parties receive a message informing them that the requested data cannot be made available for security reasons.

Monitor health status:

Individuals already signed up to Data4Help can use the functionality provided by AutomatedSOS service: by comparing health parameters with certain thresholds, the system is able to provide help to the individuals. In particular, AutomatedSOS sends, in less than 5 seconds, the external partner a message containing the location of the person who feels bad and its health parameters. The external partner is a switchboard in charge of forwarding the message to the nearest hospital. AutomatedSOS is automatically activated by Data4Help if an individual is over 70 years old, otherwise it must be manually activated by the interested person.

Create a run:

Third parties can create a run by using the functionality provided by Track4Run. They have to insert the title of the run, its date, its starting and ending time and also its path. While selecting the nodes, third party can either delete a previously inserted node or add new nodes, as shown in state diagram in section 2.1 Product Perspective. Track4Run effectively creates the run and adds it to the list of all runs if and only if all the fields are filled with valid data. The created run can be seen on the application both by third parties and by individuals.

Enroll to a run:

Individuals can see on their devices the list of all upcoming runs and they can also decide to enroll to one of them. In this case, since Track4Run exploits Data4Help data, there's no need for the individuals to enter their name, surname and age: they only have to enter their password to actually enroll the selected run. Once the enrollment is completed, the individuals are provided with their bib number.

See runners' position:

Individuals can see on their devices the list of all runs taking place and they can also decide to view on a map the position of all runners. The position is known because Track4Run exploits Data4Help measurements.

2.3. User characteristics

The intended users of the system are individuals and third party. Individuals can see their health status and position, can either accept or refuse the requests they receive, can enroll to a run, can see the position of all runners during a run and, if they use AutomatedSOS, are guaranteed to receive help in case their health parameters go below certain thresholds. Third parties can require data about both individuals and groups of individuals and can also create a run.

2.4. Assumptions, dependencies and constraints

- [D1] - The individuals' devices are able to measure all the health parameters needed by Data4Help
- [D2] - The individuals' devices are able to provide an accurate detection of the health parameters
- [D3] - The GPS must be active on the individuals' devices
- [D4] - The individuals' devices are able to provide an accurate enough current location
- [D5] - The external partner takes charge of the help request
- [D6] - An ambulance arrives to the user's location
- [D7] - A run present in the list of all runs is actually held
- [D8] - The third parties know the ID of the individual they've asked data for
- [D9] - An internet connection must be active
- [D10] - The individuals are always with the device when they are using Data4Help
- [D11] - Bluetooth must be active on individuals' devices
- [D12] - The smartwatch communicates with the smartphone through Bluetooth

3. Specific Requirements

3.1. External Interface Requirements

3.1.1. User Interfaces

The following mockups represent a basic idea of what the application will look like in the first release.

Figure 5: Sign up

Figure 6: Sign up as individual

Figure 7: Sign up as a third party
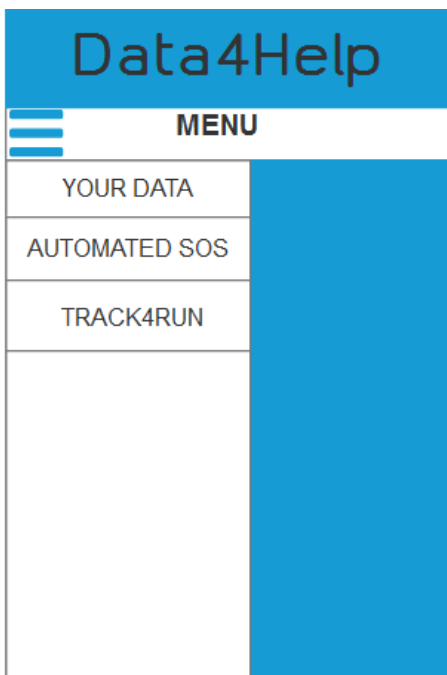

Figure 8: Log in
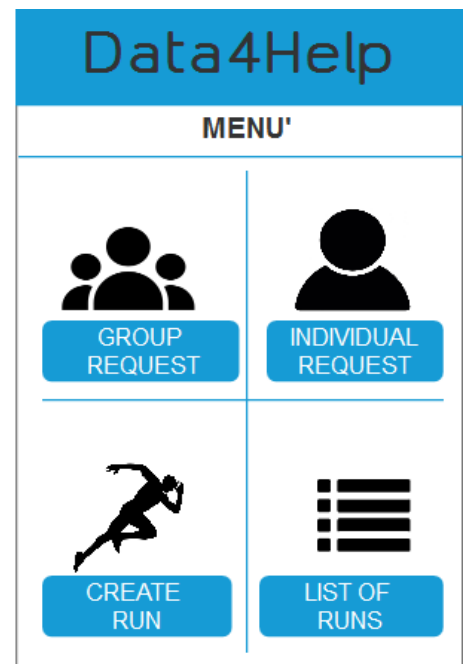
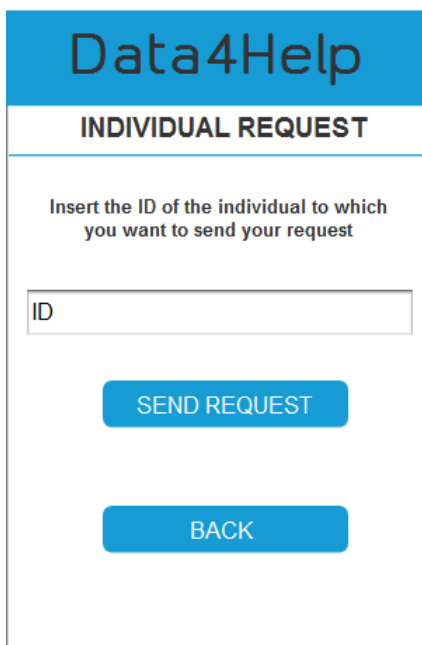Figure 9: Menu with services

Figure 10: Menu seen by third parties
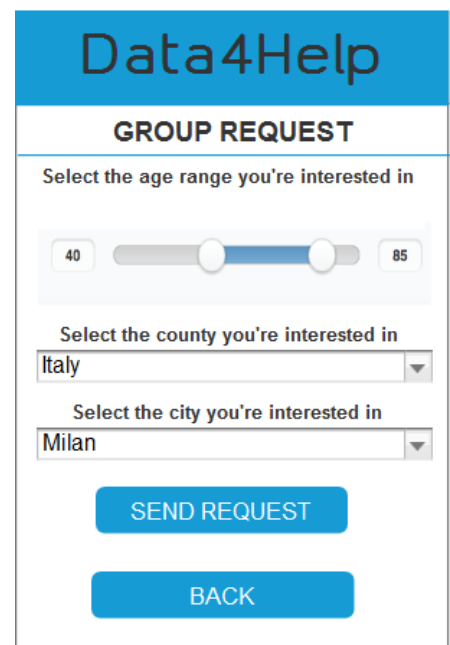

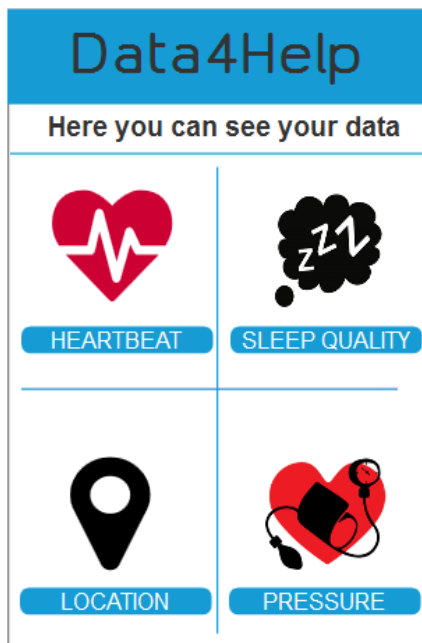Figure 11: Individual request

Figure 12: Group request

Figure 13: Individual data
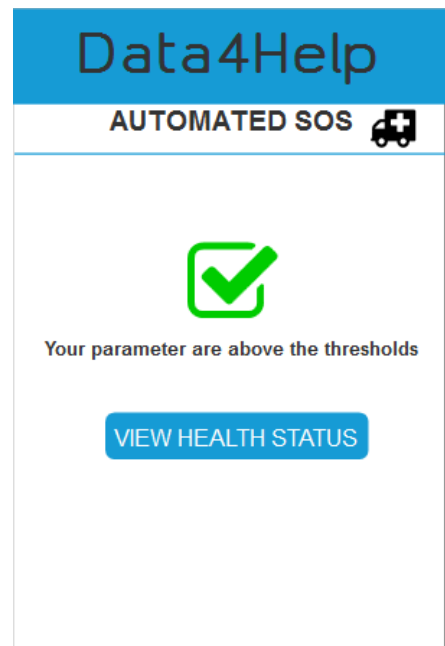
Figure 14: AutomatedSOS
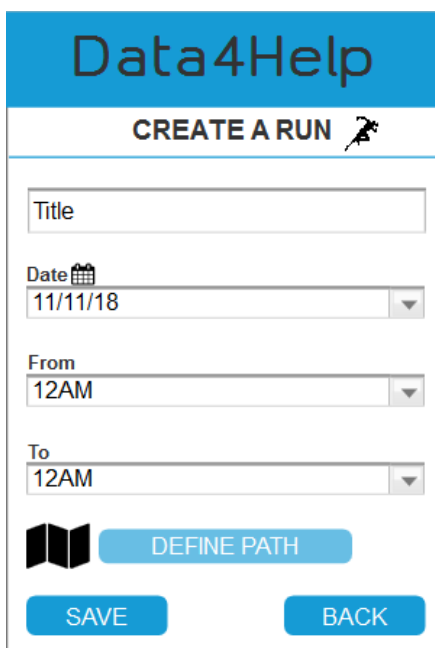
Figure 15: Create a run

3.1.2. Hardware Interfaces

The application doesn't require any hardware interfaces. However, it is required that the individuals use a smartwatch able to acquire their location, heartbeat, pressure and quality of sleep. This means that the smartwatch must be equipped, respectively, with GPS, a heartbeat sensor, a pressure sensor, and speakers. Instead, third parties can use the application on tablets, smartphones and personal computers.

3.1.3. Software Interfaces

The system will need:

- A server database where it will store users' data such as location and health parameters.
- Google Maps API to allow spectators to see the position of all runners during a run

3.1.4. Communication Interfaces

The clients communicate with the server via HTTPS requests (port 443), and TLS protocols guarantees communication security. There will be the need to build an API that lets the application to store and/or retrieve data from the database, which is located server-side.

3.2. Functional Requirements

The following list contains the functional requirements of the system to be developed:

- [R1] - Individuals' health status is provided to Data4Help by their smartwatches
- [R2] - Individuals' location is provided to Data4Help by their smartwatches
- [R3] - Third parties can identify a specific individual entering on Data4Help his/her unique ID
- [R4] - Third parties can send a specific individual the request for his/her data through Data4Help
- [R5] - Third parties can send a request for anonymized data of groups of individuals through Data4Help
- [R6] - Third parties' request is sent to the interested user by Data4Help
- [R7] - Individuals can view on Data4Help the request they've received
- [R8] - Individuals can accept or refuse the request using Data4Help
- [R9] - Third parties can view on their devices data provided by Data4Help

- [R10] - Data4Help has access to the requested data

- [R11] - Data4Help must take into account individuals' response

- [R12] - Third parties can view requested data on their devices

- [R13] - The application sends the users' data to the external partner

- [R14] - The users are shown the map

- [R15] - Third parties can create a run by entering the title, the date, the starting and ending time and the path

- [R16] - Individuals are shown a list of athletic runs

- [R17] - Individuals can select a specific run from the list

- [R18] - Track4Run shows the location only of the individuals who actually enrolled the run

- [R19] - Users can create a Data4Help account

- [R20] - Users can log in to the application by providing the combination of a username and a password that match an account

- [R21] - Users can sign up to the application by providing the required information

- [R22] - Individuals can enroll to a run through Data4Help

- [R23] - Individuals can activate AutomatedSOS functionality through Data4Help

These requirements ensure the satisfaction of the goals in the context of the domain assumptions.

- [G1] - Allows individuals to make available their position
    - [D3] - The GPS must be active on the individuals' devices
    - [D4] - The individuals' devices are able to provide an accurate enough current location
    - [D9] - An internet connection must be active
    - [D10] - The individuals are always with the device when they are using Data4Help
    - [D11] - Bluetooth must be active on individuals' devices
    - [D12] - The smartwatch communicates with the smartphone through Bluetooth
    - [R2] - Individuals' location is provided to Data4Help by their smartwatches
- [G2] - Allows individuals to make available their health status
    - [D1] - The individuals' devices are able to measure all the health parameters needed by Data4Help
    - [D2] - The individuals' devices are able to provide an accurate detection of the health parameters
    - [D9] - An internet connection must be active

- o [D10] - The individuals are always with the device when they are using Data4Help

- o [D11] - Bluetooth must be active on individuals' devices

- o [D12] - The smartwatch communicates with the smartphone through Bluetooth

- o [R1] - Individuals' health status is provided to Data4Help by their smartwatches

- [G3] - Allows third parties to request data of some specific individuals
    - o [D8] - The third parties know the ID of the individual they've asked data for
    - o [D9] - An internet connection must be active

    - o [R3] - Third parties can identify a specific individual entering on Data4Help his/her unique ID
    - o [R4] - Third parties can send a specific individual the request for his/her data through Data4Help

- [G4] - Allows third parties to request access to anonymized data of groups of individuals
    - o [D9] - An internet connection must be active
    - o [R5] - Third parties can send a request for anonymized data of groups of individuals through Data4Help

- [G5] - Allows individuals to choose whether to accept or not the request for sharing data required by third parties
    - o [D9] - An internet connection must be active
    - o [D11] - Bluetooth must be active on individuals' devices
    - o [D12] - The smartwatch communicates with the smartphone through Bluetooth

    - o [R6] - Third parties' request is sent to the interested user by Data4Help
    - o [R7] - Individuals can view on Data4Help the request they've received

    - o [R8] - Individuals can accept or refuse the request using Data4Help

- [G6] - Allows third parties to be able to see saved data as soon as a request is approved by the individual.
    - o [D9] - An internet connection must be active
    - o [R9] - Third parties can view on their devices data provided by Data4Help
    - o [R10] - Data4Help has access to the requested data

- [G7] - Allows third parties to have access to new data as soon as they are produced.
    - o [D9] - An internet connection must be active
    - o [R1] - Individuals' health status is provided to Data4Help by their smartwatches

- o [R2] - Individuals' location is provided to Data4Help by their smartwatches

- o [R9] - Third parties can view on their devices data provided by Data4Help

- o [R10] - Data4Help has access to the requested data

- [G8] - Allows third parties to be notified with the user's response
    - o [D9] - An internet connection must be active
    - o [R11] - Data4Help must take into account individuals' response
    - o [R12] - Third parties can view requested data on their devices

- [G9] - Allows individuals to receive help if their health parameters go below certain thresholds
    - o [D2] - The individuals' devices are able to provide an accurate detection of the health parameters
    - o [D3] - The GPS must be active on the individuals' devices
    - o [D4] - The individuals' devices are able to provide an accurate enough current location
    - o [D5] - The external partner takes charge of the help request
    - o [D6] - An ambulance arrives to the user's location
    - o [D11] - Bluetooth must be active on individuals' devices
    - o [D12] - The smartwatch communicates with the smartphone through Bluetooth
    - o [R1] - Individuals' health status is provided to Data4Help by their smartwatches
    - o [R2] - Individuals' location is provided to Data4Help by their smartwatches
    - o [R13] - The application sends the users' data to the external partner
    - o [R23] - Individuals can activate AutomatedSOS functionality through Data4Help

- [G10] - Allows third parties to create athletic runs
    - o [D9] - An internet connection must be active
    - o [R14] - The users are shown the map
    - o [R15] - Third parties can create a run by entering the title, the date, the starting and ending time and the path

- [G11] - Allows individuals to enroll to a run
    - o [D9] - An internet connection must be active
    - o [D11] - Bluetooth must be active on individuals' devices
    - o [D12] - The smartwatch communicates with the smartphone through Bluetooth
    - o [R16] - Individuals are shown a list of athletic runs

- o [R17] - Individuals can select a specific run from the list

- [G12] - Allows individuals to see the position of the runners on a map during a run
    - o [D3] - The GPS must be active on the individuals' devices
    - o [D4] - The individuals' devices are able to provide an accurate enough current location
    - o [D7] - A run present in the list of all runs is actually held
    - o [D9] - An internet connection must be active
    - o [R2] - Individuals' location is provided to Data4Help by their smartwatches
    - o [R14] - The users are shown the map
    - o [R16] - Individuals are shown a list of athletic runs
    - o [R18] - Track4Run shows the location only of the individuals who actually enrolled the run

- [G13] -The user can be recognized by providing a form of identification
    - o [R20] - Users can log in to the application by providing the combination of a username and a password that match an account
    - o [R21] - Users can sign up to the application by providing the required information

3.2.1. Scenarios

Scenario 1:

Andrea secretly skips classes because of a test for which he isn't well prepared. He decides to take a walk in Milan. While doing breakfast and reading the horoscope he receives a request asking for his current location. Andrea doesn't want his data to be available to third parties because he is afraid his mother will find out, in some strange way, he's not at school. Therefore, he refuses the request and orders another muffin.

Scenario 2:

While running along Naviglio river, Lucio receives a request from a company asking him to share his current data. He accepts the request and the company is able to find out what Lucio is doing. As soon as Lucio accepts, the company receives previously saved data from TrackMe and it is asked whether to subscribe to new Lucio's data or not.

Scenario 3:

Shebi is a company that wants to make a survey to compare engineering and philosophy students' quality of sleep. It uses Data4Help and by filling the form in the request section, acquires all the needed informations because TrackMe is able to properly anonymize the requested data. By comparing the average of the values of the two, Shebi finds out that engineer students sleep less.

Scenario 4:

The pharmacy of Sedriano, a small city in Milan province, wants to know if it has to increase the purchase of Norvasc, a medicine for the hypertension treatment. For doing so, it asks Data4Help the pressure data about people living in Sedriano with age between 60 and 80 years. Since TrackMe hasn't enough registered users to properly anonymize data, it is not able to provide the pharmacy with the requested information.

Scenario 5:

Martina is 78 years old and she is celebrating Christmas' Eve with her family. One of the gifts she receives is a smartwatch. Her niece, Giorgia, suggests her to download Data4Help on the new device. Martina signs up as an individual providing her e-mail, password, fiscal code and her personal information. Because of her age, AutomatedSOS service is automatically activated, in order to monitor her health status and to guarantee help in case of need.

After a week Martina is walking to the supermarket with her smartwatch tied to her wrist. Suddenly she feels a little weak. As soon as her parameters go below certain thresholds, the smartwatch starts ringing and AutomatedSOS sends Martina's data to an external partner. She faints and when she wakes up, she is on the ambulance trying to figure out what happened.

Scenario 6:

Politecnico Di Milano wants to organize a run called PolimiRun. For doing so, it decides to use a new service in Data4Help application called Track4Run. The organizer enters the title, the date, the starting and ending time and also the path of the run. The event is visible to all the users of the application.

Scenario 7:

Stefano is a Computer Science and Engineering student who wants to attend PolimiRun. He has Track4Run installed on his smartwatch and decides to use for the first time this application to enroll

the run. He doesn't need to insert his data because Data4Help provides Track4Run all the personal information. Therefore, he receives his bib number.

Scenario 8:

Mariangela has two sons attending PolimiRun. During the run she decides to use the Track4Run functionality to see on a map the position of her sons. Since the application exploits the features offered by Data4Help, including the users' location, Mariangela finds out that the youngest son has already finished instead the eldest one is still missing 3 km to the finish line.

3.2.2. Use cases

The use case diagram is represented in Figure 16.
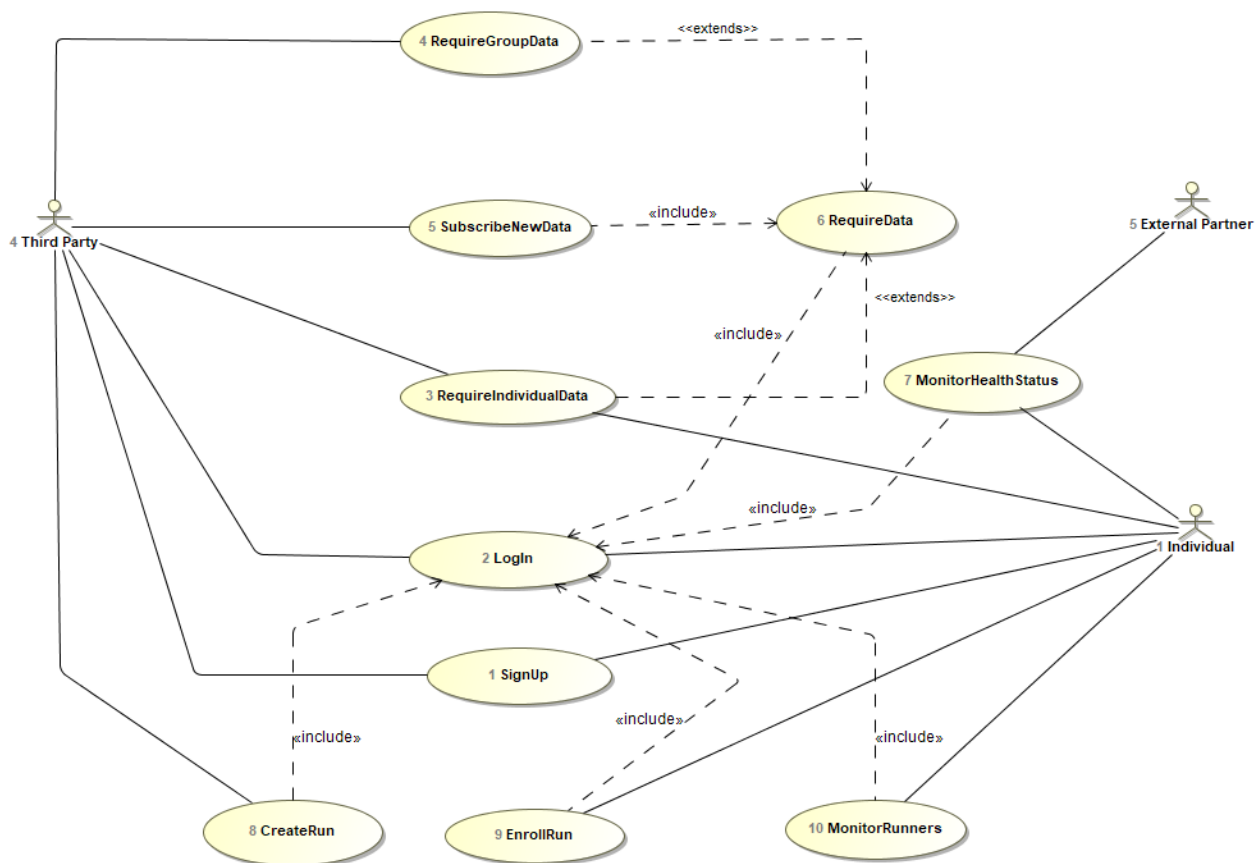
Figure 16: Use case diagram

A detailed description of the use cases is now provided.

Use case 1:

Name	SignUp
Actor	Individual Third party
Entry conditions	The user has installed Data4Help on his/her device
Events flow	1. The user clicks on "Sign up" button 2. The user chooses whether to sign up as a third party or an individual 3. The user fills all the mandatory fields and provides the necessary information 4. The user clicks on "Sign up" button
Exit conditions	The user has successfully registered and now is able to use the application
Exceptions	1. The user is already signed up 2. The user didn't fill all the mandatory fields with valid data 3. The username is already taken 4. The e-mail is already registered
Constraints and nonfunctional requirements	/

Use case 2:

Name	LogIn
Actor	Individual Third party
Entry conditions	The user has already signed up
Events flow	1. The user opens the application on his/her device 2. The user enters his/her username and password 3. The user clicks on the "Log in" button
Exit conditions	The user is successfully logged in
Exceptions	1. The user enters invalid username 2. The user enters invalid password
Constraints and nonfunctional requirements	/

Use case 3:

Name	RequireIndividualData
Actors	Individual Third Party
Entry conditions	Users have already logged in
Events flow	1. Third party clicks the "send individual request" button 2. Third party inserts the individual's ID 3. Third party selects the individual's parameters he is interested in 4. Third party sends the request to the individual 5. The individual chooses whether to accept or not the request 6. Third party is notified with the response
Exit conditions	Third party can view the response on his device
Exceptions	1. Individual's ID inserted by the third party doesn't exist 2. Third party doesn't select any parameter

Use case 4:

Name	RequireGroupData
Actor	Third Party
Entry conditions	Third party has already logged in
Events flow	1. Third party clicks the "send group request" button 2. Third party fills the form with the required data 3. Third party sends the request 4. Third party is notified with the response and is provided with the required information if and only if TrackMe is able to properly anonymize data
Exit conditions	Third party can view the response on his device
Exceptions	Third party doesn't fill all the mandatory fields with valid data
Constraints and nonfunctional requirements	The number of users satisfying third party's request must be higher than 1000

Use case 5:

Name	SubscribeNewData
Actor	Third party
Entry conditions	Third party has already received a positive response to his request
Events flow	1. Third party is asked if he wants to subscribe to new data and to receive them as soon as they are produced 2. Third party chooses whether to subscribe or not to new data
Exit conditions	TrackMe is notified with the third party's response
Exceptions	/

Use case 7:

Name	MonitorHealthStatus
Actors	External partner Individual
Entry conditions	Users has already logged in
Events flow	1. Data4Help sends AutomatedSOS an individual's health parameters 2. If these parameters go below certain thresholds, AutomatedSOS notifies the external partner, otherwise Data4Help repeats the measurements
Exit conditions	External partner is notified with all the information needed
Exceptions	/
Constraints and nonfunctional requirements	The reaction time from the moment the parameters go below the thresholds must be less than 5 seconds

Use case 8:

Name	CreateRun
Actor	Third party
Entry conditions	Third party has already logged in
Events flow	1. Third party enters the list of all runs 2. Third party clicks the "Create Run" button 3. Third party fills the form with all required information (title, date, starting time and ending time) 4. Third party defines the path on the map 5. Third party clicks the "Save" button
Exit conditions	The run has been correctly created
Exceptions	Third party doesn't fill all the mandatory fields with valid data
Constraints and nonfunctional requirements	/

Use case 9:

Name	EnrollRun
Actor	Individual
Entry conditions	Individual has already logged in
Events flow	1. Individual sees the list of all runs 2. Individual select the run he wants to enroll 3. Individual enters his password
Exit conditions	Individual has correctly enrolled to the run and has received his/her bib num
Exceptions	/
Constraints and nonfunctional requirements	/

Use case 10:

Name	MonitorRunners
Actors	Individual
Entry conditions	Individual has already logged in Individual has enrolled to the run
Events flow	1. Individual sees the list of all runs 2. Individual selects the run he/she wants to view 3. Track4Run returns the spectator a map showing all the location of all the participants
Exit conditions	Individual can see the position of all runners on his/her device
Exceptions	/
Constraints and nonfunctional requirements	/

3.2.3. Diagrams

The activity diagram of use case 2 is represented below.
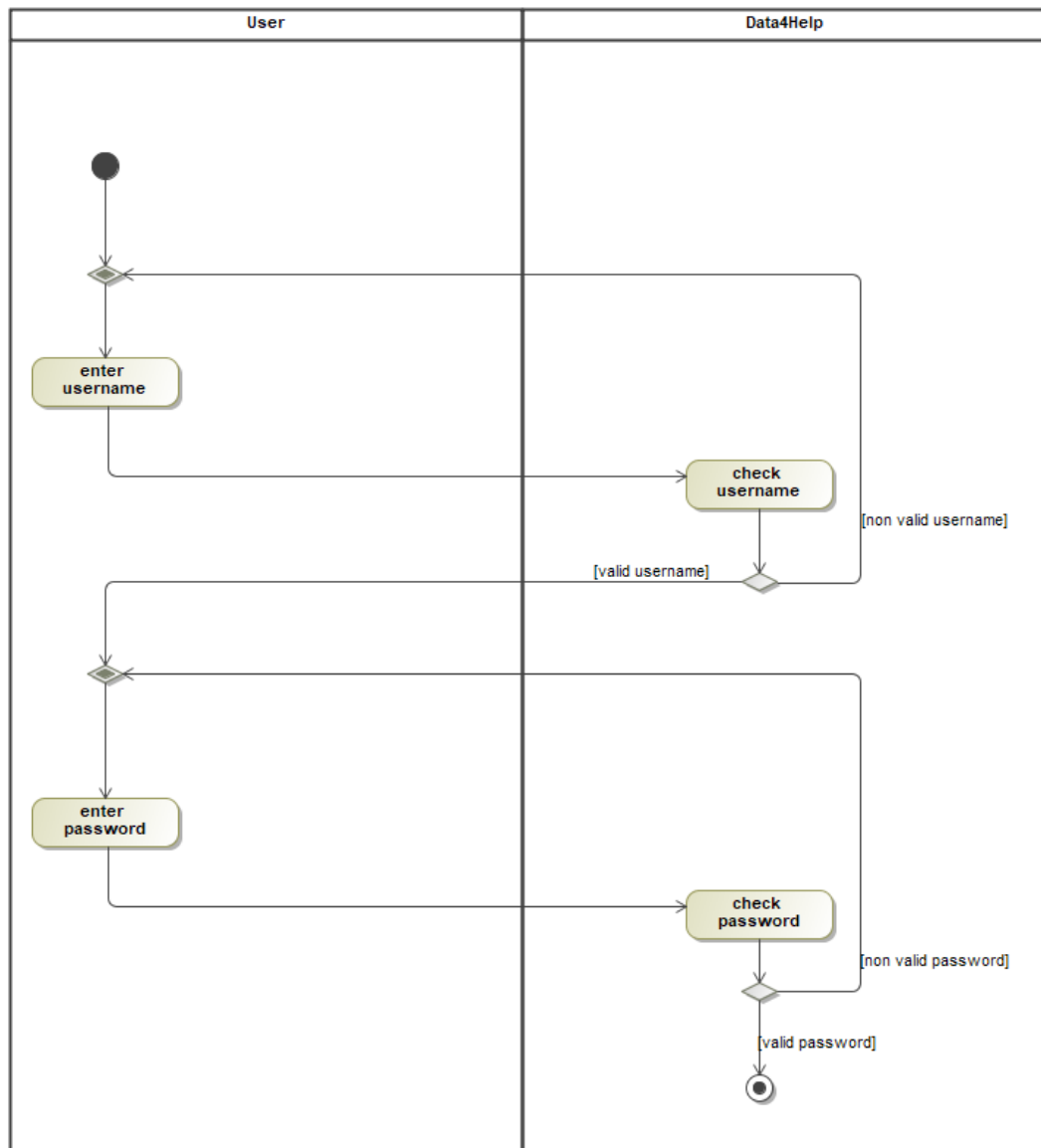

Figure 17: Activity diagram for use case 2

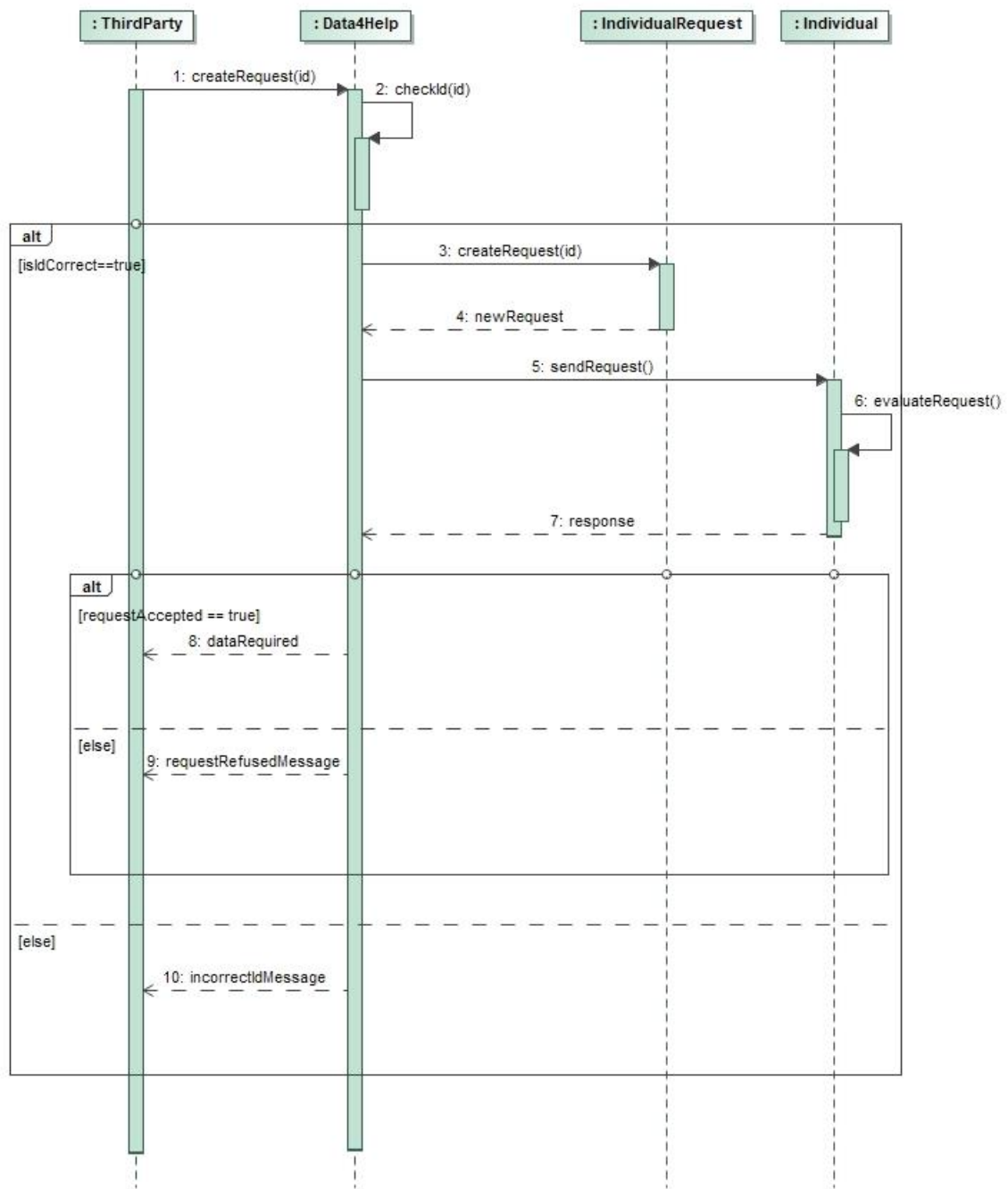The sequence diagram of use case 3 is represented below.

Figure 18: Sequence diagram for use case 3

The sequence diagram of use case 4 is represented below.
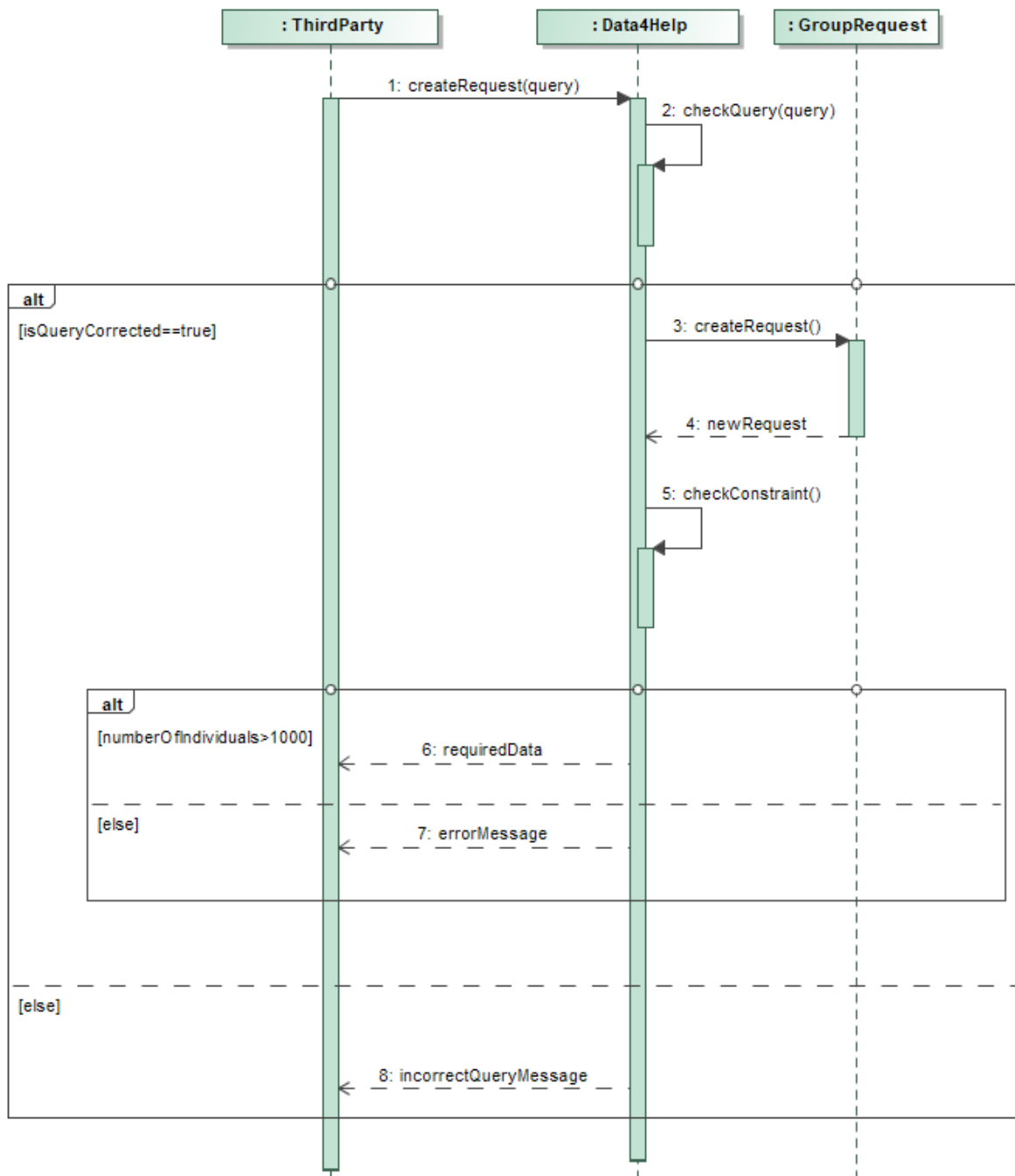
Figure 19: Sequence diagram for use case 4

The sequence diagram of use case 7 is represented below.
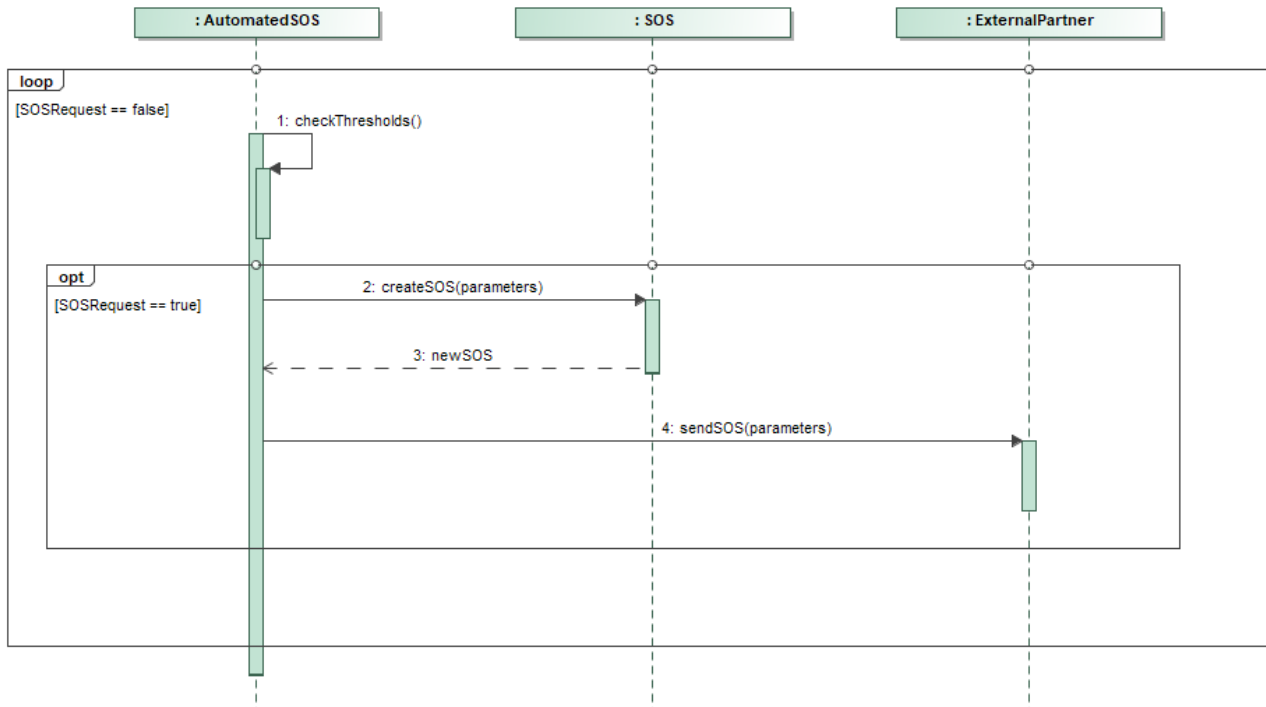
Figure 20: Sequence diagram for use case 7

3.3. Performance Requirements

- The system must support at least 300 simultaneously requests for data of both individuals and groups.
- 95% of requests shall be processed in less than 5 seconds.
- 100% of requests shall be processed in less than 10 seconds.
- The server database should be fast enough to store and search data in order of milliseconds, even if millions of records are stored.

3.4. Design Constraints

3.4.1. Standards compliance

The application must comply to the standard "iOS Human Interface Guidelines" and

"App Store Review Guidelines" by Apple regarding the iOS App and "Google

Play Guidelines" for Android. Furthermore, it preserves its state when leaving the foreground and prevents accidental data loss due to state changes. When returning to the foreground, the app must restore the preserved state and any significant stateful transaction that was pending.

3.4.2. Hardware limitations

Depending on the type of user, the application can be used on smartwatches, smartphones, tablets, and computers. These devices must have an internet connection (2G/3G/4G/Wi-Fi). Smartwatches and smartphones must have a GPS system and must communicate through a Bluetooth connection. The mobile app requires Android or iOS devices.

3.4.3. Any other constraint

The application will ask the individuals permission to get their current position and health parameters.

3.5. Software System Attributes

3.5.1. Reliability

The system is not required to guarantee a 24 hours per day and 7 days per week service. However, it should be reliable enough to offer the users a good service.

3.5.2. Availability

In order to guarantee high degree of availability, system of redundant servers may be considered. The system is expected to be available 99.99% of the time.

3.5.3. Security

- All the communication between server and clients must be protected by strong encryption using the SSL protocol.
- All attempts of establishing an unsecure communication channel, such as HTTP, with the server must be refused.
- Users' passwords must not be stored in plain text in the database.

3.5.4. Maintainability

The code should be well documented in order to enable other developers to easily understand and edit it. The system will be developed using a modular architecture that represents an important aspect to make it more maintainable and stable.

3.5.5. Portability

The web application must support the current version of Internet Explorer, Firefox, Chrome, Safari and Opera. The mobile application must be supported by the last two major version of Android and iOS.

4. Formal Analysis Using Alloy

This section contains the alloy model of some specific part of the system. The focus is on the requests that a third parties can send to a specific individual, on the SOS that is created when the parameters go below certain threshold and on the runs the third parties can create.

```
open util/integer
open util/time


----------------------------------------------sig ----------------------------------------------


abstract sig User{
    id: Int
}
{ id > 0}

sig Individual extends User {
    has: Data,
    associatedTo: set Request,
    enrolls: set Run,
    seesRun: set Run,
    SOSMessage : set SOS
}

sig Data{
    dataOf : Individual,
    hasThreshold : Threshold,
    hasValue : Int
}
{hasValue >0}

one sig ExternalPartner{
    SOSReceived : set SOS
}

sig SOS{
    SOSAssociatedTo : Individual,
    SOSAbout : Data
}
```

```
sig Threshold{
    hasThresholdValue : Int
}
{hasThresholdValue >0}


abstract sig Request{
    status: Status
}

sig IndividualRequest extends Request{
    concerns : Individual
}

sig GroupRequest extends Request {}

abstract sig Status{}

one sig Accepted extends Status{}

one sig Refused extends Status{}

sig ThirdParty{
    requires : set Request,
    seesData : set Data,
    creates : set Run
}

sig Date{}
```

```
sig Run{
    title: Int,
    date: Date,
    startTime: Time,
    endTime: Time,
    enrolledBy: some Individual,
    hasPath: Path,
}
--the beginning of a run must precede its end
{gt[endTime, startTime]}

sig Path{
 hasNode: some Node
}

sig Node{}



------------------------------------------------facts--------------------------------------------------
------------------DATA4HELP-------------------
-- Individuals have a unique ID
fact{
 all disj u1, u2: User | u1.id != u2.id
}

--Every request is required by a third party
fact{
    all r : Request | one t : ThirdParty | r in t.requires
}
--Every individual has its own data
fact{
    all disj i1, i2 : Individual | i1.has != i2.has
}
```

35

```
--Every data is linked to one individual
fact {
    all d: Data | one i: Individual | d in i.has
}


--Requests concers only individuals
fact {
    concerns = ~associatedTo
}


--If a request is accepted the third party sees the data and vivecersa
fact{
    all t: ThirdParty, r: Request | (r in t.requires and r.status = Accepted) <=> r.concerns.has in t.seesData
}


--Data are seen by third parties if and only if the request has been previously accepted by the individual
fact {
    all t:ThirdParty, d : Data | d in t.seesData <=>
    ( some i : Individual, r : Request | d in i.has and i in t.requires.concerns and r.status = Accepted  and r in t.requires)
}



------------------AUTOMATEDSOS----------------
fact {
    SOSAssociatedTo = ~SOSMessage
}

fact {
    dataOf = ~ has
}
--Every SOS is received by the external partner
fact{
    all s: SOS | s in ExternalPartner.SOSReceived
}
```

--Every threshold is associated to one health status
fact{
    all t : Threshold | one d : Data | t in d.hasThreshold
}
--If the value of data is above its threshold, there are no SOS
fact {
    all d: Data | d.hasValue >= d.hasThreshold.hasThresholdValue <=> no s : SOS | s.SOSAbout = d
}
--There are no different SOS referring to the same data
fact{
    no disj s1,s2 : SOS | s1.SOSAbout = s2.SOSAbout
}

--Every SOS is associated to the individual that has the same data of the SOS
fact{
    all s : SOS | s.SOSAssociatedTo = s.SOSAbout.dataOf
}

------------------TRACK4RUN--------------------

--Runs can be enrolled only by individuals
fact{
    enrolls = ~ enrolledBy
}
--A run can be created by only one Third party
fact {
    all r: Run | one t : ThirdParty | t.creates = r
}
--An individual cannot see a run for which he/she has enrolled and viceversa
fact{
    all i: Individual | i.enrolls & i.seesRun = none
}

```
.
--There are no two different runs with the same date and with a time overlap that cross a same node
fact{
    no disj rl, r2 : Run | rl.date = r2.date and  ( (lte[rl.startTime, r2.startTime] and gte[rl.endTime, r2.endTime]) or
                                                    (lte[rl.startTime, r2.startTime] and lte[rl.endTime, r2.endTime]) ) and
                                                    rl.hasPath.hasNode  & r2.hasPath.hasNode != none
}


--Every node belongs to at least one path
fact{
    all n: Node| some p: Path | n in p.hasNode
}


--Every path is associated to a run
fact {
    all p: Path | some r : Run | r.hasPath = p
}


--Every path has at least two nodes
fact{
    all p : Path | some disj nl, n2 : Node | nl in p.hasNode and n2 in p.hasNode
}


--------------------------------------------------predicates-------------------------------------------------
--Add a request to a third party
pred addRequest [r :IndividualRequest, t, t' : ThirdParty]{
    t'.requires = t.requires + r
}
--Accept an individual request
pred acceptRequest[r:IndividualRequest]{
    r.status = Accepted
}

pred addRun[r: Run, t,t': ThirdParty]{
    t'.creates = t.creates + r
}
```

```
--predicate to show only the part concerning the requests
pred showRequest{
    #ThirdParty>=2
    #Individual >= 3
    #Run = 0
    #SOS = 0
    #IndividualRequest >= 1
}

--predicate to show only the part concerning the SOS
pred showSOS{
    #ThirdParty=0
    #Run=0
    #Individual >= 3
    #SOS >=1
    some d : Data | d.hasValue >d.hasThreshold.hasThresholdValue
}



pred showRun{
    #ThirdParty>=1
    #Individual >= 2
    #Run >= 2
    #SOS = 0
    #IndividualRequest = 0
}

pred show {}
---------------------------------------------assertion---------------------------------------------
--If a third party sends a new request and this request is accepted then the third party can see the required data
assert ThirdPartySeesData {
    all r: IndividualRequest, t, t':ThirdParty | addRequest[r,t,t'] and acceptRequest[r] implies r.concerns.has in  t'.seesData
}


--run showRun
--run showRequest
run showSOS
--run show for 3
check  ThirdPartySeesData  for 15
```

One of the worlds generated by running predicate showRequest is the following one:
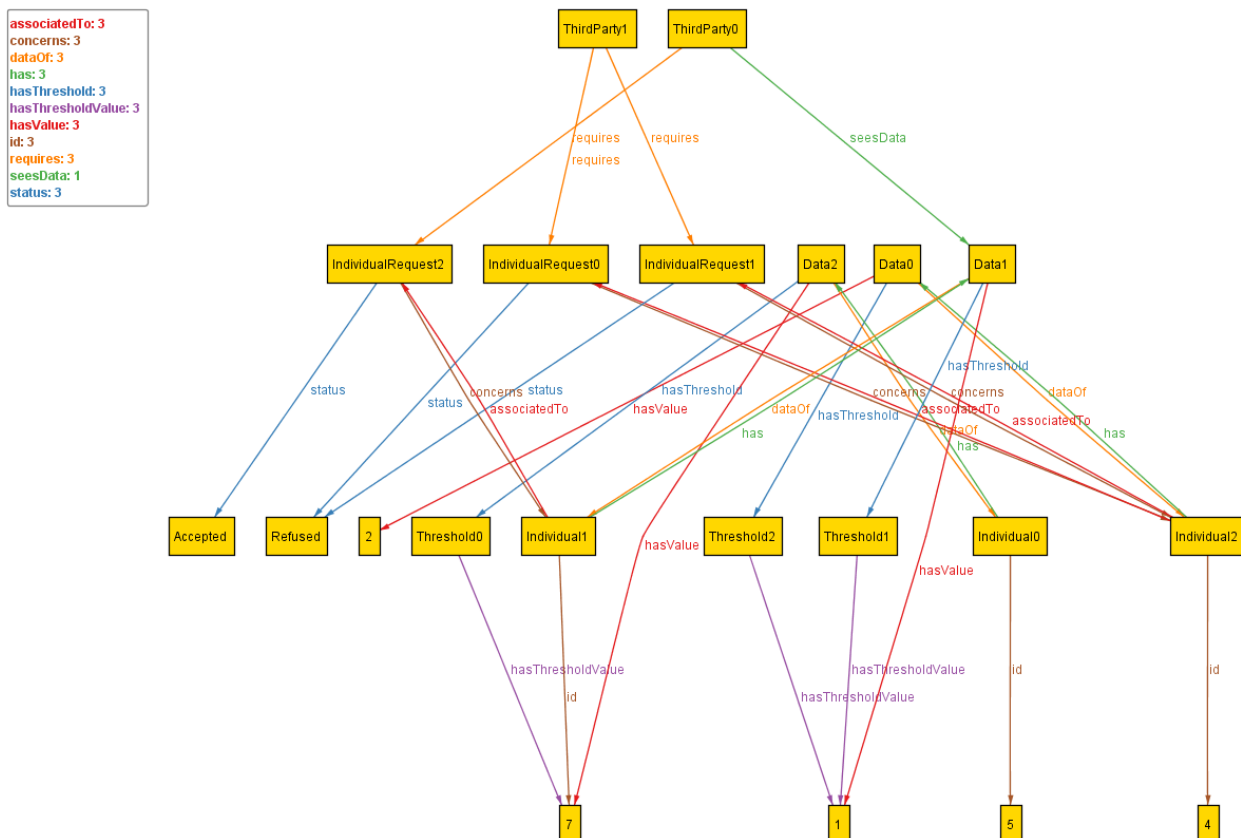

Figure 21: world generated by predicate showRequest

It is possible to observe that third parties see data only associated to accepted requests: ThirdParty0 requires IndividualRequest2, that is associated to Individual1 and is accepted. ThirdParty0 can see Individual1's data, that are Data1.

One of the worlds generated by running predicate showSOS is shown in Figure 22. It is possible to observe that an SOS is received by the external partner only if data are below threshold: SOS is about Data1, that have value 1 and threshold value 2.

Figure 22: world generated by predicate showSOS

One of the worlds generated by running predicate showRun is the following one:

Figure 23: world generated by predicate showRun

Checking assertion ThirdPartySeesData gives us the result in Figure 24


```
Executing "Check ThirdPartySeesData for 15"
  Solver=sat4j Bitwidth=4 MaxSeq=7 SkolemDepth=1 Symmetry=20
  277074 vars. 6390 primary vars. 799074 clauses. 898ms.
  No counterexample found. Assertion may be valid. 2844ms.
```

Figure 24: result of assertion ThirdPartySeesData

5.   Effort Spent

- Mattioli Daniele: 65 hours
- Romano Leonardo:30 hours
- Tonelli Alessio: 45 hours

6.   References

- Specification document "Mandatory Project Assignment AY 2018-2019"
- "Requirements engineering Part I" from Beep
- "Requirements engineering Part II" from Beep
- "Alloy" from Beep
- IEEE Standard 830-1998: IEEE Recommended Practice for Software Requirements Specifications