POLITECNICO DI MILANO

School of Industrial and Information Engineering

Computer Science and Engineering

POLITECNICO
MILANO 1863

TRACKME DD
Design Document

Software Engineering 2 Project

The project was made by

Luca Alessandrelli 846260
Andrea Caraffa 919970
Andrea Bionda 921082

Version 1.0 - 2018/2019

Deliverable:	DD
Title:	Design Document
Authors:	Luca Alessandrelli, Andrea Caraffa, Andrea Bionda
Version:	1.0
Date:	23-November-2018
Download page:	https://github.com/lucaalessandrelli/AlessandrelliCaraffaBionda.git
Copyright:	

Contents

1 Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, Abbreviations

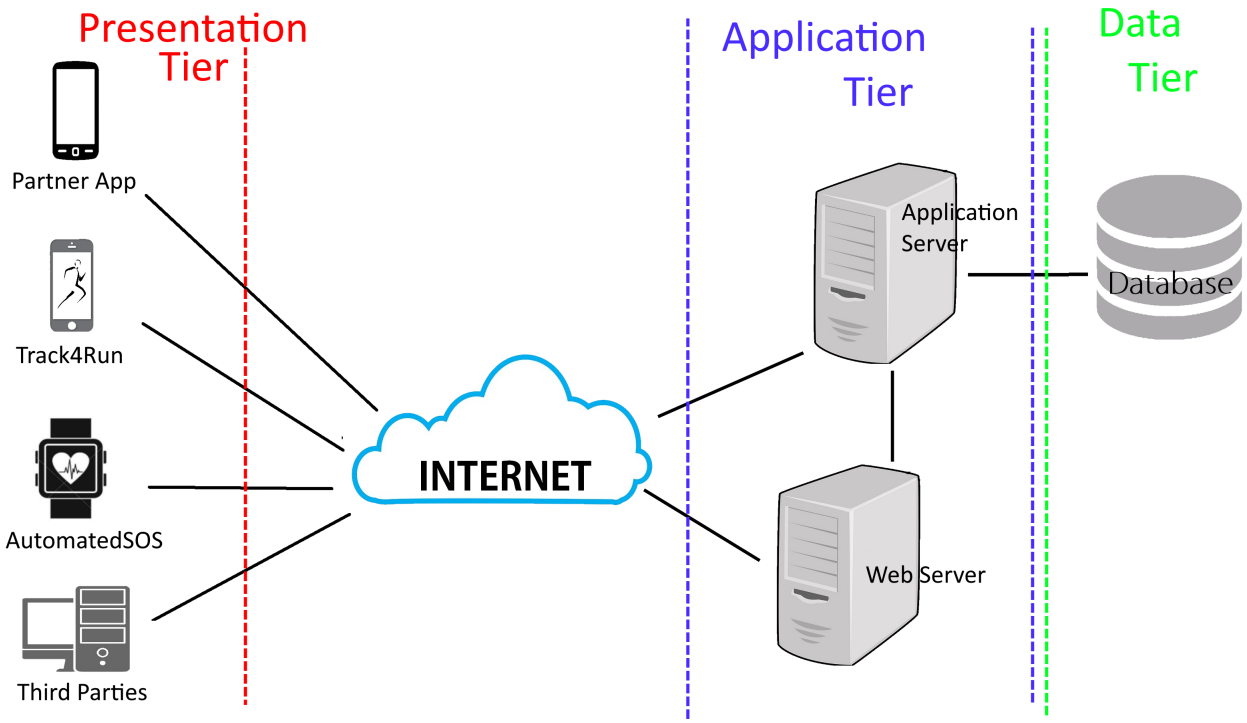
1.4 Revision History

1.5 Document Structure

2 Architectural Design

2.1 Overview

The TrackMe services are built on a client-server structure, this way the system is organized through abstraction levels. We chose to adopt a 3-tier architecture:



- **Presentation Tier**
  This layer makes the interaction possible between the user and the system. Here the user sees all the information provided by the system in a easily way to understand them.

- **Application Tier**
  This layer is managed almost totally by Data4Help service that is in charge of:

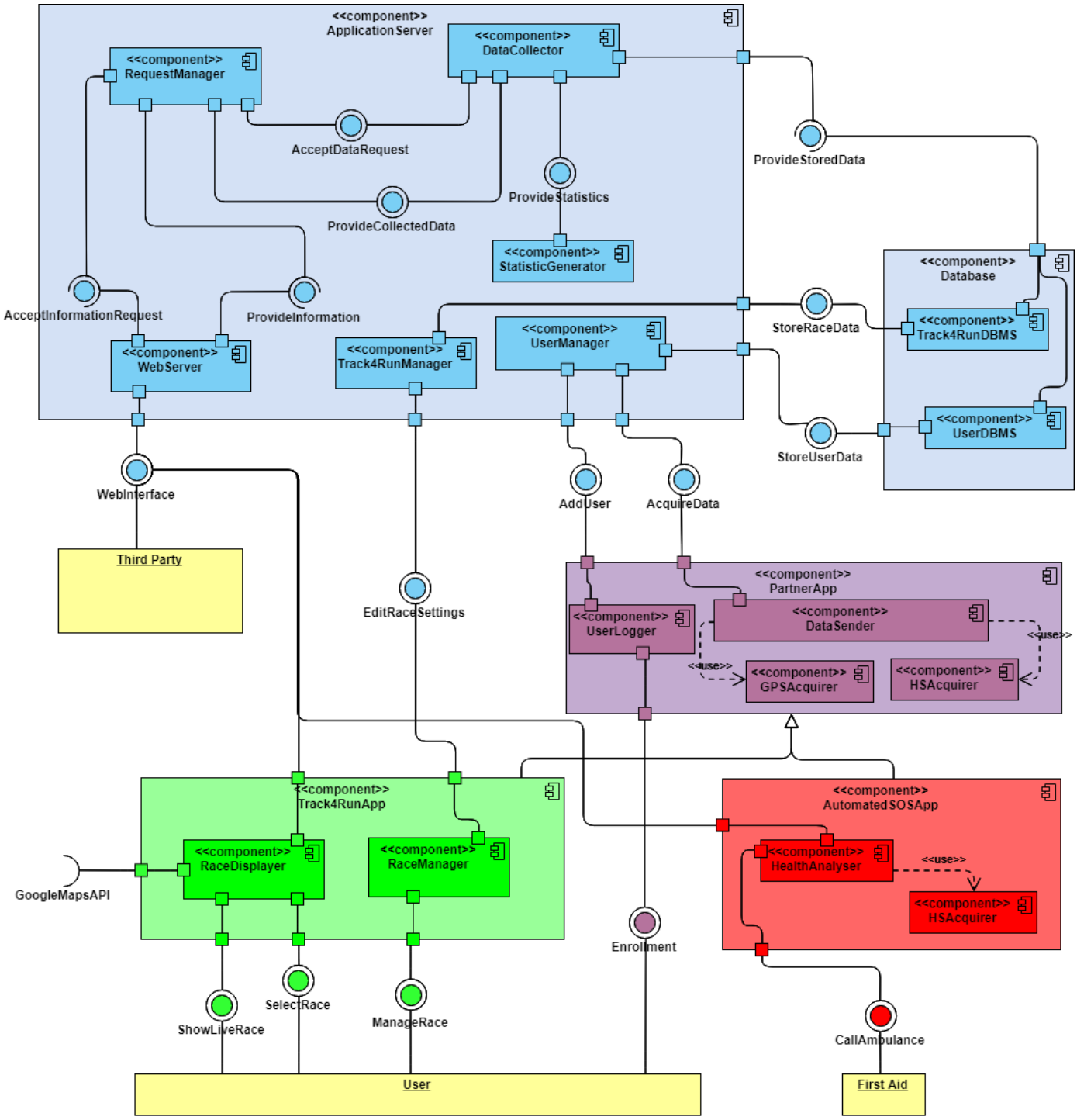
  - store data incoming from the external;
  - collect data information from database in order to execute Third parties' requests;
  - also generates data statistics on data collected;
  - send to third parties requested data.

  Moreover even AutomatedSOS has logic application in order to continuously monitor users' health status.

- **Data Tier**
  In this layer all the sensible users' data (location, health status) are stored into Databases and are retrieved by the application tier in order to do statistics and answer third parties' requests.

More specifically Data4Help manage the data and core logic sections while AutomatedSOS and Track4Run manage the presentation section. Actually, a small part of application tier is also present in AutomatedSOS, this is due to the fact that the Health Monitoring process requires to be executed as fast as possible.

2.2 Component View



Component diagram description

1 **ApplicationServer** This huge component is in charge of manage Data4Help services so store and provide, for interested companies, users' data (such as GPS location and daily Health Status). Moreover it manages race information of Track4Run application.

1.1 **WebServer:** In order to accept and supply information to who need them, this component offer a friendly web interface to simplify these operations

1.2 **RequestManager:** In order to support web server in its job, this component manage all the incoming request: it sorts them per urgency, it wrap the request in a smart data structure

and send it to Data Collector, it unwrap the answer from Data Collector and it continuously generates data exchange if a live acquisition is active

1.3 **DataCollector:** This component is in charge to communicate with Database in order to retrieve information and supply them to Request Manager. To perform these operations the component should receive request from Request Manger, unwrap it, creates a query to database that cover the question and send it to database; once the database has responded it should wrap the answer and provide it to Request Manager. Moreover if a statistic on data is required it requires it to Statistic Generator.

1.4 **StatisticGenerator:** This component is in charge of generates statistic on data provided by data collector such as arithmetic mean, variance from average, standard deviation and median.

1.5 **UserManager:** This component is in charge to communicate with users' device in order to log (or sign up) user and retrieve data collected.

1.6 **Track4RunManager:** This component is in charge to communicate with Track4Run applications in order to allow promoter to manage races.

2 **Database** This component is in charge of store physically data and organize them in a smart way according to DBMS rule, moreover it allows the access to those data.

2.1 **UserDBMS:** This component is in charge of store all the users' acquired data and quickly provide them to Data collector whenever it is required.

2.2 **Track4RunDBMS:** This component is in charge of store all the races' informations provided by promoters.

3 **PartnerApp:** This component implements how the application partner of TrackMe is basically structured, from the components in charge of retrieve raw data from device's API to who is in charge to communicate with the Main Server. This component is extended by all the partner applications that want to exploit Data4Halp service, so even by AutomatedSOS and Track4Run.

3.1 **GPSAcquirer:** This component is developed in order to acquire GPS location from user's device at constant time period.

3.2 **HSAcquirer:** This component is developed in order to acquire Hearth rate, Blood pressure and Calories consumed from user's device at constant time period. (Obviously if device support it).

3.3 **UserLogger:** This component offers to client the possibility to become Data4Help user, so it is in charge to show to client the policy to accept and acquire all the credentials inserted during registration. Moreover has to communicate to Main Server the registration of a new user.

3.3 **DataSender:** This component, exploiting GPSAcquirer and HSAcquirer features, is in charge to provide to UserManager component inside main server all the retrieved data whenever are ready to be sent.

4 **AutomatedSOSApp:** This component should extend all that is specified in partner application component to exploit all its features. AutomatedSOS component,also, has to use (as Data Sender) the HSAcquirer in order to check health status parameter and call the first aid whenever is required.

4.1 **HealthAnalyser:** This component exploiting HSAcquirer features is in charge to analyse continuously health parameters, compare last acquired data with historical data in order to improve the reaction time, check data in order to prevent user's diseases and call an ambulance whenever such parameters are below a certain threshold, compiling and providing also a special report.

4.2 **HSAcquirer:** This component is developed in order to acquire Hearth rate, Blood pressure and Calories consumed from user's device at constant time period. (Obviously if device support it).

5 **Track4RunApp:** This component should extend all that is specified in partner application component to exploit all its features. Track4Run component,also, should provide to spectator the possibilities to select and spectate to a run,also allows promoter to create and manage a race providing all the useful information.

5.1 **RaceDisplayer:** This internal component is in charge to provide a human interface to the user that allows him to specify the race that he want to spectate and then show the position of all the athletes in the race, exploiting Google Maps API.

5.2 **RaceManager:** This internal component is in charge to provide to user the possibility to promote a run inside the system, date,path,name,maximum number of partecipants and description. Moreover a promoter can invite specific users providing their fiscal code.

2.3 Deployment View

The following Deployment Diagram captures the topology of the system's hardware. The SmartphoneApp and SmartWatchApp (Presentation Tier) communicate to the Application Server through RMI, while the WebBrowser communicates to the WebServer through HTTP protocol. The Application Server (Application Tier) communicates to the Database Server (Data Tier) through JDBC.

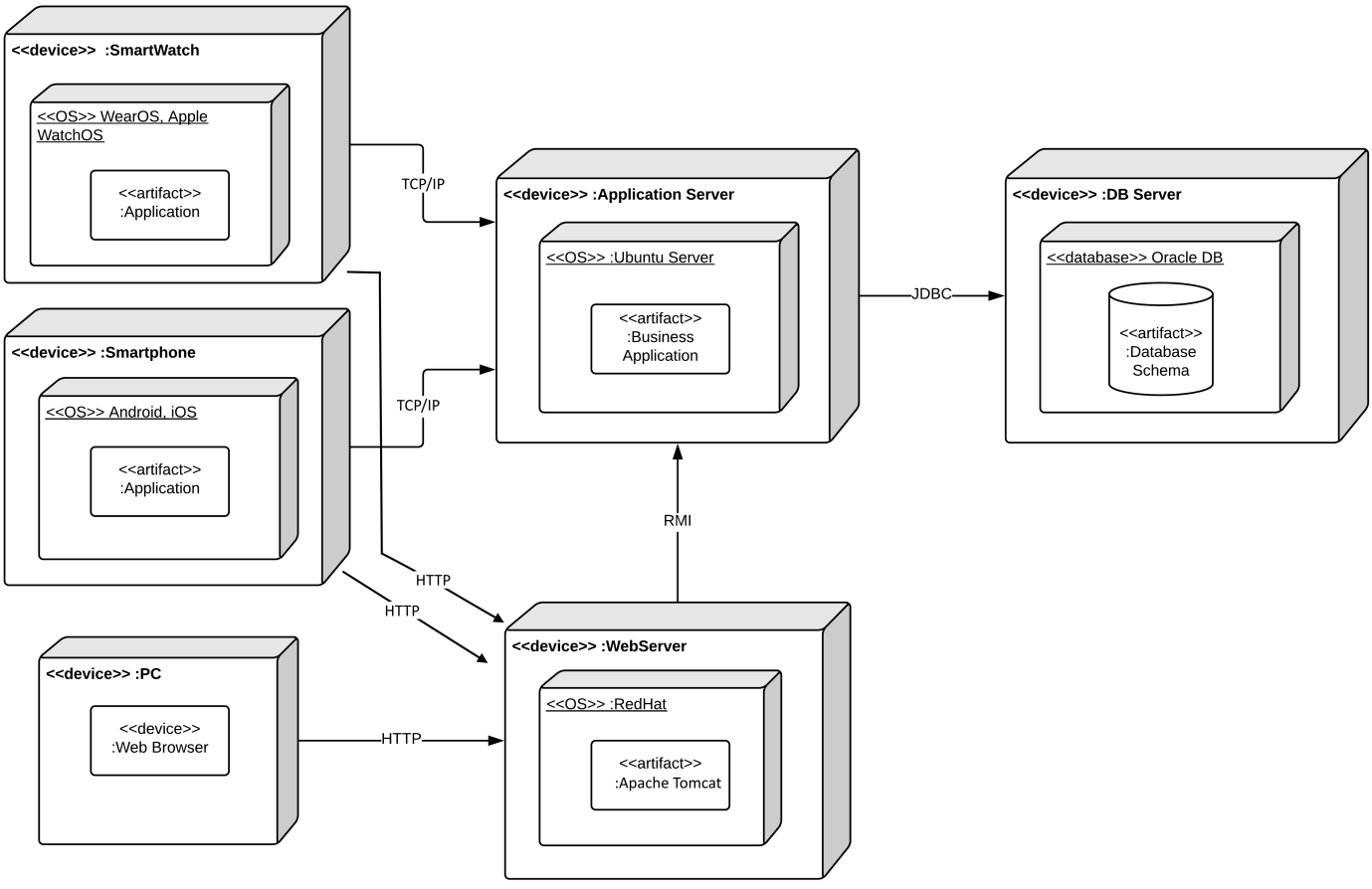


Figure 1: Deployment Diagram.

2.4 Runtime View

In this section several Sequence Diagrams are shown in order to point up the interaction among components and their behavior in particular scenarios.

2.4.1 Individual Request

2.4.2

2.4.3

2.5 Component Interfaces

1.1 **WebServer:**

1.1.1 **WebInterface:** Human interface for third parties that want to log in or retrieve information from Data4Help service. Software interface for Track4Run and for AutomatedSOS in order to provide requested information.
boolean: thirdPartyRegistration(String: Name,String Piva)
boolean: thirdPartyLogIn(String: Name)
void: makeRequests (Request[ ] requests)

1.2 **RequestManager:**

1.2.1 **AcceptInformationRequest:** Software interface that allows WebServer to submit requests to be evaluated.

1.2.1 **ProvideInformation:** Software interface that provides to web server information answers that match previous requests.

1.3 **DataCollector:**

1.3.1 **AcceptdataRequest:** Software interface offered by DataCollector that accept requests from RequestManager formatted in the proper way.

1.3.1 **ProvideInformation:** Software interface offered by DataCollector that provide to Request Manager information answer formatted in the proper way.

1.4 **StatisticGenerator:**

1.4.1 **ProvideStatistics:** Software interface that provides statistic values to data collector. (The Software interface to receive data from Data collector to be managed is trivial and not explicitly specified).

1.5 **UserManager:**

1.5.1 **AcquireData:** Software interface that allows partner application to send data acquired to Data4Help.

1.5.1 **Enroll user:** Software interface that allows partner application's users to become registered users into TrackMe's system.

1.6 **Track4RunManager:**

1.6.1 **EditRaceSettings:** Software interface that allows Track4Run application to create or modify races information.

2.1 **UserDBMS:**

2.1.1 **StoreUserData:** Software interface that allows to store users' data into the database.

2.1.2 **ProvideStoredData:** Software interface that allows to retrieve users' data already collected in the database.

2.2 **Track4RunDBMS:**

2.2.1 **StoreRaceData:** Software interface that allows to to store races' data with queries.

2.2.2 **ProvideStoredData:** Software interface that allows to retrieve users' data already collected in the database.

3.3 **UserLogger:**

3.3.1 **Enrollment:** Software interface that permits to inform Users' Manager that a new user is successfully enrolled and with which credentials.

4.1 **HealthAnalyser:**

4.1.1 **CallAmbulance:** Software interface that allows AutomatedSOS to call, through the internet or in the worst case through telephone call, First Aid whenever is necessary in order to provide help to user. Moreover it provides to FirstAid a special report on which parameters are critical.

5.1 **RaceDisplayer:**

5.1.1 **SelectRace:** Human interface that allows user to specify the race to be displayed.

5.1.1 **ShowLiveRace:** Human interface that allows to provide to the end user a map in which all the athletes involved in the selected race are displayed.

5.2 **RaceDisplayer:**

5.2.1 **ManageRace:** Human interface that allows user to create and manage a run providing all the useful information.

2.6 Selected Architectural Styles and Patterns

This system is designed to be a Client-Server application with three tier that well separate the different components as shown in Overview description in order to implement MVC software design pattern.

- **Architectural Design:**

  - Client-Server architectural pattern is chosen because it is the most used and easily implemented communication pattern in fact, in our application, is the best one to supply on third parties information requests and to acquire data from users via internet. Regarding the acquisition of users' data, is the smartphone that decides, when data are successfully collected from the device, to call Data4Help server in order to provide and store data, furthermore even third parties call Data4Help server first and then the exchange of the information can be proceed; these two operations are the main core of the service then is logic that client-server architecture will be the right choice.
  - Three tier architecture should be implement to assure reuse and maintainability of the system, in fact this division can permit to change some parts of the system without reconfigure the entire solution.

* Presentation Tier, as described above, is in charge of display on final users screen a human interface that allows the interaction with our system. This tier is present only on users' device: regarding third parties the software that is in charge to display human interface will be his own browser so the exchanged information will be an HTML page on Http connection; instead, regarding users, the software will be the application that runs on the device that works on HTML pages as well; so for both the type of users the interface to the Application Tier is the WebServer that allocate the right page to the right target. In order to supply on this aspect the partner application (AutomatedSOS and Track4Run as well) will be developed in HTML (with the support of JS and CSS) that are web languages, this aspect make also the application multi platform so they can be launched on either iOS or Android devices. Http communication protocol will be used for both connection.
* Application Tier, as described above, is in charge to acquire all the data incoming from users, to store them on Data Tier and to handle the request on viewing data. This Tier is present on the largest part on Application server but even AutomatedSOS has inside his software an application service. All the component inside this tier will be developed in Java, in particular the Web Server runs tomcat Java servlet that generates the html pages for Presentation Tier, acquire data that users submit and communicate through RMI with the business application that runs on the Application Server. The business application is the main core of Data4Help service, and is composed by all the other components inside ApplicationServer (as Request Manager,Data Collector,Statistic Generator..); is required that it can interface with Web server, in order to acquire the external requests, and then using the components described above supply the information required; furthermore is in charge to manage all the information of users, as login or registration,or manages Track4Run races. Another very important aspect is that this application has to retrieve data from users implementing a daemon on a specific port and listening for a TCP/IP connection incoming from partner application in order to acquire users' data. As mentioned before even AutomatedSOS implements a small part of application tier inside his software that is in charge to continuously monitor health status acquired, compare it with old data and call an ambulance via web if there is a necessity filling out a form which are indicated all the critical parameters detected.
* Data Tier, as described above, is in charge to store and provide informations. This tier is present only on Database Server that is an Oracle machine using SQL schemas. The database is divided in two parts, one regarding Data4Help service that stores and manages users' data: historical users' location and health parameters; the other one regarding Track4Run that stores races' information. This Tier offers an SQL interface that can be exploit by Business application using JDBC libraries that supports SQL functions.

- **Software Pattern:**

  - The MVC (Model-View-Controller) is the obvious software pattern that can be applied on three tier architecture because can split the entire software solution in the three main region: the Model implements all that is described in data tier, the controller all that is described on Presentation tier and View all that is described in Presentation tier. All these 3 groups communicates each others through the so called Adapter Pattern.
  - The Adapter design pattern is developed to create an interface that permits to the MVC groups to easily communicate each others using different languages or different operating system that is mandatory in our system. This pattern is the representation of what is described on Component Interface section.

- Strategy design pattern can be so much useful in the implementation of RequestManager component because it can dynamically change how to handle request queue in order to supply efficiently on heavy request load.
- Proxy design pattern is an obvious consequence of Web Server, because this component stay in the middle between the two tier and creates a sort mask that obscure what is behind it and simplify a lot the Presentation Tier software that has only to display what Web server indicates

2.7 Other Design Decisions

3 User Interface Design

in this Section the user interface design, already presented in *Section 3.1.1 User Interfaces* of *Requirements and Analysis Specification Document* with several mockups, is explained in more detail. A special attention is focused on the interaction between the user and the systems, and how the mockups are correlated each other.

- Data4Help

- AutomatedSOS

TrackMe offers to AutomatedSOS users an App for smartwatches, with which the users can see their location and health status information.

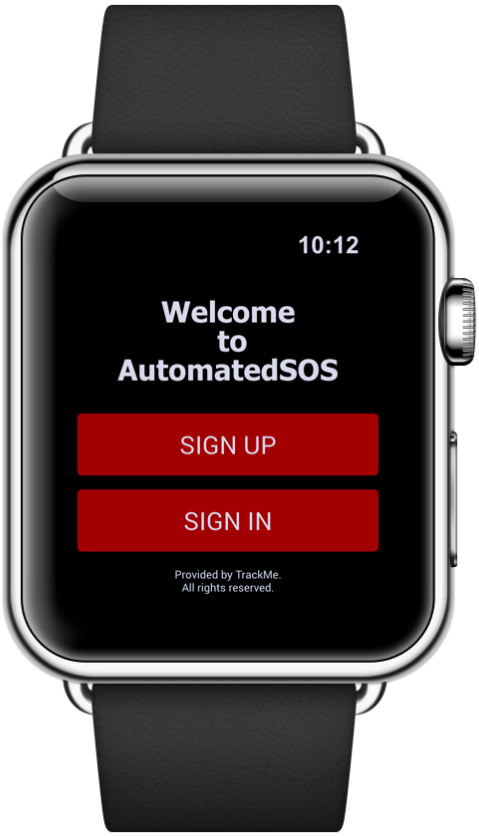


Figure 2: Welcome page.

Figure 3: Registration form.

In the first access to AutomatedSOS App the user is asked to sign up or to sign in (Figure 2). Based on the fact that the user already has a TrackMe account or not will choose the right option. In future accesses to the App the user do not need to select each time one of these two possible options because the App automatically remembers the account which is logged in. In case of sign in (Figure 3), the user has to fill all the mandatory fields in order to complete the registration form. Scrolling down the screen other fields will appear. Once every field is correctly filled, it would be possible to select the **NEXT PAGE** button.

Figure 4: Privacy Policy Conditions.

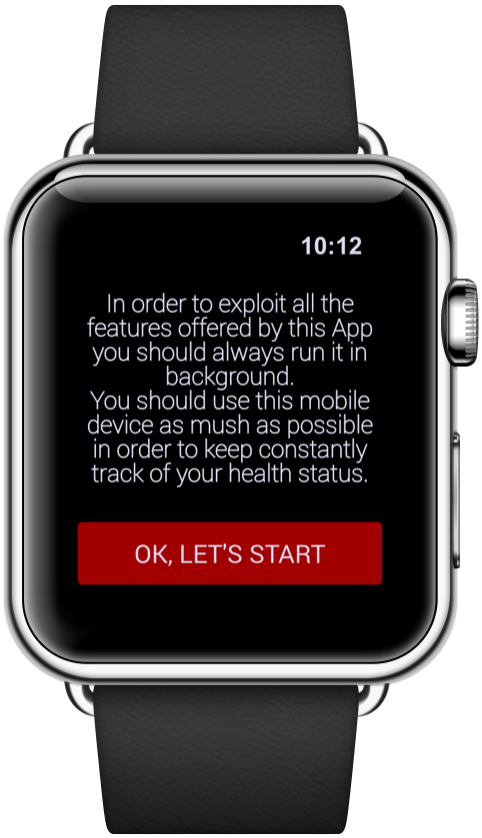


Figure 5: Usage Conditions.

During the first access to the App, the next step after sign up/sign in is to agree to the Privacy Policy Conditions (Figure 4). In order to use this App the user has to agree to the treatment of Location and Health Status data by Data4Help. In addition, these data could be used by third parties for the group monitoring request. Without agreeing the Privacy Policy Conditions the user cannot go to the next page, therefore he/she will not be able to use this App. The last step before starting use the App is taking note of the importance of wearing the Smartwatch as much as possible and to let the App runs in background (Figure 5).

Figure 6: Main menu.

Figure 7: Warning message.

The main menu of the App, which is immediately accessible by selecting the AutomatedSOS App on the Smartwatch home, is composed by three parts (Figure 6). Selecting **Monitor Health** the user can see live health informations, like the current Heart Rate, Blood Pressure and so on. Choosing the **Acquired Info** button the user can see all the historical data stored since the first access to the App. Finally, **Preferences** option allow to the user to set own threshold parameters according to the particular illnesses he/she is affected, own age and so on. As soon as an ambulance request is done due to the user's critical health status a warning message (Figure 7) appears on the screen of the Smartwatch notified by an alert sound.

- Track4Run

Track4Run users can use an App for smartphone and another one for smartwatches. The first one could be used by everyone, while the second one is made only for the athletes.

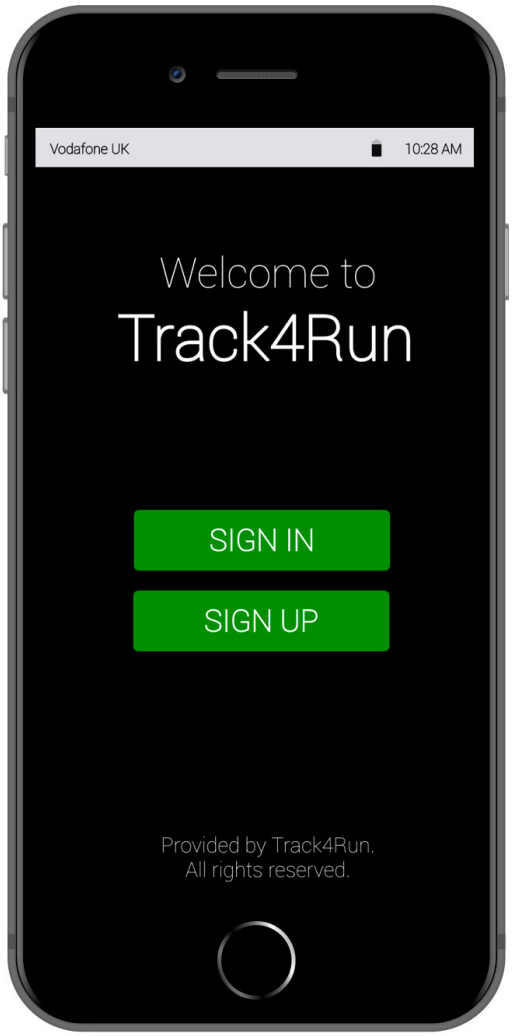


Figure 8: Welcome page.

Figure 9: Registration form.

In the first access to Track4Run App the user is asked to sign up or to sign in (Figure 2). Based on the fact that the user already has a TrackMe account or not will choose the right option. In future accesses to the App the user do not need to select each time one of these two possible options because the App automatically remembers the account which is logged in. In case of sign in (Figure 3), the user has to fill all the mandatory fields in order to complete the registration form. Once every field is correctly filled, it would be possible to complete the registration selecting the **Register Now** button.

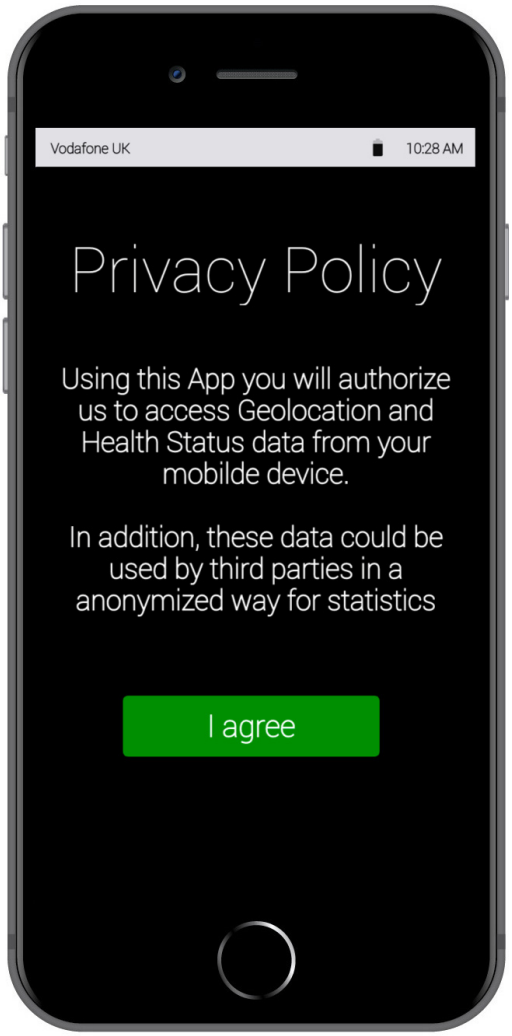


Figure 10: Privacy Policy Conditions pt.1

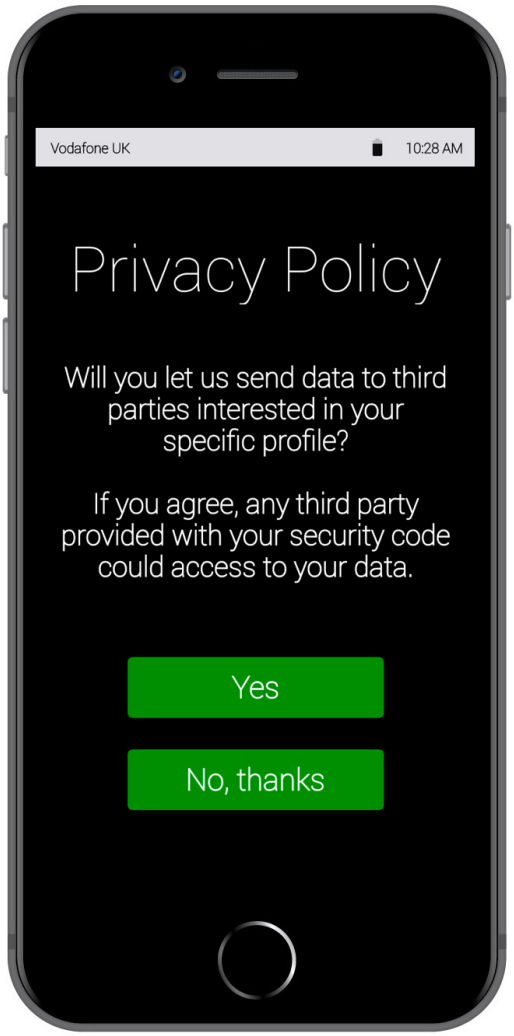


Figure 11: Privacy Policy Conditions pt.2

During the first access to the App, the next step after sign up/sign in is to agree to the Privacy Policy Conditions (Figure 4). In order to use this App the user has to agree to the treatment of Location and Health Status data by Data4Help. In addition, these data could be used by third parties for the group monitoring request. Without agreeing the Privacy Policy Conditions the user can not go to the next page, therefore he/she will not be able to use this App. All the three first steps presented so far are substantially the same of AutomatedSOS. The next step regards the second part of the Privacy Policy Conditions (which was not presented for the previous system) about individual monitoring request (Figure 9). In this case it is not strictly necessary to accept it, the user simply can select the **No, thanks** option.

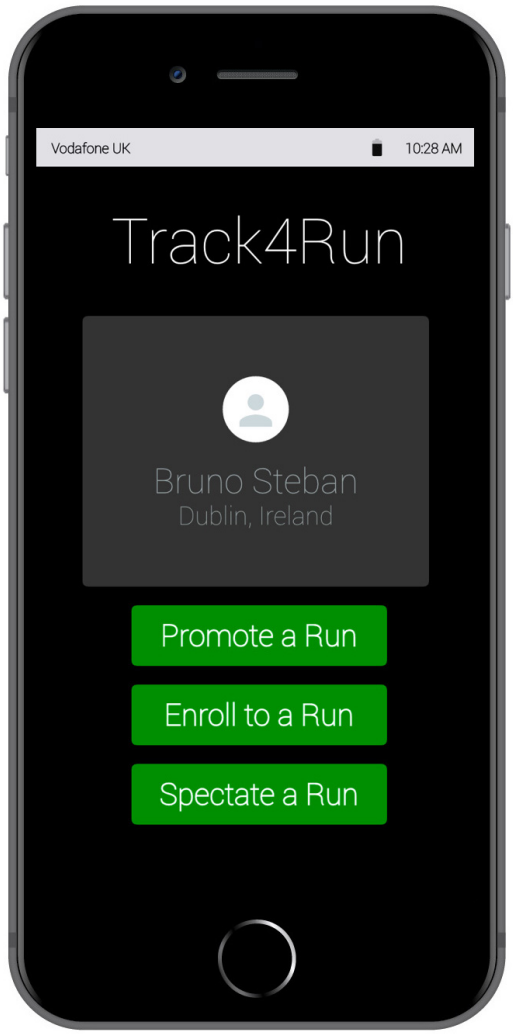


Figure 12: Main menu.

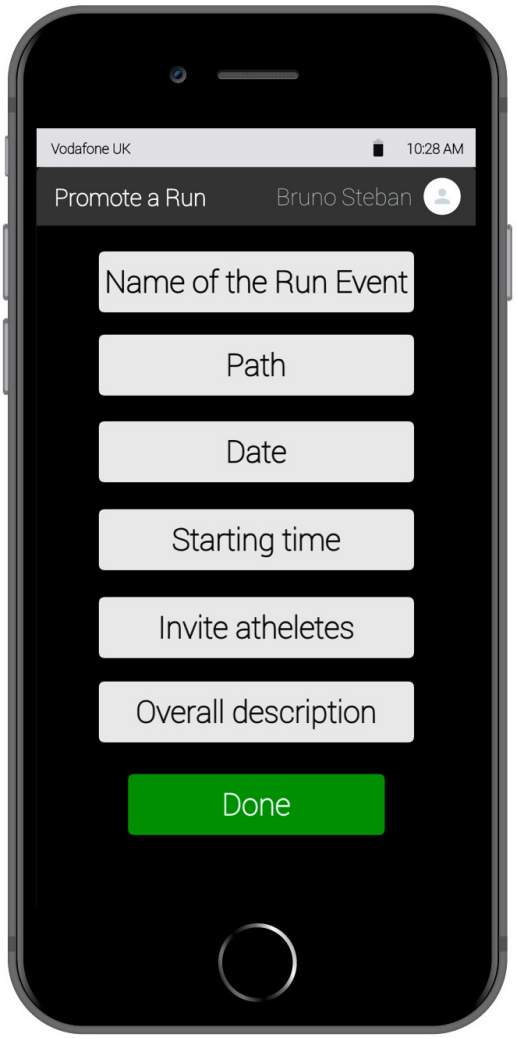


Figure 13: Pomote a run view.

In Track4Run Home three options are possible. Selecting the first one, **Promote a Run**, a user can create a run event and promote it inviting athletes. The second option, which is **Enroll to a Run** allows the user enrol to a run. Finally, choosing **Spectate a Run** a spectator can see on a map the position of all runners during the run. Now, we analyze these three possibilities in detail. Choosing the first one the App leads the user to a new page in order to manage the event (Figure X). Here, it is possible to set all the necessary features of the run, like to define the path on a map, set the date, the starting time and so on. It is also possible to invite athletes to the run in order to promote the event.

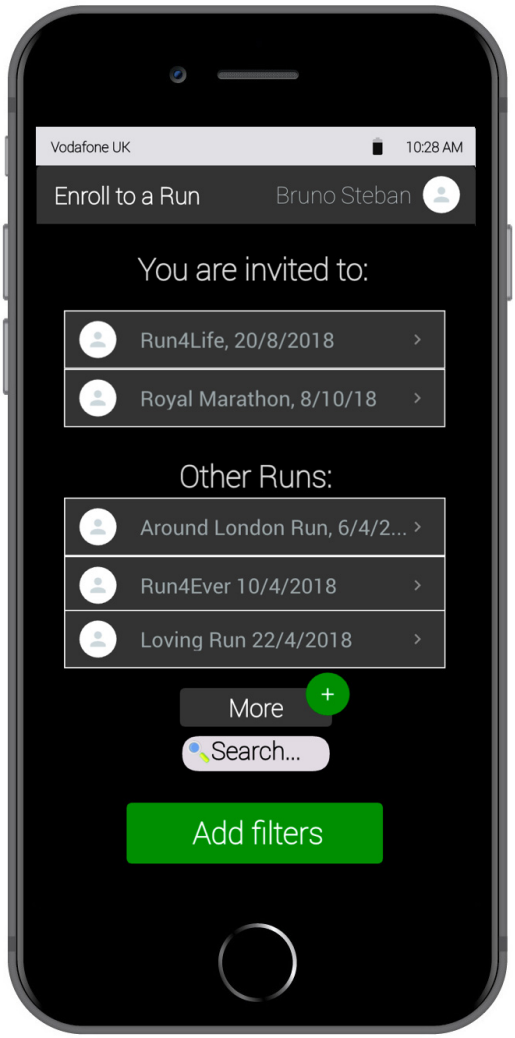


Figure 14: Enroll to a run.

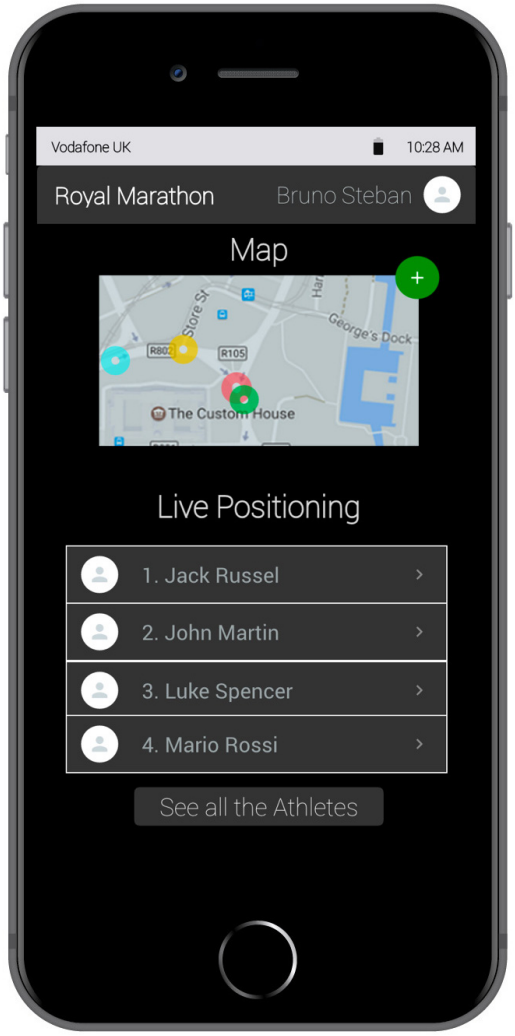


Figure 15: Spectate a run view

.

In the section Enroll to a Run (Figure X) a user can see a list of all the runs in which he/she has been invited. The user can select each of these runs to see furthermore details about it and at the end to enroll to it. A list of all the runs that will be soon held is showed immediately below. Since the large number of all the future runs, only a little portion of them is showed, selecting the **More +** button the user can see another set of run events. In addition, through **Add filters** it is possible to limit the list of the all runs to a spefic date, location and similar options. To easily access to a specific run, the **Search** button allows the user to immediately find the run he/she was looking for. In the last section, Spectate a Run (Figure X), it is possible to spectate to a specific run in a interactive way. A map shows all the runners enrolled in it, each one represented by a different coloured circle. Selecting the **+** button is possible to zoom in the map that will appear on full-screen mode. Below the map the live positioning show the first four athletes, anyway selecting **See all the Athletes** option it is possible to see the live positions of all the participants.

4 Requirements Traceability

5 Implementation, Integration and Test Plan

6 Effort Spent

In this section contains information about how much hours each group member spent in working at this document.

6.0.1 Luca Alessandrelli

Date	Task	Hours
23/11/18	Overview	1
19/10/18	Deployment View	3
20/10/18	Runtime View	6
24/10/18		
24/10/18	Domain Assumptions	1
30/10/18	Text Assumptions	0.5
30/10/17	Domain Assumptions	0.5
30/10/18	State Chart	1.5
4/11/18	Goals	2
4/11/18	Text Assumptions	2
4/11/18	Domain Assumptions	2
5/11/18	State Chart	1.5
5/11/18	Use Case	4
6/11/18	Use Case	3
6/11/18	Use Case Diagram	3
7/11/18	Use Case	1.5
7/11/18	Use Case Diagram	1
8/11/18	Use Case	2
8/11/18	Use Case Diagrams	0.5
8/11/18	Scenarios	0.5
9/11/18	Scenarios	1
10/11/18	Document revision	3.5
11/11/18	Alloy	8
Overview		1
Deployment View		3
Runtime View		6
Use Case		10.5
Use Case Diagram		4.5
Alloy		8
Document Revision		3.5
Total		47

6.0.2 Andrea Caraffa

Date	Task	Hours
18/10/18	Goals	2
19/10/18	Domain Assumptions	3
20/10/18	Text Assumptions	3
21/10/18	Introduction	2
27/10/18	Goals	2
30/10/18	Product Functions	3
1/11/17	Mockups	3
3/11/18	Mockups	3
4/11/18	Goals	2
4/11/18	Mockups	2
5/11/18	External requirements	2
5/11/18	Alloy	2
6/11/18	External requirements	3
9/11/18	Revisioning	2
9/11/18	Alloy	3
10/11/18	Revisioning	3
11/11/18	Revisioning	3
11/11/18	Alloy	3
Text Assumptions		3
Goals		6
Domain Assumptions		3
Introduction		2
Product functions		3
External requirements		5
Mockups		8
Alloy		8
Document Revisiong		8
Total		46

6.0.3 Andrea Bionda

/	Task	Hours
	Text Assumptions	3
	Goals and Introduction	6
	Domain Assumptions	3
	Functional requirements	13
	Class Diagram	5
	Sequence Diagram	5
	Performance requirements and Constraints	3
	Alloy	7
	Total	45

7 Reference Documents

- Specification Document "Mandatory Project Assignment AY 2018-2019".
- Slides "Structure of RASD".
- Slides "Use of Alloy in RE".
- Use Case Diagrams created with https://www.lucidchart.com
- Mockups created with https://www.fluidui.com