Software Engineering II mandatory project
A.Y. 2018/2019

TrackMe

RASD

Version: 1.0
Release date: 11/11/2018

Submitted by:
Claudia Conchetto
Riccardo Corona
Andrea Crivellin

Contents

1 Introduction

1.1 Purpose

This document is intended to give a broad description of the three services offered by TrackMe: Data4Help, AutomatedSOS and Track4Run. This will be done by a detailed presentation of the proposed solution and its purpose, listing its objectives, and the requirements and assumptions through which they will be achieved. The document is meant to be used by the clients, users and also by the parties designated with the task of creating the specified system, mainly the system and requirements analysts, the project managers, software developers and testers.

The TrackMe company wants to offer third parties the ability to monitor the position and health status of groups of people. To do this they decided to create an app, called TrackMe, which can offer this type of service and at the same time also allows ordinary users to register and make their data available in exchange for continuous monitoring.

Data4Help is a service that manages the data collected by users, who can decide whether to be available to be completely traceable or to participate in anonymous groups from which third parties can draw information. The system uses the smartphone GPS, while the acquisition of health data relies on sensors in smartwatch and smart band.

TrackMe also offers free the possibility to activate the AutomatedSOS service, recommended for elderly people but open to anyone, thanks to which the monitoring of the health status of registered users will be taken to a subsequent level: in fact, if the vital parameters exceed certain thresholds, an emergency call will be automatically activated to mobilize an ambulance, with the guarantee that this call will take place within 5 seconds from the time the thresholds are exceeded.

Finally, TrackMe offers an additional service to take advantage of the data collected with Data4Help: Track4Run. This last service allows people to be tracked as athletes participating in a race, therefore allows the organizers to create events that other users can decide whether to participate or simply follow the race from their smartphone.

1.2 Scope

1.2.1 Description of the given problem

As already explained above, TrackMe is a versatile app that once purchased provides three good services. Being an app for both companies that want to access user data, and for common users who want to monitor their data, there will be two slightly different interfaces. The generic user will be able to view his / her collected data, accept (or reject) requests to use his / her data by third parties and access the Track4Run section where he / she can create, participate or simply follow sports events. On the other hand, a user with access to a company account will have much more limited functions since the home page will be able to access lists of users who are registered for data collection, or a search section from which they will be able to search for new users or anonymous groups to sign up for. When a company account wishes to subscribe to the data of a specific user, a notification will be sent to the user in question, this will accept or reject the request, then the company can cancel the registration as the user who provides the data will prevent access to your data.

Regarding the Track4Run section, users will be able to create public or private competitions and to do so it will be necessary to insert a valid race path, consisting of: a start, an arrival (also coinciding in case of at least one further stage) start date and time (that can be postponed by the organizer or by other users who have been defined as co-organizers), in case you want to create a tender invitation, the list of invited users and finally the registration fee (optional). If you want to participate or follow a race, you can do so by selecting one of the races in the NEARBY, YOUR RACES sections.

The AutomatedSOS service is activated automatically if the user is over the age of 60 at the time of enrollment, otherwise it can be activated from the settings and always works in the background, checking that the health data does not exceed certain thresholds.

1.2.2 Goals

[G1] - Users and third parties can be recognized by providing a form with their data

[G2] - Allow third parties to access to the data of some specific individuals

[G3] - Allow third parties to access to anonymized data of groups of individuals

[G4] - Allow third parties to subscribe to new data and to receive them as soon as they are produced

[G5] - Allow the users, through the AutomatedSOS service, to come help by an ambulance when such parameters fall below certain thresholds

[G6] - Allow racing organizers, through the Track4Run service, to manage runs and define a path for runs

[G7] - Allow racing participants to enroll runs

[G8] - Allow racing spectators to see on a map the position of all runners during the run

[G9] - Allow users to monitor their own health status

1.3 Definitions, Acronyms, Abbreviations

1.3.1 Definitions

- **Generic user**: a user who does not have special accordances with TrackMe and who is the object of third parties surveys.
- **Third party**: user who has an accordance with TrackMe; the latter recognizes its profile as a public entity that can carry out surveys.
- **Data (temporary and permanent)**: parmeters recordered over time (temporary, that change from time to time) or once for all (permanent, that don't change).
- **Survey**: a research carried out on some specific parameters of one or more individuals over time.
- **By invitation run**: a run whose partecipants are only the invitated ones.

- **Open run**: a public run, everyone (according to the it's rules) can join it.
- **Identifiable individual**: a generic user is defined so (with a reasonable approximation) if his/her complete name or fiscal code is known or if he/she forms part of a group of fewer than 1000 people that is object of a survey.

1.3.2 Acronyms

- RASD – Requirement Analysis and Specification Document
- API - Application Programming Interface
- GPS - Global Positioning System
- BPM - Beats (of heart) Per Minute
- NUE - Numero Unico Per Emergenze

1.3.3 Abbreviations

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

1.4 Document Structure

Chapter 1 gives an introduction to the problem and describes the purpose of the TrackMe application. The scope of the application is defined by stating the goals and the description of the problem.

Chapter 2 presents the overall description of the project. The product perspective includes details on the shared phenomena and the domain models. The class diagram describes the domain model used, and the state diagram analyzes the process of arranging a meeting and reaching it in time. Here the majority of functions of the system are more precisely specified, with respect to the already mentioned goals of the system. In the user characteristics section the types of actors that can use the application are described.

Chapter 3 contains the external interface requirements, including: user interfaces, hardware interfaces, software interfaces and communication interfaces. Few scenarios describing specific situations are listed here. Furthermore, the functional requirements are defined by using use case and sequence diagram. The non-functional requirements are defined through performance requirements, design constraints and software system attributes.

Chapter 4 includes the Alloy model with its representation and the discussion of its purpose.

Chapter 5 shows the effort spent by each group member while working on this project.

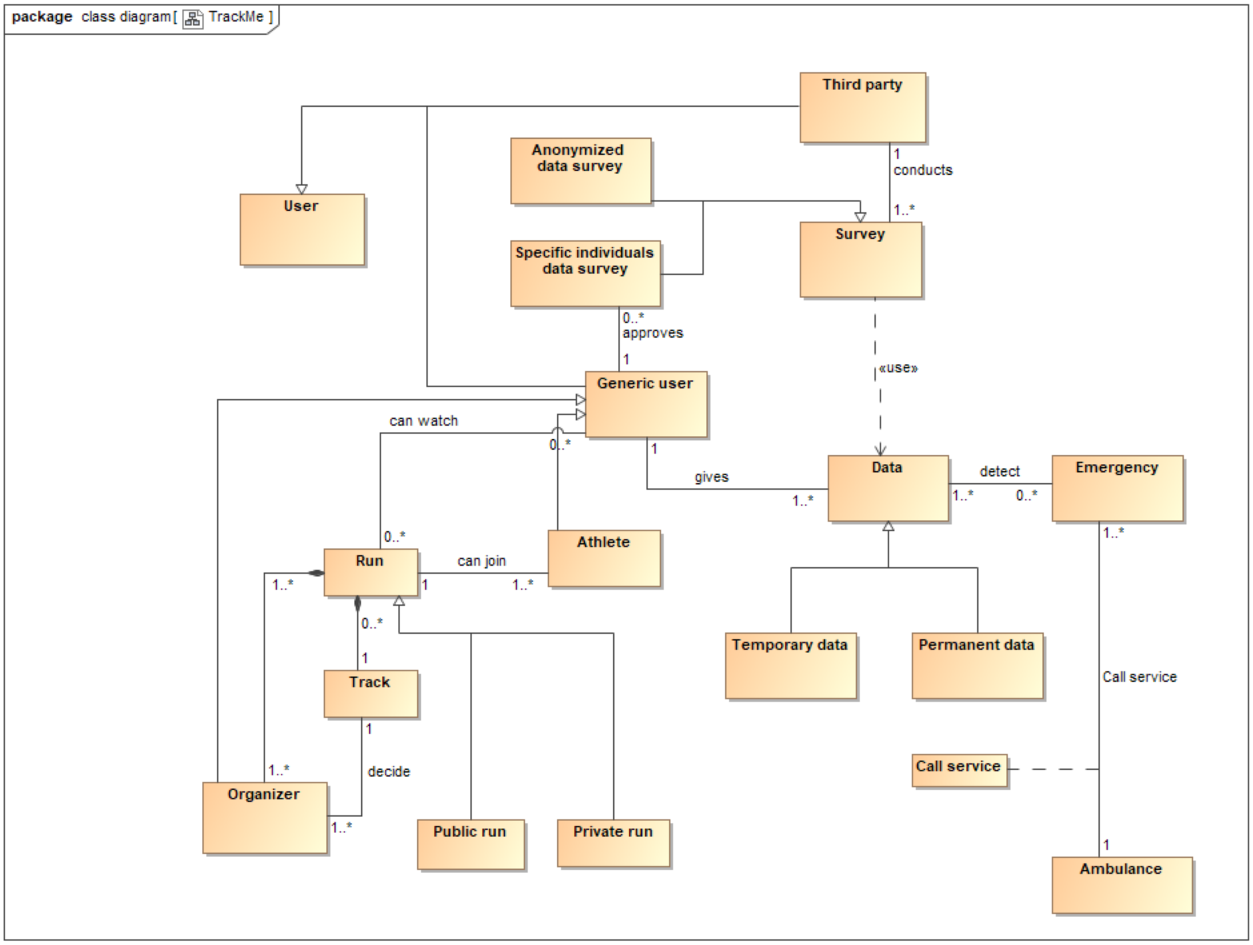
Chapter 6 includes the reference documents.

2 Overall description

2.1 Product perspective

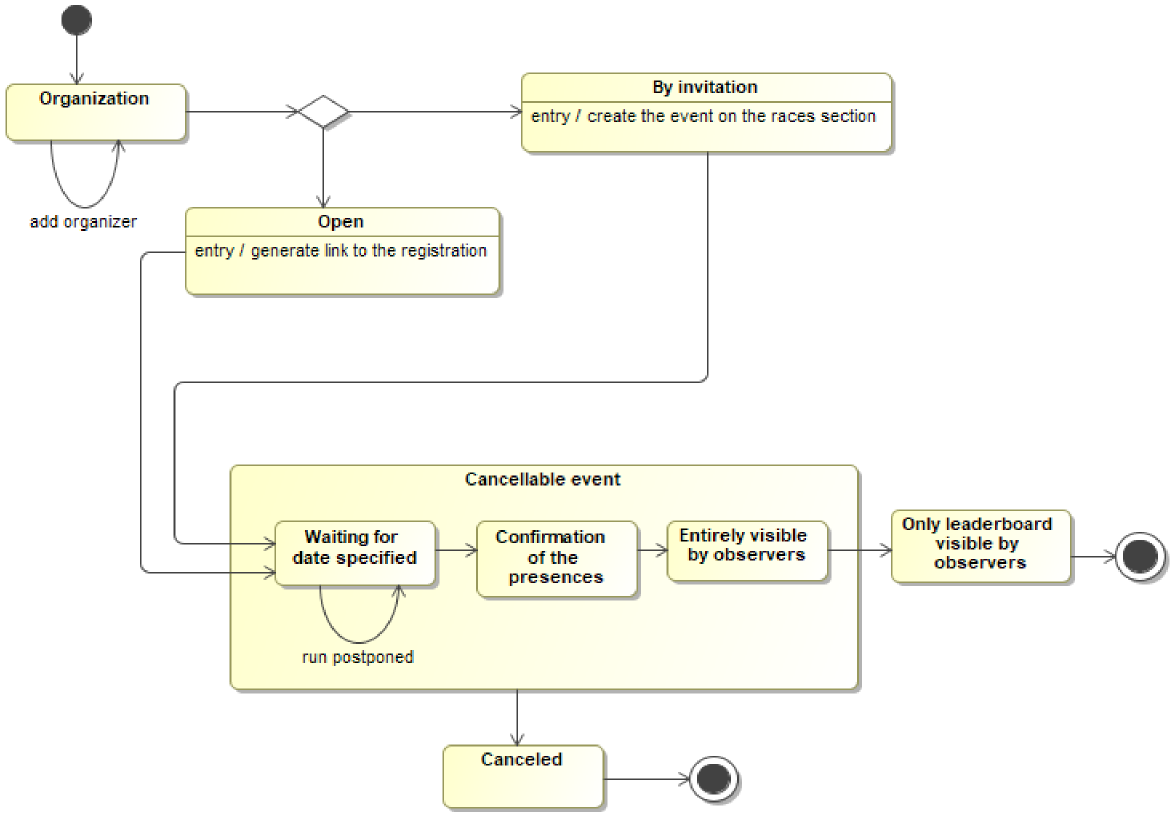
The TrackMe software-based service, Data4Help, has other two services built on its top: AutomatedSOS and Track4Run. These two services exploit Data4Help's features, resources and links to external services. Not all the system's tasks in fact are provided only by the application: maps or payment services are committed to external systems. The application provides some services also for users who install it on a non-weareble device, but it mainly exploits weareble devices' features for all the three services. They are indeed based on health and location features that must be updated in real-time.

A view of the relations that intervene among concepts that will be at the base of the system is given by the following diagram.



A central role is performed by the generic user and by data that can tell him/her and third parties about his/her status. Besides data management (with a semi-passive role for the generic user), the application is about running races and it allows generic users to enter in contact with each other watching or organizing them. Finally, we can see different types of data, races and surveys, that requires different interactions with users.

As the running race management is one of the most complex services among those offered to be understood in its phases, here's a specification of them.


Organization
add organizer
By invitation
entry / create the event on the races section
Open
entry / generate link to the registration
Cancellable event
Waiting for date specified
run postponed
Confirmation of the presences
Entirely visible by observers
Only leaderboard visible by observers
Canceled

In order to know how to behave consequently to the organization form filling, it is required to indicate if the running race will be open or by invitation. After this first part the run enters in a state in which it is allowed to cancel it until it's end is reached.

2.2 Product functions

The requirements of all services dealt with above can be grouped in the following system's main functions.

2.2.1 Data collection and updating

This function is the one on which is based Data4Help service and that is exploited by the other two services. For a generic user (not a third party), data are collected through a form during the registration phase and subsequently through automatic update.

This takes place through the sensors and the GPS system of the wearable device on which the service is active, which update the health status and position of the user. These data collected in real-time are the ones that AutomatedSOS needs (if the user is subscribed at it) to call for help if the user's health is in a state of emergency. Data inherent only to localization will be instead fundamental to allow the service Run4Track to let other users follow the runners' movements during the race.

2.2.2 Data exchange with external entities

Complementary to the previous one, this function deals with the management of those aspects that affect third parties and those that allow completing AutomatedSOS service.

Data4Help is in fact thought to give a service to third parties: companies, research institutes, and whatever entity could be interested in getting hold of data of a person for some reason (such as leading surveys or sending specific advertising to some groups of people). Third parties, after being adequately identified through a code agreed with the company TrackMe, can obtain two types of data from the service: specific individuals ones or anonymized ones.

In order to obtain data of specific individuals, in which the survey's object can be considered identifiable, a third party can search through a search dialog or filters an individual or a group of fewer than 1000 persons. Then, Data4Help creates a request for use of data that the interested users will receive as a notification. Only after receiving their consent Data4Help will allow the third party to receive those data, updated from time to time.

If the third party wants to lead an anonymized survey instead, the application doesn't need the users' consent and it gives automatically data of groups of users of more than 1000 persons.

Regarding AutomatedSOS, once a situation of emergency is ascertained, the application is capable, thanks to the data collected and the agreement with emergency services in Europe and USA, of providing an immediate and accurate help service.

2.2.3 Run management

Here "management" is intended to indicate the management of a running race led by a user, who is allowed by the application to organize it and update its status. In fact, the user is able to organize a new race through a form and an external map service.

The organizer can optionally indicate in the form other organizers, who will help in the race management, once the initial phase of race profiling is completed. After the event is created, every organizer has the possibility to postpone the race before its begin and to cancel it before its end. Other two actions can be performed by the organizers: sign a runner as "present" at the race and as "disqualified" with respect to its rules. Those two actions can be very useful to be performed by several persons, especially during races of certain dimensions. Furthermore, those flags will be fundamental for the race monitoring of other users.

During the form compilation, the user has to specify if the race will be open or by invitation, in order to create respectively the public announcement or private links to a registration form.

2.2.4 Data monitoring and consequent actions on them

The generic user, daily opening the application finds immediately his/her data recorded through the wearable device and notifications (such as the request of data sharing from a company). Naturally, the user can answer the notification as far as required. In another section that can be found through the menù, the user can monitor races to whom he/she subscribed or which are organized by him/her, and next to it he/she finds a list of races in neatness and time order. There the user can subscribe to a race or watch it in real-time.

The display of the race takes place through maps provided by external services and on the shown path there are markers associated with runners signed as "present". On the screen, there's also a leaderboard with the runners' names associated with the markers through a number. It is visible if a runner is disqualified (the marker is blocked where he/she was disqualified and the name is grey).

2.3 User characteristics

The system has the following actors:

- **Generic user**: a person who has a user account through which he/she monitors his/her data and can subscribe to services for emergency automated calls and for joining running races, as well as organizing them.
- **Third party**: a company or association registered agent who can ask for groups' or individuals' data.
- **Administrator**: A TrackMe employee who can make some changes in the forms other actors have to compile.

2.4 Assumptions, dependencies and constraints

[D1] - Third parties own an alphanumerical code received from a TrackMe administrator used to verify their identities and match them with the company account.

[D2] - Device sensors can provide accurate data to the app.

[D3] - TrackMe is affiliated with NUE 112 (Numero Unico per le Emergenze) to offer assistance in Europe, and with 911 to offer assistance in the USA.

[D4] - The run can be joined only by persons enrolled through the app.

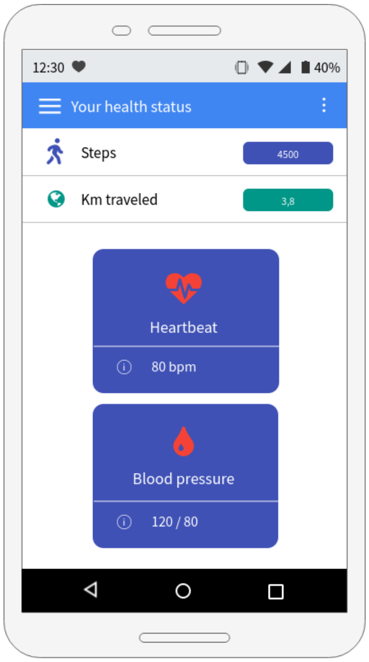
[D5] - If it is required a payment to enroll a run, an external service will guarantee secure transactions and receipts by e-mail.

3 Specific requirements

3.1 External interface requirements

3.1.1 User interfaces

The following mockups represent a basic idea of what the mobile app will look like in the first release. The last one shows instead an idea about the minimal interface of the wearable application.



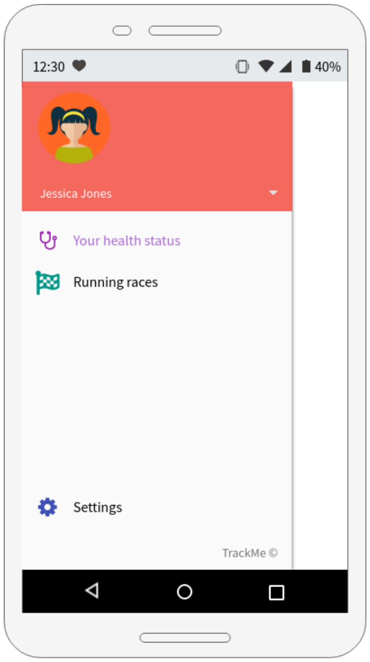
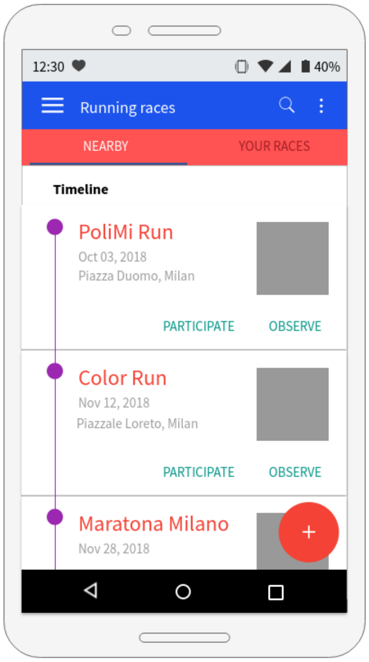


Figure 1: [Mockup] - Home page of user's account (the user can monitor his/her own parameters)
Figure 2: [Mockup] - Side menu

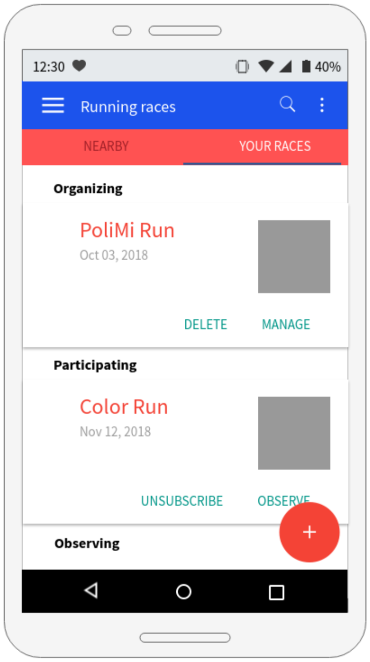


Timeline

PoliMi Run
Oct 03, 2018
Piazza Duomo, Milan

Color Run
Nov 12, 2018
Piazzale Loreto, Milan

Maratona Milano
Nov 28, 2018



Organizing

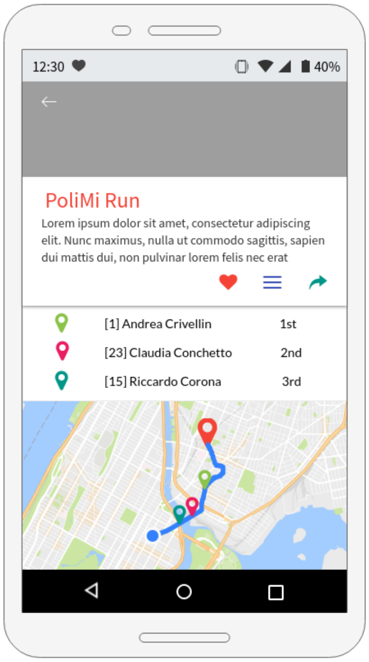
PoliMi Run
Oct 03, 2018

Participating

Color Run
Nov 12, 2018

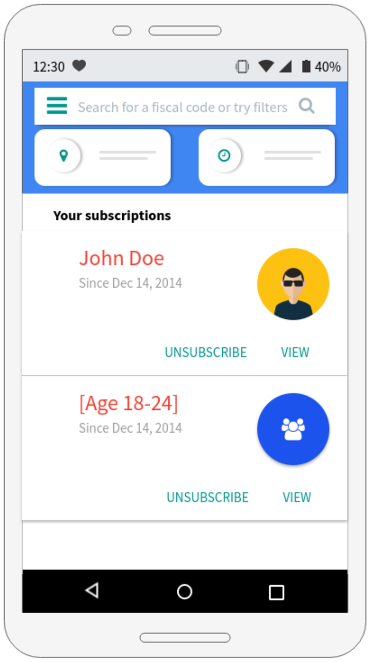
Observing

Figure 3: [Mockup] - Races screen (NEARBY tab)
Figure 4: [Mockup] - Races screen (YOUR RACES tab)



PoliMi Run
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem felis nec erat

[1] Andrea Crivellin	1st
[23] Claudia Conchetto	2nd
[15] Riccardo Corona	3rd

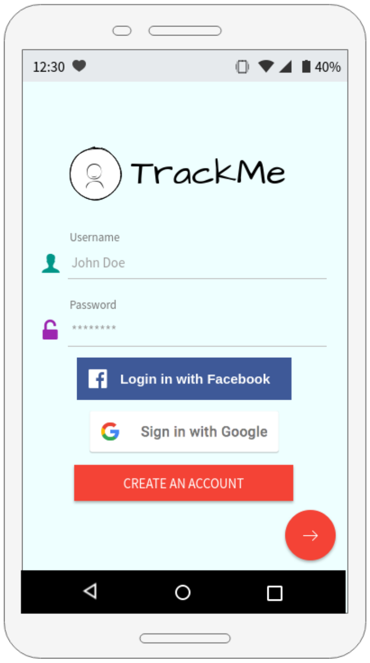


Your subscriptions

John Doe
Since Dec 14, 2014

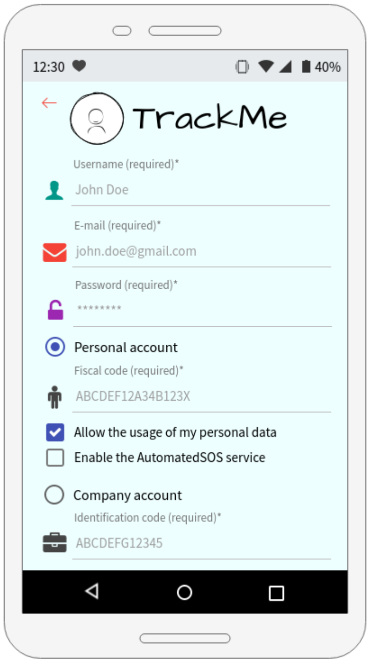
[Age 18-24]
Since Dec 14, 2014

Figure 5: [Mockup] - Race overview with leaderboard and track
Figure 6: [Mockup] - Home page of company's account (the company can search and manage data)



Username
John Doe

Password
****



Username (required)*
John Doe

E-mail (required)*
john.doe@gmail.com

Password (required)*
****

● Personal account

Fiscal code (required)*
ABCDEF12A34B123X

☑ Allow the usage of my personal data

□ Enable the AutomatedSOS service

○ Company account

Identification code (required)*
ABCDEFG12345

Figure 7: [Mockup] - Login screen
Figure 8: [Mockup] - Registration screen

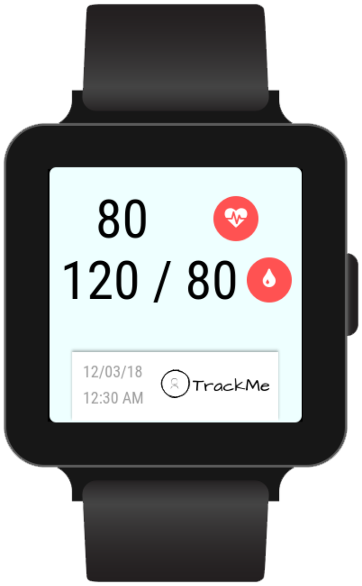


Figure 9: [Mockup] - Example of the app screen on a smartwatch

3.1.2 Hardware interfaces

The application doesn't use any hardware interface, however a smartphone paired with a wearable device like a smartwatch or a fitness tracker is required to use the Data4Help and AutomatedSOS services, and to participate to races using Track4Run (pairing in most cases is made via bluetooth).

The wearable device must have at least the following sensors: GPS, accelerometer, altimeter, gyroscope, heart rate sensor. Other biosensors supported are related to the detection of glucose level, blood pressure, body temperature and ECG and devices equipped with them are highly recommended for a complete experience with TrackMe.

Organizers and spectators of running races, and also third parties can manage their activities without specific hardware requirements (wearable device not required for their tasks).

Users who own a smartwatch with Wear OS or watchOS can also download a TrackMe version specifically designed for this kind of devices.

3.1.3 Software interfaces

The application obviously offers a graphical user interface, which allows users to easily communicate with the device.

Furthermore the app uses a city maps external service, suitable for embedding in order to simplify the overall architecture and make it more lightweight. Because the Track4Run service is based on planning, participating and following running races, the app needs accurate real-time informations for mapping the track. Google Maps APIs represent the best choice in order to provide an effective service. Google Maps also provides the users with the reliable location information they need throughout the world.

3.1.4 Communication interface

The app uses the following communication interfaces:

- Internet connection for communication of data.
- RESTful APIs and JSON data format for communication over HTTP.
- SDK Maps for Android, SDK Maps for iOS and API Maps JavaScript in order to allow an efficient Google Maps integration (these APIs allow to add an interactive and personalizable map in a mobile application, on Android and iOS).
- Interface with the push service(s) via own APIs in order to allow the app to send and receive push notifications to users' devices (both iOS and Android are supported).
- Interface with SMS and voice gateway provider via standard APIs in order to allow the app to send SMS and to make phone calls (useful for the AutomatedSOS service).
- Wearable APIs in order to get data from smartphones, fitness trackers and other wearable devices.

3.2 Scenarios

3.2.1 Scenario 1

Aurora is a non-profit organization that deals with disabled young people. Aurora's board of directors decided to use TrackMe to support medicians in their work and monitor the health status of patients in a more efficient way. Most of the patients, according with their parents, decided to download the app on their smartphone. Aurora's medicians were able to estimate the average course of the heartbeat, blood pressure, body temperature, glucose level and ECG of patients just searching for anonymous data using the age filter (18-21 years old) and the location filter (via Aurora 2, Perugia) and subscribing to the search results.

3.2.2 Scenario 2

Davide is the athletic trainer for the under 21 athletes of Milano Atletica. He decided to organize a competition between his student, so he launched TrackMe and created a new running race in the "Your running races" section, using the "+" button. He selected the location of the sport center (via Cimabue, Milan) and established the date of the race (15/11/2018) and the closing date of the registrations (14/11/2018). He obviously decided to select the participants sending them invites to join the competition.

Giovanni, one of his students, immediately joined the competition but unluckily he was not able to compete due to an injury. Through the "MANAGE" option Davide was able to mark him as "disqualified" in order to effectively know the participants in the race and monitor their performances.

3.2.3 Scenario 3

Lidia is an elderly woman who loves walking. His grandson Matteo suggested her to download the TrackMe app on her smartphone paired with her fitness tracker, in order to monitor her own activities and health status and also to make sure his grandmother receives assistance in case of health problems.

One day, during a walk, Lidia felt sick and collapsed on the ground. TrackMe detected an abnormality in the heartbeat and called immediately the 112, sending also an SMS with her position, heart rate and blood pressure. The ambulance intervened a few minutes later and Lidia was successfully rescued.

3.3 Functional Requirements

[G1] - Users and third parties can be recognized by providing a form with their data

[R1] - The usernames used in the system are unique to every user and third party.

[D1] - Third parties own an alphanumerical code received from a TrackMe administrator used to verify their identities and match them with the company account.

[R2] - Users and third parties can create an account by compiling a form.

[R2.1] - The form should contain the following: username, password, choice between personal and third party account, other anagraphical info (for the user) or company info (for the third party).

[R3] - Third parties must provide an identification alphanumerical code to confirm their identity.

[R4] - Users and third parties can log in to the application by providing the combination of a username and a password that match an account.

[G2] - Allow third parties to access to the data of some specific individuals

[D2] - Device sensors can provide accurate data to the app.

[R5] - Third parties can search for specified data by filling the search bar with the fiscal code of a specific user.

[R6] - A notify is sent to the selected user, who can accept or refuse the sharing of data with the specific company.

[R7] - A message is sent to the third party, containing user's data if he/she allowed the sharing or a notification of refuse otherwise.

[R8] - Users under 18 years old can't be shown in the results of a search.

[G3] - Allow third parties to access to anonymized data of groups of individuals

[R9] - Third parties can search for anonymized data of a specific group of users by selecting search filters (location, age, sex, time slot).

[R10] - Anonymized data are shown just if the research produces at least 1000 results.

[G4] - Allow third parties to subscribe to new data and to receive them as soon as they are produced

[R11] - Third parties can flag an option to subscribe to new data of a specific user when they receive his/her data (the user already allowed the data sharing), or they can subscribe to get results of a research with specific filters as soon as they are produced.

[R12] - Third parties can manage their subscriptions and set the frequency with which obtaining data for a specific subscription by selecting it and editing preferences.

[G5] - Allow the users, through the AutomatedSOS service, to come help by an ambulance when such parameters fall below certain thresholds

[D3] - TrackMe is affiliated with NUE 112 (Numero Unico per le Emergenze) to offer assistance in Europe, and with 911 to offer assistance in the USA.

[R13] - Users can manage the subscription to the AutomatedSOS service through a specific option in the settings menu.

[R14] - Old users (60+ years old) are automatically subscribed to the AutomatedSOS service since their registration.

[R15] - The app calls autonomously the NUE (or 911) when such parameters fall below certain thresholds, and sends concurrently user's data (GPS location and health status parameters) via SMS.

[G6] - Allow racing organizers, through the Track4Run service, to manage runs and define a path for runs

[R16] - Users can create races by defining name, date, time, start point, stages and arrival point, and selecting the method of participation (open or by invitation) and the date and time of expiration for registrations. They can also add other organizers as collaboratos: selected users will receive a notify and they will see the race in their management section.

[R16.1] - If a race by invitation is selected, a private link to a registration page is generated.

[R16.2] - The organizer can specify a maximum of 10 stages.

[R16.3] - The distance between start and arrival points (stages included) cannot exceeds 50kms, otherwise a warning popup is shown and the organizer is invited to change at least one of them before continuing.

[R17] - Organizers can choose one of the routes calculated by the system.

[R18] - The system allows the organizers to sign an enrolled runner as "present" and to disqualify a runner by signing him/her as "disqualified".

[R19] - Users can manage their organizing races in a management section in the app.

[R18.1] - Organizers can postpone or delete a run. A notification is sent to participants and spectators.

[G7] - Allow racing participants to enroll runs

[R20] - Users can visualize nearby races or search for a specific run by searching for the name or location of the race.

[D4] - The run can be joined only by persons enrolled through the app.

[D5] - If it is required a payment to enroll a run, an external service will guarantee secure transactions and receipts by e-mail.

[R21] - Runners can join a race through the race overview by clicking the ”Participate” button. For races by invitation the button is visible just to invited users following the registration link.

[R20.1] - If it’s specified that some payment is required, it will be committed to an external service.

[R22] - When the race starts, the participant have to wear his/her wearable device with Data4Help installed.

[G8] - Allow racing spectators to see on a map the position of all runners during the run

[R23] - Users can become spectators of a race through the race overview by clicking the ”Observe” button.

[R24] - For the duration of the race is always possible to see in the event page the map with markers, associated to runners’ name through numbers, and a live leaderboard showing names, order numbers and possible disqualifications.

[R25] - A marker is on the track iff it corresponds to a user enrolled, signed as ”present” at the run with his/her wearable device with Data4Help installed and not disqualified.

[R26] - The track represented is the one chosen by the organizer of the run.

[R27] - Markers move according to the corresponding persons’ movement, as reported by the location service.

[G9] - Allow users to monitor their own health status

[R28] - Users can visualize their own health status in the home page section of the app.

3.3.1 Use case diagrams

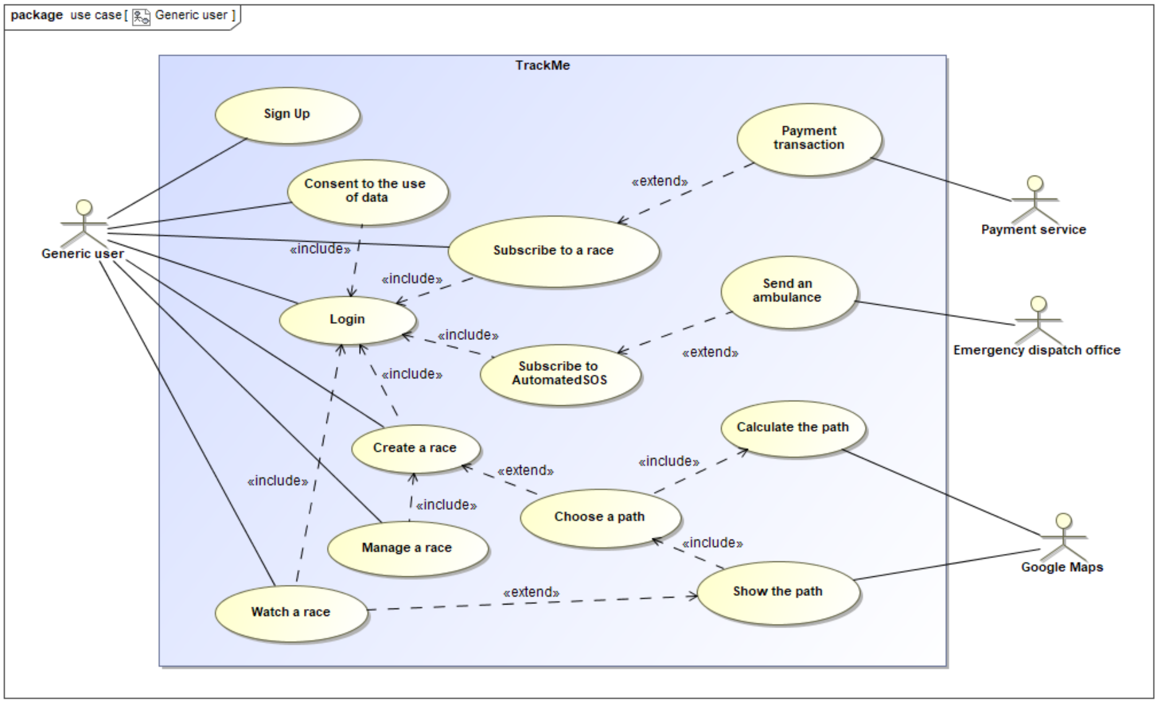


Figure 10: Use case diagram from user's point of view

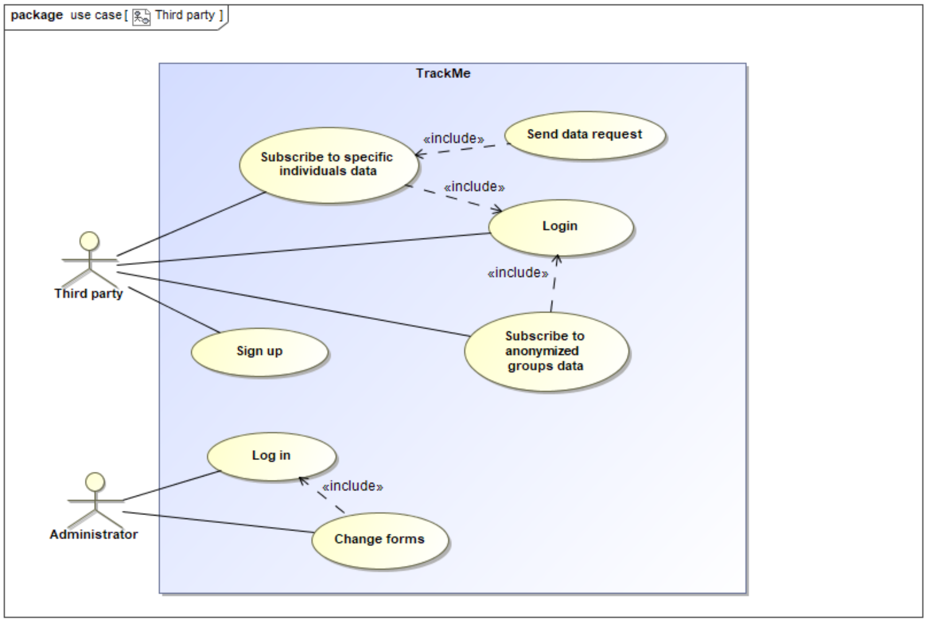

package use case [Third party]
TrackMe
Subscribe to specific individuals data
Send data request
Login
Third party
Sign up
Subscribe to anonymized groups data
Log in
Change forms
Administrator
«include»

Figure 11: Use case diagram from third party's point of view

Name	Sign Up
Actor	User, third party
Entry conditions	The user/third party has installed the application on the device
Events flow	1. Click on the ”CREATE AN ACCOUNT” button 2. Fill all the mandatory fields and provide the necessary information 3. Click on the ”CONFIRM” button 4. The system sends an e-mail of confirmation to the specified e-mail address and saves the data
Exit conditions	The user/third party has successfully registered and now he/she’s able to use the application
Exceptions	1. The user/third party already signed up 2. The user/third party didn’t fill all of the mandatory fields with valid data 3. The username is already taken 4. The e-mail is already registered All the exceptions are handled by notifying the user/third party and taking him/her back to the sign up activity

Name	Login
Actor	User, third party
Entry conditions	The user/third party is previously successfully signed up and has the application installed on the device
Events flow	1. Enter credentials in the “Username” and “Password” fields of the home page or choose one between the Google and Facebook autentication buttons 2. Click on the ”LOGIN” button [if the choice was to log in manually] 3. The user/third party is successfully logged in his/her profile and the system automatically redirects him/her to the related home page
Exit conditions	The user/third party is successfully redirected to the home page
Exceptions	1. The user/third party enters invalid username 2. The user/third party enters invalid password All the exceptions are handled by notifying the user/third party and taking him/her back to the login activity

Name	Consent to the use of data
Actor	User
Entry conditions	The user receives a push notification telling him that a specific third party wants to access to his data
Events flow	1. Click on the push notification 2. Click on the ”I AGREE” button 3. The user successfully allowed the usage of his data and the system redirects him to his home page
Exit conditions	The user is successfully redirected to the home page
Exceptions	1. The user swipe away the notification The exception is handled showing to the user the choice screen at the opening of the app

Name	Watch a race
Actor	User
Entry conditions	The user has already logged in
Events flow	1. Swipe right to show the side menu 2. Click on the ”Running races” voice 3. Search for a specific run using the search button on the upper bar or explore nearby races 4. Click on the ”OBSERVE” button 5. The user successfully chose the race to watch and the system saves the race in user’s ”Observing” section and shows the user the specific running race page
Exit conditions	The system saves the race in user’s ”Observing” section and shows the user the specific running race page
Exceptions	1. The user can’t find the race he is searching for

Name	Create a race
Actor	User
Entry conditions	The user has already logged in
Events flow	1. Swipe right to show the side menu 2. Click on the ”Running races” voice 3. Click on the ”+” button 4. Fill all the mandatory fields (name of the competition, date, time, start point, arrival point, date and time of expiration for registrations) 5. Choose one of the purposed routes 6. Choose between open or by invitation race selecting the corresponding radio button 7. Click on the ”CONFIRM” button 8. Select people to invite choosing them from the list which appears [if the choice was the by invitation race] 9. Click on the ”CONTINUE” button [if the choice was the by invitation race] 10. The user successfully created the race and the system saves the race in user’s ”Organizing” section and shows the user the specific running race page
Exit conditions	The system saves the race in user’s ”Organizing” section and shows the user the specific running race page
Exceptions	1. The user didn’t fill all of the mandatory fields with valid data 2. The distance between start and arrival point exceeds 50kms All the exceptions are handled by notifying the user and inviting him to modify data

Name	Manage a race
Actor	User
Entry conditions	The user has already logged in and he wants to modify a running race previously created
Events flow	1. Swipe right to show the side menu 2. Click on the ”Running races” voice 3. Switch to the ”YOUR RACES” tab 4. If the choice is to delete a race click on the ”DELETE” button, otherwise click on the ”MANAGE” button 5. Modify the information which need to be changed editing the relative field: start point, stages, arrival point, date, time, date and time of expiration for registrations, or choose a participant and click on the ”DISQUALIFY” button to mark him as ”disqualified” [if the chose was to manage] 6. Click on the ”CONFIRM” button [if the chose was to manage] 7. The user successfully deleted/modified the race and the system shows the user the specific running race page
Exit conditions	The system shows the user the specific running race page or the ”YOUR RACES” tab if the choice was to delete the race
Exceptions	1. The user didn’t fill all of the mandatory fields with valid data 2. The distance between start and arrival point exceeds 50kms All the exceptions are handled by notifying the user and inviting him to modify data

Name	Subscribe to a race
Actor	User
Entry conditions	The user has already logged in
Events flow	1. Swipe right to show the side menu 2. Click on the ”Running races” voice 3. Search for a specific run using the search button on the upper bar or explore nearby races 4. Click on the ”PARTICIPATE” button 5. If a payment is required it’s necessary to provide it 6. The user successfully chose the race to join and the system saves the race in user’s ”Participating” section and shows the user the specific running race page
Exit conditions	The system saves the race in user’s ”Participating” section and shows the user the specific running race page
Exceptions	1. The user can’t find the race he is searching for

Name	Subscribe to AutomatedSOS
Actor	User
Entry conditions	The user has already logged in
Events flow	1. Swipe right to show the side menu 2. Click on the ”Settings” voice 3. Scroll the page searching for the checkbox related to the AutomatedSOS subscription 4. Select the checkbox 5. Click on the ”SAVE” button to save preferences 6. The user successfully subscribed to AutomatedSOS and the system automatically redirects him to the home page
Exit conditions	The user is successfully redirected to the home page
Exceptions	/

Name	Choose a path
Actor	User
Entry conditions	The user is organizing a running race and he has already set correctly start and arrival point
Events flow	1. The user chooses one of the listed tracks 2. The system saves the data
Exit conditions	The chosen path is saved as track for the race
Exceptions	/

Name	Payment transaction
Actor	User
Entry conditions	The user has already logged in and wants to participate to a running race which requires a payment subscription
Events flow	1. The user chooses what kind of payment he wants to make 2. The user provides the system with all the required information for making a payment
Exit conditions	The payment is successful
Exceptions	1. The user did not enter a valid data

Name	Send an ambulance
Actor	TrackMe
Entry conditions	Some of the health status parameters of the user fall below specific thresholds (hearth rate, blood pressure, body temperature)
Events flow	1. The app transcript data about location and health status as a textual message 2. Using the GPS coordinates, the app recognizes the phone number to contact (112 or 911) 3. The app calls the phone number and concurrently sends user’s data via SMS
Exit conditions	Someone answers the call
Exceptions	/

Name	Calculate the path
Actor	TrackMe
Entry conditions	The user wants to participate to a running race and has already selected start point and arrival point (and optionally intermediate stages)
Events flow	1. The app gathers all the information provided by external map services 2. The app calculates routes according with user's specifications 3. The system saves the data
Exit conditions	The calculated routes are presented to the user
Exceptions	/

Name	Show the path
Actor	TrackMe
Entry conditions	The user clicks on a running race
Events flow	1. The app gathers all the information provided by external map services 2. Comparing date and time of the event with actual date and time, the app shows: – An empty leaderboard and an empty track if the actual date and time precede those of the expiration for registrations – The leaderboard with participants' info and their markers at the start point of the track if the actual date and time preced those of the race – The leaderboard with participants' info and their markers at the arrival point of the track if the actual date and time follow those of the end of the race – If the race is in progress, the app gathers all the information about participants' position provided by their devices and shows their markers moving on the track, upgrading the leaderboard
Exit conditions	The race screen is correctly shown
Exceptions	1. One or more users' devices are broken and can't provide an accurate GPS position The exception is managed leaving user's marker in the position of the last correct measurement and considering him as stopped (organizators will manually adjust the leaderboard in the end)

Name	Subscribe to specific individuals data
Actor	Third party
Entry conditions	The third party has already logged in
Events flow	1. Search for a fiscal code using the search bar in the home page 2. Click on the username associated with the searched fiscal code 3. Click on the "OK" button in the confirmation message which is displayed 4. The system sends a push notification to the selected user, who can accept or refuse the sharing of his data 5. Click on the push notification received from the user to view his fist data if he had consented, or the rejection message otherwise 6. Choose the radio button corresponding to the frequency with which obtaining data from the user (once every hour, day, week or month) [if the user allowed the data sharing] 7. Click on the "SAVE" button [if the user allowed the data sharing] 8. The system saves a link to the users' data in third party's "Your subscriptions" section of the home page [if the user allowed the data sharing] and redirects the third party to the home page
Exit conditions	The system saves a link to the users' data in third party's "Your subscriptions" section and redirects the third party to the home page successfully
Exceptions	1. The fiscal code doesn't exist in the data base 2. The user refuses the data sharing

Name	Subscribe to anonymized groups data
Actor	Third party
Entry conditions	The third party has already logged in
Events flow	1. Select one of the suggested search filters in the home page by clicking it 2. Set the filter specifying the data (e.g. "Milan" for the location filter) 3. [Optional] Repeat the previous steps choosing other filters 4. Scroll the page and select one of the displayed results 5. Choose the radio button corresponding to the frequency with which obtaining data from the group (once every hour, day, week or month) in the confirmation message which is displayed 6. Click on the "SUBSCRIBE" button in the confirmation message 7. The system saves a link to the group's data in third party's "Your subscriptions" section of the home page and redirects the third party to the home page
Exit conditions	The system saves a link to the group's data in third party's "Your subscriptions" section and redirects the third party to the home page successfully
Exceptions	1. No groups are shown (there are less then 1000 results and a group can't be created)

Name	Send data request
Actor	TrackMe
Entry conditions	A third party has already selected a user from which to obtain data
Events flow	1. The app retrieves the user’s profile from the data base 2. The app creates an association table with the request and the user’s profile 3. The app sends to the user a push notification 4. Once the user clicks the button corresponding to his decision to share or not his data, the app retrieves the third party’s profile from the request using the association table 5. The system sends to the third party a push notification with user’s data or with a message of refuse
Exit conditions	The third party receives successfully the push notification
Exceptions	/

Name	Log in
Actor	Administrator
Entry conditions	/
Events flow	1. The administrator enters his/her credentials in the “Username” and “Password” fields of the home page 2. The administrator clicks on the ”LOGIN” button 3. The administrator is successfully logged in his/her profile and the system automatically redirects him/her to the related home page
Exit conditions	The administrator is successfully redirected to the options menu
Exceptions	1. The administrator enters invalid username 2. The administrator enters invalid password All the exceptions are handled by notifying the administrator and taking him/her back to the login activity

Name	Change forms
Actor	Administrator
Entry conditions	The administrator has already logged in
Events flow	1. The administrator opens the option menu 2. The administrator chooses the option ”Change forms” 3. The administrator selects one form to modify 4. The administrator can add/remove/modify fields and buttons 5. The administrator clicks on the ”SAVE” button 6. The administrator clicks on the ”PROCEED” button in the confirmation message which is displayed 7. The system saves the data
Exit conditions	The selected form is modified
Exceptions	/

3.3.2 Sequence diagrams

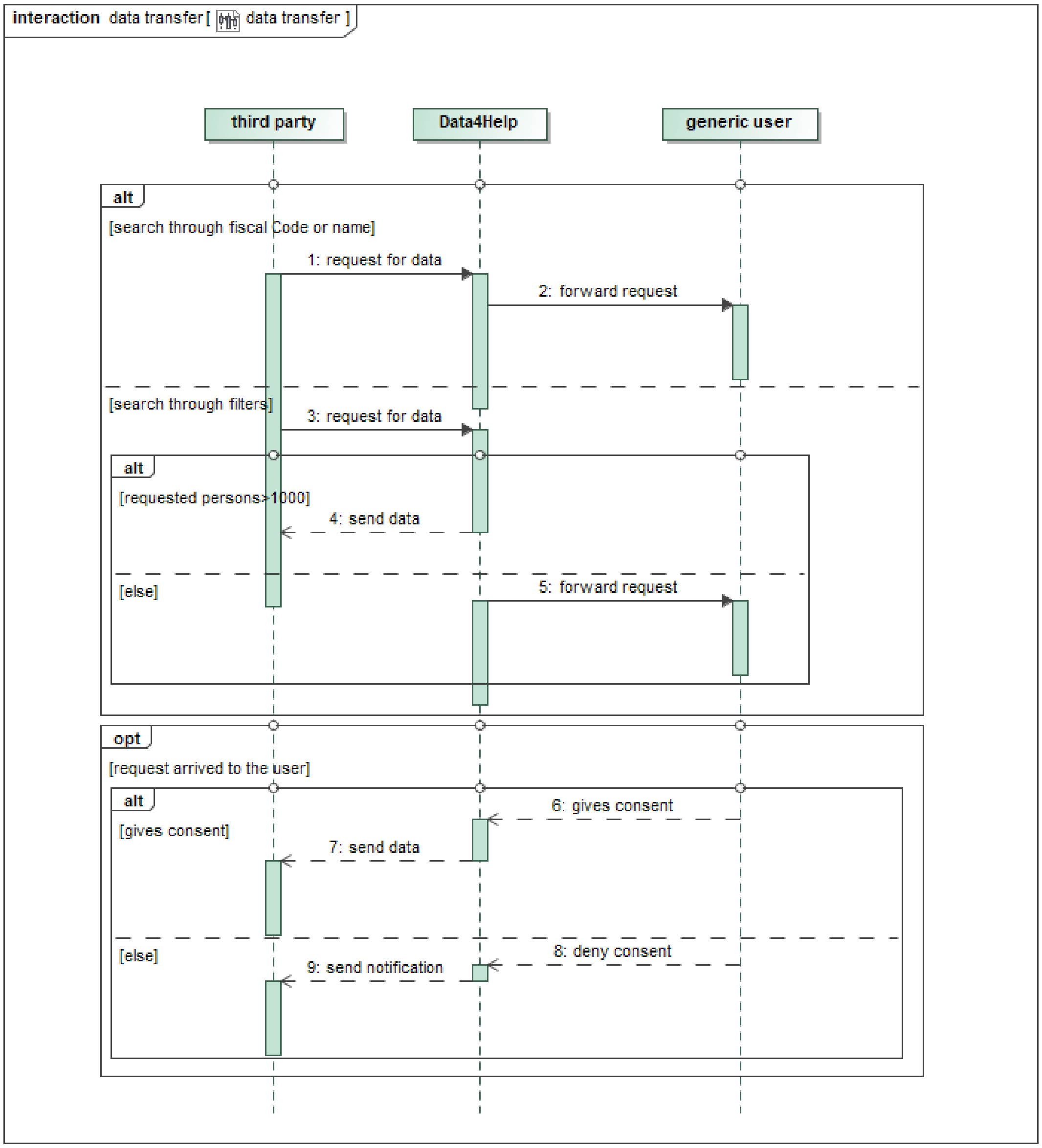


Figure 12: Sequence diagram of data transfer by Data4Help

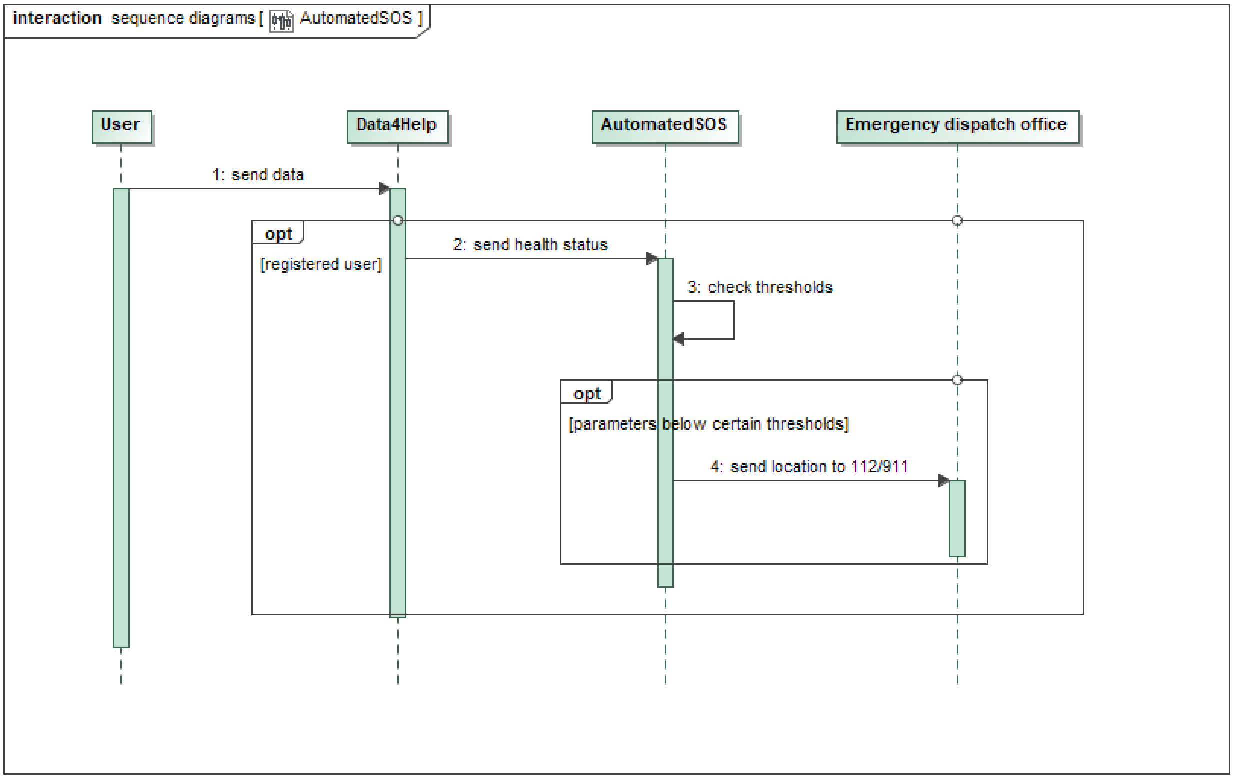

interaction sequence diagrams [AutomatedSOS]
User
Data4Help
AutomatedSOS
Emergency dispatch office
1: send data
opt
[registered user]
2: send health status
3: check thresholds
opt
[parameters below certain thresholds]
4: send location to 112/911

Figure 13: Sequence diagram of the whole AutomatedSOS service

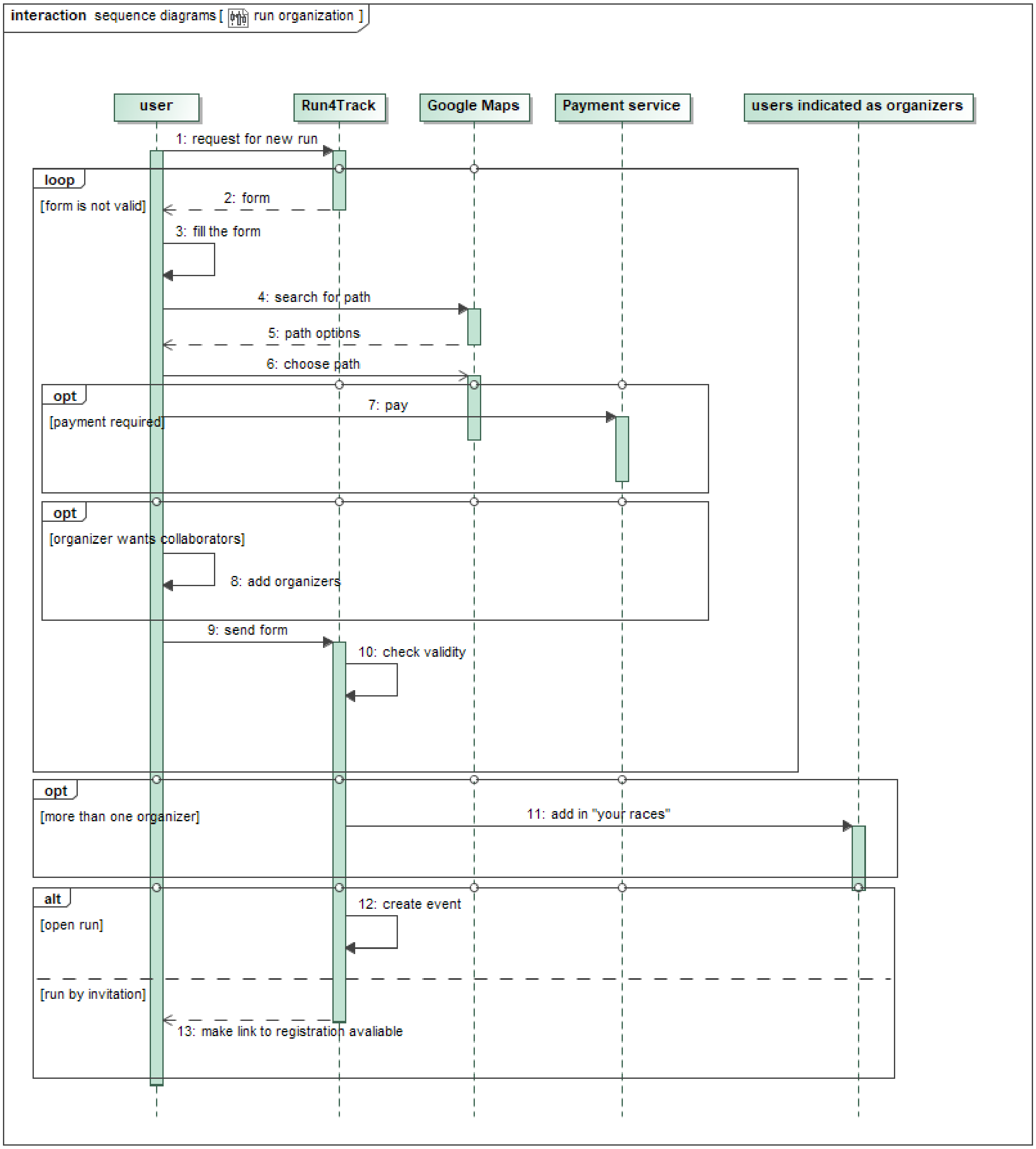


Figure 14: Sequence diagram of a run rganization by Track4Run

3.4 Performance requirements

The system is provided to serve fairly great number of users simultaneously. Because of the nature of the application, especially for what concerns the AutomatedSOS service, we have to guarantee quick, reactive and correct responses. When health status parameters fall below certain thresholds the app guarantees a reaction time of less then 5 seconds to contact the NUE (or 911).

3.5 Design constraints

3.5.1 Standards compliance

- The app requests only the absolute minimum permissions that it needs to support core functionalities.
- The mobile app supports both landscape and portrait orientations (if possible). Orientations expose largely the same features and actions and preserve functional parity. Minor changes in content or views are acceptable. The smartwatch app fits to the dimension of the device.

3.5.2 Hardware limitations

As previously mentioned in the ”Hardware interfaces” section, the app requires a device with a GPS sensor and a wearable device with at least the following sensors: GPS, accelerometer, altimeter, gyroscope, heart rate sensor (biosensor related to the detection of glucose level, blood pressure, body temperature and ECG also supported and recommended). Both iOS and Android are supported, and the connection required can be based on the following technologies: 2G, 3G, 4G, LTE.

3.5.3 Other constraints

Regulatory policies are respected: the app share user's data just with third parties allowed by the user himself, and he can also manage his concessions. Data are not used for commercial purpose.

User's payment informations could be asked in particular situations (organization of some special running races) but the app will use them only for fees and rides payments and will not share them with anyone (neither third parties allowed to receive user's data).

3.6 Software system attributes

3.6.1 Reliability

The application must be available 24/7. Because of the nature of the app, especially for what concerns the AutomatedSOS service, deviations are not tolerated and system's maintenances must be communicated to users (both personal and third parties accounts) via mail at least 1 day before.

3.6.2 Availability

In order to guarantee high degree of availability, system of redundant servers may be considered. In this way, if possibly one server fails, the other one will be ready to take over. The system is expected to be available 99.99% of the time.

3.6.3 Security

Users' credentials and payments informations should be confidentially stored and encrypted. Security of the data and of the communications between users and app is a primary concern.

3.6.4 Maintainability

In order to guarantee a good maintainability of the app, the code will be simplified as much as possible, and always commented both with natural language (english) and a first order logic based language (when possible). Design patterns will be used and all the documentation will be provided. Furthermore, the app will always been tested before a new release and changelogs will be generated for each new version of the system.

3.6.5 Compatibility

- Android versions supported: 5.0 (Lollipop) and following.
- iOS versions supported: iOS 7 and following.
- WearOS (related to Android watches) and watchOS (related to Apple watches) supported in all versions.

4 Formal analysis using Alloy

In this section, the Alloy model is given. Using it, some of the features of the system are specified and explained in more details, with the focus being on the constraints:

- The organizer must create a ”valid” competition: track with well-defined start and finish, well-defined intermediate stages, maximum number of stages is 10 (track’s lenght of a run can be at max 50km).
- Data sent to a third party must be part of or data of a specific individual or an anonymous group of at least 1000 subjects.
- Generic users that are in a specific individual data survey will recieve a notification.
- If at least one between BPM, blood pressure or glucose of a user will exceed minimum or maximum threshold, an Emergency dispatcher will recieve a warning (CallAmbulanceWarning), that will be used to send an ambulance to the user who needs help.
- Users can participate to competitions and can be marked as disqualified by the organizer, and if an athlete is marked as disqualified he will not appear in the list of active runners.

In the model of Alloy described we have simplified the numerical values, for example the races can be a maximum of 7 (= 50km) and there con be a maximum of 7 stages (= 10), the position, coordX and coordY, are limited respectively between -3 and 3, between -6 and 6. The number of people in a survey that indicates that it can be an anonymized data survey is 2 (= 1000). BPM, blood pressure, glucose are represented by values between 1 and 6, so it is for thresholds.

4.1 Alloy model

```
open util/integer
open util/boolean

sig Position{
coordX: one Int,
coordY: one Int,
}
{ coordX ≥ -3 and coordX =< 3 and coordY ≥ -6 and coordY =< 6 }

sig Track{
startPoint: one Position,
endPoint: one Position,
trackLenght: one Int,
stages: set Position
}
{((startPoint.coordX = endPoint.coordX and startPoint.coordY = endPoint.coordY) implies #
    ↪ stages > 0) and
        (trackLenght > 0 and trackLenght =< 7) and (#stages =< 7) and
        (#stages > 0 implies (startPoint not in stages and endPoint not in stages))}

abstract sig Data{}

sig BPM extends Data{
value: one Int,
thrMax: one Int,
thrMin: one Int
}
```

```
{(value > 0 and value < 7) and (thrMax > 0 and thrMax < 7) and
        (thrMin > 0 and thrMin < 7) and  thrMin < thrMax}

sig BloodPressure extends Data{
value: one Int,
thrMax: one Int,
thrMin: one Int
}
{(value > 0 and value < 7) and (thrMax > 0 and thrMax < 7) and
        (thrMin > 0 and thrMin < 7) and thrMin < thrMax}

sig Steps extends Data{}

sig Temperature extends Data{}

sig Glucose extends Data{
value: one Int,
thrMax: one Int,
thrMin: one Int
}
{(value > 0 and value < 7) and (thrMax > 0 and thrMax < 7) and
        (thrMin > 0 and thrMin < 7) and thrMin < thrMax}

sig IdentifyingData {}

sig GenericUser{
dataCollected: one UsersData,
personalData: one IdentifyingData,
surveysAllowed: set SpecificIndividualsDataSurvey,
notifications: set Notification,
runsScheduled: set Run
}

sig UsersData{
bpmUser: one BPM,
bloodPressureUser: one BloodPressure,
stepsUser: one Steps,
temperatureUser: one Temperature,
glucoseUser: one Glucose,
locationUser: one Position
}

sig Athlete{
userAthlete: one GenericUser,
disqualified: one Bool,
attendant: one Bool
}

sig SurveyID{}

abstract sig Survey{
surveyID: one SurveyID,
involvedThirdParty: one ThirdParty,
researchResult: one Int
}
{researchResult > 0 and researchResult < 7}

sig AnonymizedDataSurvey extends Survey{
genericData: set UsersData
}
{researchResult > 2 and #genericData = researchResult}

sig SpecificIndividualsDataSurvey extends Survey{
specificData: set GenericUser
}
```

```
{researchResult =< 2 and #specificData = researchResult}

abstract sig Notification{}

sig SignedInASurvey extends Notification{
surveyInvolved: one Survey
}

sig CallAmbulanceWarning{
userWhoNeedsHelp: one GenericUser,
warningID: one WarningID
}

sig WarningID{}

sig ThirdParty{
anonymSurveysCarriedOut: set AnonymizedDataSurvey,
specificSurveyCarriedOut: set SpecificIndividualsDataSurvey
}

sig EmergencyDispatcher{
warnings: set CallAmbulanceWarning,
}

abstract sig Run{
track: one Track,
organizer: some GenericUser,
spectators: set GenericUser,
activeAth: set Athlete,
subscribedAth: set Athlete
}

sig ByInvitationRun extends Run{
invitatedAthletes: set Athlete
}

sig OpenRun extends Run{}

--------------------------------------------------------------------------------------------

fact SurveysIdAreUnique{all s1, s2: Survey|
        s1.surveyID = s2.surveyID implies s1 = s2}

fact WarningIDAreUnique{all caw1, caw2: CallAmbulanceWarning|
        caw1.warningID = caw2.warningID implies caw1 = caw2}

fact IdentifyingDataAreUnique{all usr1, usr2: GenericUser|
        usr1.personalData = usr2.personalData implies usr1 = usr2}

fact NotificationSentToUserCausedBySurvey{
all user: GenericUser, specificSurvey:  SpecificIndividualsDataSurvey|
        (user in specificSurvey.specificData implies
                 (one sn1:  SignedInASurvey| sn1 in user.notifications and sn1.
                     ↪ surveyInvolved.surveyID = specificSurvey.surveyID)) and
        (user not in specificSurvey.specificData implies
                (no sn2: SignedInASurvey| sn2 in user.notifications and sn2.surveyInvolved.
                    ↪ surveyID = specificSurvey.surveyID))
}

fact EmergencyDispathcerGetCallAmbulanceWarning{
all user: GenericUser|
        (       (user.dataCollected.bpmUser.value < user.dataCollected.bpmUser.thrMin or
            ↪ user.dataCollected.bpmUser.value > user.dataCollected.bpmUser.thrMax) or
                (user.dataCollected.bloodPressureUser.value < user.dataCollected.
                    ↪ bloodPressureUser.thrMin or user.dataCollected.bloodPressureUser.
                    ↪ value > user.dataCollected.bloodPressureUser.thrMax) or
                (user.dataCollected.glucoseUser.value < user.dataCollected.glucoseUser.
                    ↪ thrMin or user.dataCollected.glucoseUser.value > user.dataCollected.
                    ↪ glucoseUser.thrMax)
        ) iff
        (one ed: EmergencyDispatcher, aw: CallAmbulanceWarning| aw in ed.warnings and user
            ↪ in aw.userWhoNeedsHelp)
}

fact DisqualifiedUsersAreNotActive{
all a: Athlete| some r: Run|
        (not isTrue[a.disqualified] and isTrue[a.attendant]) implies (a in r.activeAth)
}

fact ActiveAthAreSubscribed{
all a: Athlete| some r: Run|
        (a in r.activeAth implies a in r.subscribedAth) and
        (a in r.subscribedAth)
}

fact AthAreUniqueInRuns{
all a1, a2: Athlete| some r: Run|
        (a1 in r.subscribedAth and a2 in r.subscribedAth) implies a1 = a2
}


fact NoNotificForAnonSurvey{all n: SignedInASurvey| some sp: SpecificIndividualsDataSurvey|
         sp in n.surveyInvolved}

fact UsersDataExistWithUsers{all d: UsersData| some u: GenericUser|
        d in u.dataCollected}

fact TrackExistOnlyInRuns{all t: Track| some r: Run|
        t in r.track}

fact IdentifDataExistWithUsers{all id: IdentifyingData| some u: GenericUser|
        id in u.personalData}

fact SurveyIdExistWIthSurvey{all sid: SurveyID| some s: Survey|
        sid in s.surveyID}

fact NotificExistWithUsers{all n: SignedInASurvey| some u: GenericUser|
        n in u.notifications}

fact WarningIdExistWithWarning{all wid: WarningID| some caw: CallAmbulanceWarning|
        wid in caw.warningID}

fact WarningsAreGivenToDispatcher{all caw: CallAmbulanceWarning| some ed:
    ↪ EmergencyDispatcher|
        caw in ed.warnings}

fact PositionExistInSthElse{all p: Position| some t: Track, ud: UsersData|
        (p in t.startPoint or p in t.endPoint or p in t.stages) or (p in ud.locationUser)}

fact BPMExistInSthElse{all bpm: BPM| some ud: UsersData|
        bpm in ud.bpmUser}

fact BloodPressExistInSthElse{all blp: BloodPressure| some ud: UsersData|
        blp in ud.bloodPressureUser}

fact GlucoseExistInSthElse{all g: Glucose| some ud: UsersData|
        g in ud.glucoseUser}
-----------------------------------------------------------------------------------------

pred showData4Help{
```

```
all u: GenericUser| some t: ThirdParty| some sp: SpecificIndividualsDataSurvey|
         (sp.specificData = u and sp.involvedThirdParty = t)
#GenericUser = 3
#AnonymizedDataSurvey = 1
}

pred showAutomatedSOS{
some u: GenericUser|
        (u.dataCollected.bpmUser.value < u.dataCollected.bpmUser.thrMin or
        u.dataCollected.bloodPressureUser.value < u.dataCollected.bloodPressureUser.thrMin
            ↪ or
        u.dataCollected.glucoseUser.value < u.dataCollected.glucoseUser.thrMin)
#GenericUser = 1
#ThirdParty = 0
#EmergencyDispatcher = 1
}

pred showTrack4Run{
some ath: Athlete|
        (isTrue[ath.disqualified] and isTrue[ath.attendant]) or
        (isTrue[ath.disqualified] and not isTrue[ath.attendant]) or
        (not isTrue[ath.disqualified] and isTrue[ath.attendant])
#GenericUser = 3
#ThirdParty = 0
}

---------------------------------------------------------------------------------------------
run showData4Help for 5 but 1 Run, 0 EmergencyDispatcher, 0 CallAmbulanceWarning

run showAutomatedSOS for 5 but 1 Run

run showTrack4Run for 5 but 1 EmergencyDispatcher, 3 Run
```

4.2 World generated

Figure 15 shows a possible world generated with the run of the second pred (run showAutomatedSOS). On top of the image there is an ”OpenRun” and at the bottom (under Track) there is the track’s lenght (7).

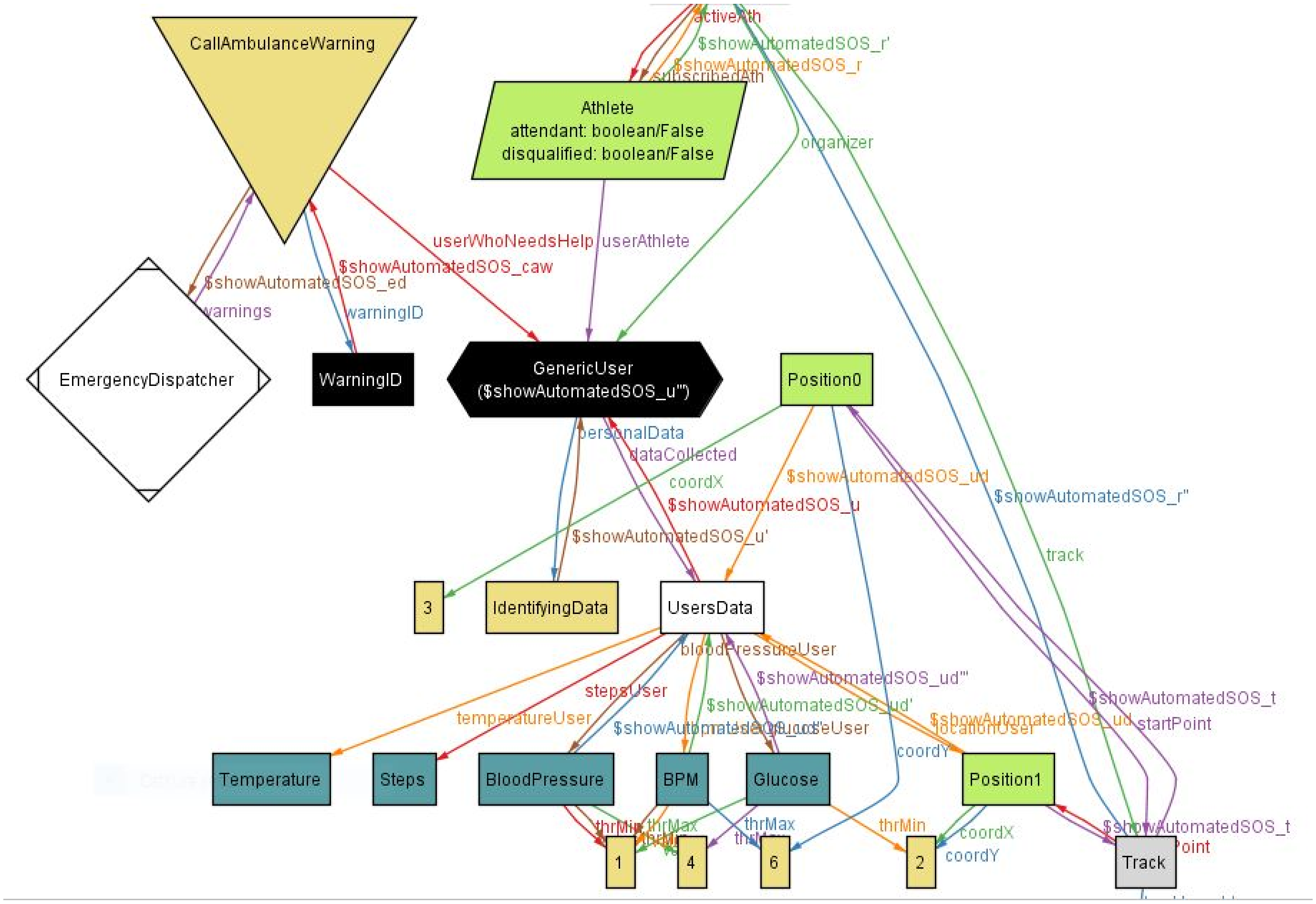
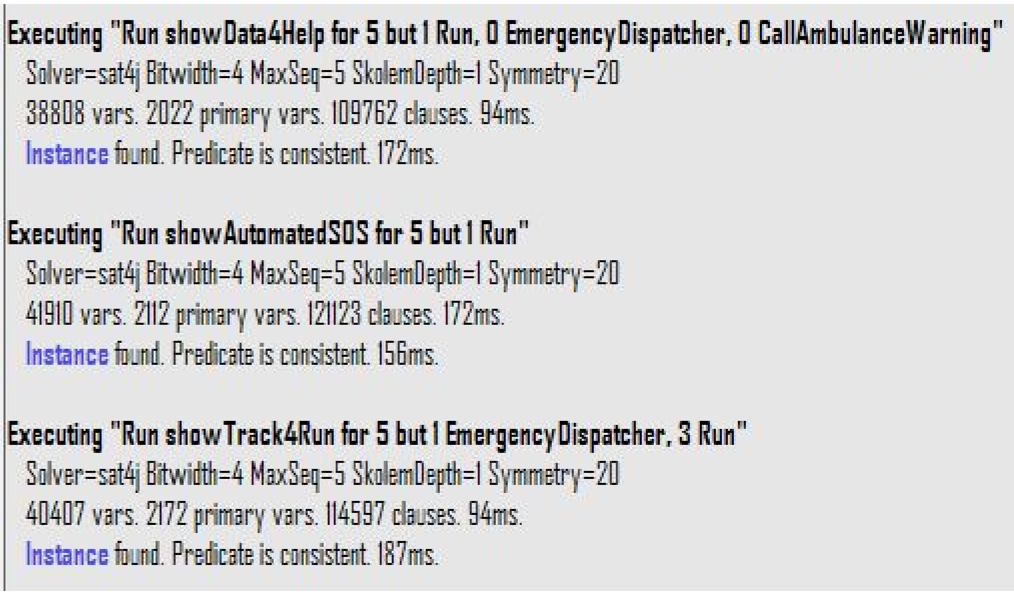


Figure 15: World generated

4.3 Alloy results

These are the results produced by the code shown above.



```
Executing "Run showData4Help for 5 but 1 Run, 0 EmergencyDispatcher, 0 CallAmbulanceWarning"
  Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
  38808 vars. 2022 primary vars. 109762 clauses. 94ms.
  Instance found. Predicate is consistent. 172ms.

Executing "Run showAutomatedSOS for 5 but 1 Run"
  Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
  41910 vars. 2112 primary vars. 121123 clauses. 172ms.
  Instance found. Predicate is consistent. 156ms.

Executing "Run showTrack4Run for 5 but 1 EmergencyDispatcher, 3 Run"
  Solver=sat4j Bitwidth=4 MaxSeq=5 SkolemDepth=1 Symmetry=20
  40407 vars. 2172 primary vars. 114597 clauses. 94ms.
  Instance found. Predicate is consistent. 187ms.
```

Figure 16: Alloy results

5 Effort spent

5.1 Claudia Conchetto

Description of the task	**Approximate hours**
Assumptions, dependencies and constraints	4
Product perspective	7
Product functions, user characteristics	4
Functional requirements	10
Formal analysis using Alloy	8

5.2 Riccardo Corona

Description of the task	**Approximate hours**
Assumptions, dependencies and constraints	4
External interface requirements	9
Scenarios	1
Functional requirements	10
Design constraints	1
Software system attributes	1

5.3 Andrea Crivellin

Description of the task	**Approximate hours**
Introduction	4
Assumptions, dependencies and constraints	4
Functional requirements	4
Formal analysis using Alloy	15

6 References

- ”Mandatory Project Assignment AY 2018-2019” specification document, available on BeeP’s course channel.
- Slides about the structure of the RASD, available on BeepP’s course channel.
- Alloy documentation `http://alloy.lcs.mit.edu/alloy/documentation.html`, in addition to slides available on BeeP’s course channel.