INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Tecnologia e Gestão

SMACovid-19 — Autonomous Monitoring System for Covid-19

Rui Victor Pires Fernandes – 49212

Dissertation submitted to Higher School of Techonology and Management in order to obtain the Master Degree in Industrial Engineering.

Dissertation Advisers:
Professor PhD José Barbosa

Bragança
2020-2021

Acknowledgement

I would like to thank my supervisor, Professor José Barbosa, for his guidance and support throughout the execution of the work that culminated in this Thesis. I would also like to thank my colleagues of the TECH group for the solidarity and companionship shown throughout the execution of this project.

I extend my thanks to all the people who, in some way, were involved in the development of the SMACOVID-19 project.

Resumo

Atualmente, existe uma pandemia global, COVID-19, que pode provocar consequências devastadoras para a saúde humana. Além disso, pessoas idosas têm uma probabilidade maior de sofrerem os efeitos mais severos da doença. *wearable devices* podem monitorizar parâmetros biológicos/físicos, e uma análise de previsão aos dados resultantes deve permitir a identificação mais rápida possível de pessoas que possam estar infetadas com o vírus.

Para criar um sistema com estas capacidades, desenvolveram-se os módulos *health data provider* e *data analysis* para recolher os dados e fazer as previsões, com base nesses dados, respetivamente. A plataforma FIWARE foi usada para implementar o *backend* do sistema, através de módulos *open source ready-to-use* para gerir os dados no sistema.

Os médicos do Hospital da Terra Quente definiram os parâmetros biológicos/físicos de interesse e, considerando os diferentes tipos de variáveis, implementaram-se três tipos distintos de previsões: última observação, estimador linear de tendência e método aditivo Holt-Winters. As previsões obtidas são usadas para classificar o estado de uma pessoa, para um cenário de pior caso possível: provável que esteja infetado com o vírus ou provável que não esteja infetado com o vírus. As regras de classificação foram definidas pelos médicos do Hospital da Terra Quente.

Os resultados obtidos são promissores e trabalho futuro pode ser feito para melhorar o sistema na inclusão de mais fontes de dados, ajuste dos métodos de previsão e na introdução de novas metodologias de classificação final.

Keywords: COVID-19, FIWARE, Dispositivos *Wearable*, Previsão.

Abstract

Nowadays, there exists a global pandemic, COVID-19, that may provoke devastating consequences to human health. Moreover, elder people have a higher probability to suffer the most harsh effects of the disease. By monitoring biological/physical parameters with the aid of wearable devices and applying forecast methods to the collected data, it should be feasible to identify, as soon as possible, people that may be infected with the virus.

To create a system with such capabilities, a health data provider and a data analysis modules were implemented, to fetch the biological/physical data and to generate forecasts based on that data, respectively. The backend of the system was implemented through the FIWARE platform, using open source ready-to-use modules of this platform to manage the data in the system.

The biological/physical parameters of interest were defined by the medical personnel of Hospital da Terra Quente and, considering the different type of variables in analysis, three types of forecasts were implemented: last observation, linear trend estimator and additive Holt-Winters method. The obtained forecasts are used, combined with a worst case scenario approach to classify the state of a person: likely to have been infected by the virus or not likely to have been infected by the virus. The classification rules were devised by the Hospital da Terra Quente medical staff.

The obtained results are promising and future work can be done to improve the system in terms of providing extra data sources, fine tuning of the forecast methods and by introducing new final classification methodologies.

Keywords: COVID-19, FIWARE, Wearable Devices, Forecasting.

Contents

List of Tables

List of Figures

Acronyms

AAL Ambient Assistant Living. 5, 8, 11, 12

API Application Programming Interface. 21–23, 25–27, 35, 39, 41–44, 54

ARIMA AutoRegressive Integrated Moving Average. 50

CB Context Broker. 18, 23, 25–33, 36, 37, 41, 43, 45–50

COVID-19 COronaVIrus Disease of 2019. 1–4, 51, 56, 59, 60

FET Field Effect Transistor. 14

FHIR Fast Healthcare Interoperability Resources. 40, 41

GPS Global Positioning System. 8, 13, 15, 16

HAR Human Activity Recognition. 11

HL7 Health Level Seven International. 40

HTQ Hospital da Terra Quente. 2, 3, 24, 41, 51, 56

HTTP Hypertext Transfer Protocol. 20

INE Instituto Nacional de Estatística. 5

IoT Internet of Things. 19, 20

JSON JavaScript Object Notation. 19, 35, 45

LED Light Emmiting Diode. 17

MAE Mean Absolute Error. 50, 55–57, 59

NTC Negative Temperature Coefficients. 17

PTC Positive Temperature Coefficients. 17

RMSE Root Mean Square Error. 55–57

SMS Short Message Service. 20

SpO2 Saturation of Peripheral Oxygen. 9

URN Uniform Resource Name. 39, 46, 48

UUID Universally Unique IDentifier. 39

ZigBee Zonal Intercommunication Global-standard. 8

Chapter 1

Introduction

Nowadays, there's a global pandemic in effect, COronaVIrus Disease of 2019 (COVID-19), with the most diverse consequences, changing the way people live their lives and, regrettably, taking people's lives in great numbers. Moreover, this is a disease that has strong correlation with people's age, regarding the influence of the effects it may cause on people's health. This implies the existence of groups, within the general population, that are more easily affected by COVID-19 and/or have a greater probability of suffering the more devastating effects of the virus. Given the fact that age is a critical factor, the elderly people are one of those groups. Just like with any other disease, an early diagnosis can be extremely important to mitigate the virus effects, and live monitoring of these people can aid to achieve that objective, hence, wearable devices present themselves as a tool to create a system to monitor patients that may, or may not, be infected with COVID-19.

Wearable devices are far from the traditional watches people once wore on their wrists. They still inform about the time but they are also equipped with several different sensing capabilities that allow them to provide information about other type of variables, such as, body temperature, heart rate, oxygen saturation, calories, steps, flights, etc. These new capabilities allow the development of applications in different areas, e.g., fitness, wellness, healthcare.

The project SMACOVID-19 — Autonomous Monitoring System for COVID-19 (70078

– SMACOVID-19) [1] involves three institutions and, as the name indicates, has the purpose of developing a COVID-19 monitoring system. The institutions are:

- MORE – Laboratório Colaborativo Montanhas de Investigação – Associação [2]. Project leader.
- Riskivector, Unipessoal, Lta [3].
- Hospital da Terra Quente (HTQ) [4].

The idea behind the project is to develop an innovative solution that allows users to monitor their health status and alerts health professionals to potential deviations from the normal pattern of each user. To that end, data is collected, from wearable devices and from manual input, to be processed by predictive algorithms, in order to forecast values for the short-term future, aiding the prediction of the potential worsening of the clinical situation of the patients.

1.1 Objectives

The SMACOVID-19 project comprises the following objectives:

- To develop an innovative solution that helps to diagnose/identify possible infected by COVID-19 at an early stage by monitoring biological/physical parameters and by conducting medical surveys.
- To process the collected data through predictive algorithms in order to analyze the health status of each person.
- To make the data available to different stakeholders on a platform adjusted to the profile of each user.
- To allow health professionals to monitor their patients closely and continuously and quickly identify possible epidemiological situations, but also to allow wearable wearers to be more aware of their health status.

All three stakeholders contribute to achieve these objectives:

- MORE is responsible for the data gathering, storage and analysis.
- Riskivector is responsible for platform creation/management and the frontend.
- HTQ is responsible for the biological/physical parameters definition and classification. It also states the metric to use to define if a patient is likely to have been infected with the COVID-19 virus.

Figure 1.1 shows the activities of the SMACOVID-19 project that were defined to achieve the proposed objectives. The ones marked with the green color represent activities where there was work done in the context of this thesis: activity 2 was executed in collaboration with Riskivector; the work implemented in activity 3 resulted in the Health Data Provider module; the work performed in activity 4 resulted in the Data Analysis module.

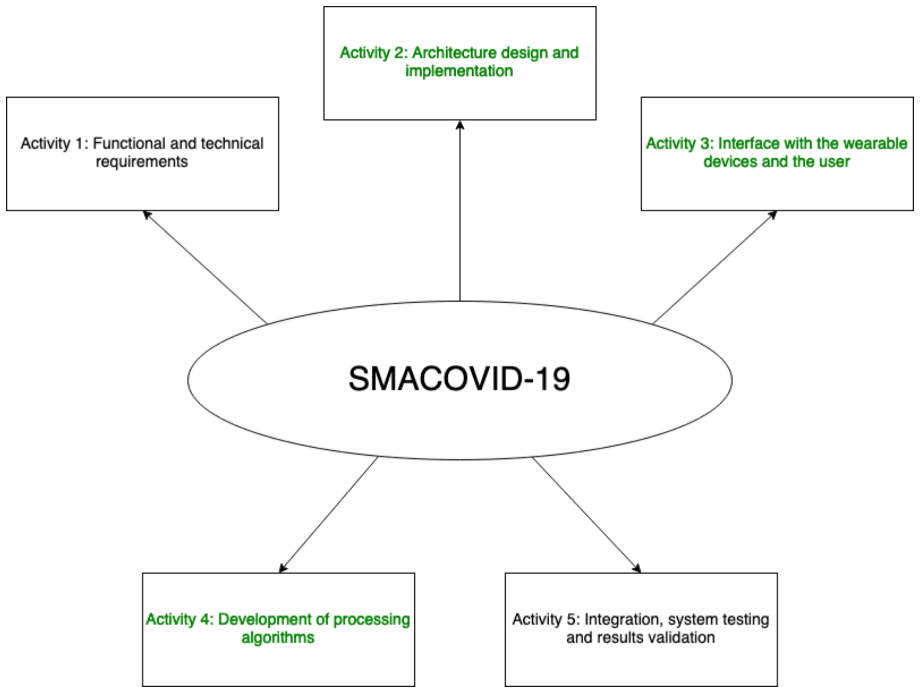

Activity 2: Architecture design and implementation
Activity 1: Functional and technical requirements
Activity 3: Interface with the wearable devices and the user
SMACOVID-19
Activity 4: Development of processing algorithms
Activity 5: Integration, system testing and results validation

Figure 1.1: SMACOVID-19 project activities.

In this scenario, this Thesis objectives are:

- to create the backend part of the system, based on the FIWARE platform, to collect contextual data about variables relevant to COVID-19 infection detection;
- to store efficiently this contextual data;
- to develop/implement predictive algorithms that, based on the available contextual data, generate short-term forecasts, for the variables defined by the medical personnel, to identify as quickly as possible eventual cases of possible COVID-19 infection.

1.2 Thesis Structure

The remainder of this thesis comprises six extra chapters.

Chapter 2 presents the state-of-the-art.

Chapter 3 discusses elderly monitoring technologies, introducing the wearable devices used in such type of monitoring. It describes the sensors that are usually incorporated in the wearable devices and provides a brief explanation of its working principle. This chapter finishes by providing a description of the FIWARE platform.

The final architecture of the devised system is introduced in Chapter 4, discussing how the modules work collaboratively to comply with the proposed objectives, finishing with the presentation of the individual modules that are part of the system, presenting them in greater detail.

Insight about the defined data model used in the system is provided in Chapter 5. Afterwards, this chapter continues with details on how the data is acquired and stored, finishing with the data analysis algorithms and forecast creation.

Chapter 6 presents and discusses the implementation of the system and some of the obtained results.

In Chapter 7, some conclusions are taken and future work is suggested.

Chapter 2

State of the Art

Elderly people have seen their numbers grow and are already an important slice of the general population. For instance, in Portugal, a study from Instituto Nacional de Estatística (INE) and Pordata [5], related to the year 2020, indicates that this age group (over 65 years) already corresponds to 22.3% of the general population. Furthermore, data from recent years shows that this is the only age group presently growing in numbers. Both young (0 to 14 years) and active adult (15 to 64 years) groups are decreasing, as shown in the INE report about demographic statistics of Portugal [6]. Beyond their numbers, this is an age group where people have to deal with some age associated difficulties. For instance, physical conditions deteriorate with age, consequently, they are more accident-prone and there are a lot of elderly living alone, in a scenario where, if an accident happens, they are isolated and without anyone near to help them. In this type of scenario, elderly monitoring can be implemented for different purposes, however, the idea behind all approaches is to improve their quality of life. Having this in mind, Ambient Assistant Living (AAL) is a field of research that includes systems and services designed to:

- monitor elderly activities (providing help in case of emergency),
- compensate/alleviate the age related physical/psychological limitations,
- provide autonomy and safety to the elderly people,

- improve their life quality, allowing them to continue their life in their own environment (with a constant remote supervision),

- monitor particular parameters in case of specific diseases.

This field of research includes areas such as video surveillance, fall detection, emergency response and sensor-based (wireless and/or environment sensors) systems.

Considering the accident-prone scenario, in which accidents may result in dire consequences, fall detection is a field of investigation where several works have been implemented. In [7] the authors use only acceleration information, from two accelerometers, one placed on the chest and other placed on the wrist of the person, to infer if a fall happened. [8] also presents a system that detects falls using the information from an accelerometer placed inside a custom wearable device the person wears on his wrist, coupled with ground truth data, for fall conditions, recorded with that wearable. They are able to identify forward, backward and lateral falls. [9] uses infrared technology to detect falls from the bed by applying a set of infrared sensors on the bed itself. The authors of [10] developed a wrist worn wearable device containing an accelerometer and a gyroscope and, based on their readings, determine the acceleration and angular velocity, that are compared with threshold levels to identify a fall. A similar approach was used in [11, 12]. In [13], the authors take another approach and use a Pi Camera recording the environment the elder person is in combined with image processing: Motion History Image and C-Motion techniques are used to identify a person's fall. Other work that also uses image processing is [14], that proposes a fall detection algorithm based on depth images of the recorded fall event.

The presented works so far use primordially custom wearable devices worn by the elderly and projected/implemented according with the needs required by the type of monitoring being investigated. This customization may be taken a step further and include sensors able to read physiological/biological parameters of the person. Several different parameters can be read with the purpose of detecting abnormal values, for these variables, that may suggest a distressful clinical situation for the person. In this scenario, the systems developed usually send some sort of alert (to family members, to medical personnel), containing the position of the person, so that aid can be presented to the elderly person as soon as possible. This allows other type of monitorization, namely, of diseases that may have sudden consequences, e.g., heart attacks, epilepsy, Parkinson, etc. In [7], data from heart rate and body temperature is collected and, afterwards, is put available to the healthcare professionals through a website. In [15], a wearable shirt is presented, capable of reading pulse, frequency of breathing, underclothing temperature, positioning inside and outside the house. This allows a continued overview of the health status of the monitored person. A healthcare data collector framework is developed in [16] to provide healthcare providers information about individuals' health conditions and their living environment. To that end, they developed a wearable that contains an accelerometer, a skin temperature sensor, and two reflective photoplethysmography sensors to measure heartbeat and oxygen saturation level in the blood. In [17], wearable data was used to do physical activity recognition, for elderly with Parkinson's, based on machine learning algorithms. They used a wearable with a triaxial accelerometer and a gyroscope for activity recognition — they employed a comprehensive study of several features and classifiers for recognising different activities. In [18], wearable devices were used to read the temperature, heart rate and blood oxygen saturation level sending notifications when a deviation in values is detected. [19] presents a system that integrated wearable devices capable of sensing temperature, heart pulse and galvanic skin response with Raspberry Pi and Arduino devices to develop an IoT solution for elderly monitoring. These two works have the disadvantage that they require multiple wearable devices to be able to sense the variables they intend to know about. In [20], the authors use the Apple Watch series 5 as the wearable, using only a single wearable device, and read the heart rate, blood oxygen saturation, calories burnt, number of hours of sleep vs the number of hours in bed and the patient's activity type. All this information is displayed in real-time to the caregiver. They set rules to trigger alerts based on the values these parameters have.

Other works present an interaction between wearable devices and a home/environment framework to develop a system to help the elderly. Usually, these works tend to detect falls and the person position, to detect activities and abnormal behaviour inside the house. An example of such type of cooperation is the fall detection work performed in [21], in which a system that couples an accelerometer, a cardiotachometer and ambient environment sensors information (temperature, humidity) is presented, where they analyse impact magnitude, trunk angle and after-event heart rate to identify if a fall happened. They also establish four different levels of emergency to define what type of alert should be sent and who should receive it (relatives, caregiver, medical emergency services personnel). Furthermore, a Zonal Intercommunication Global-standard (ZigBee) network is deployed at the person's home, to ensure the wearable device connectivity. Another study, [9], presents an indoor monitoring system that detects, based on infrared technology, bed-exiting and provides a washroom alarm subsystem, that is triggered when the person is inside the washroom for a certain period, surpassing a threshold of time set beforehand. Again, the connectivity of the system is ensured by a ZigBee network deployed at the house.

Another important aspect to consider is the fact that there are a lot of elderly people that live alone, thus, activities and behaviour monitoring can be used with wearable devices to aid these people. [22] presents an AAL system that uses wearable sensors data, cognitive, medication and motivation information to implement a cognitive assistant for a physically active lifestyle. Wearable sensors are used to measure vital signs: heart rate, body temperature, respiration rate, daily steps, posture, to assess the elder's health status. With the purpose of encouraging the elder, it is possible to adapt the suggested actions the elder should perform, from an Exercise Motivations Inventory questionnaire coupled with the elder's biological sensed data. The caregiver can also change the suggestions. In [23], the authors present a system they named "The SafeNeighborhood Approach", to monitor outdoor activities performed by the elders, based on both a wearable device with sensing capabilities (Global Positioning System (GPS), accelerometer, temperature and light), as well as a trusted neighborhood network, to aid the elder in case an emergency is detected. It is capable of notifying five different possibilities of state, for the elder person:

(1) Ok, (2) Fallen — has not gotten back up in a given time period, (3) adverse weather — has been exposed to rough weather conditions more than a given time interval, (4) wandering — has been pursuing erratic paths and (5) Risk of getting lost — is getting away from an acceptable area. In [24], the authors propose a wearable device, with an accelerometer, a gyroscope and an inertia sensor, capable of recognizing six human daily activities (walking, walking upstairs, walking downstairs, sitting, standing, and lying) through a deep learning algorithm using convolutional neural networks.

In terms of healthcare, in addition to the areas already described, other different aspects of wearable usage have been the case study of researchers. For instance, in [25], the authors present a system for automated rehabilitation; in [26] big data analysis is used to define healthcare services; in [27] it is presented a framework to provide real-time epilepsy detection. The authors of [28] use the random forest supervised learning method and wearable sensors more easy to use/find like temperature, electrodermal activity, and acceleration to predict heart rate and blood oxygen level of patients. In [29], wearable technology is used to track and classify real-time hand strike motion in a combat sport.

In terms of wearable devices, a characteristic that is present throughout the revised works is that custom wearable devices had to be created, to read certain characteristics of the human body. Only one work,[20], used a ready-to-use commercial wearable. Moreover, present commercially available wearable devices incorporate numerous sensing capabilities able to sense the necessary parameters for healthcare solutions to be created. Thus, given the fact that for this type of applications the wearable device should be used continuously, it makes sense to use one of the available solutions that people already associate with that type of usage. In this work, the Fitbit Sense was used, since it incorporates skin temperature, Saturation of Peripheral Oxygen (SpO2) levels and heart rate sensing capabilities.

Chapter 3

Technologies for Elderly Monitoring

This chapter discusses some of the technologies available to perform elderly monitoring. Section 3.1 briefly presents the idea behind retirement homes monitoring systems, Section 3.2 discusses how wearable devices can be used for elderly monitoring, presenting some of the available sensors and its working principle and Section 3.3 presents the FIWARE platform on which the development of this work was based.

3.1 Retirement Home Monitoring

Technology has been aiding caregivers at retirement homes in elderly monitoring, for AAL purposes. AAL technologies may contribute to the preservation of the quality of life of aging population in an affordable way, thus, decreasing the need of social and health-care services [30]. AAL includes systems and services designed to: compensate the age related physical and psychological limitations, guarantee autonomy and safety to the elderly people; constantly monitoring their activities and providing help in case of emergency; improve their life quality, allowing them to continue their life in their own environment with a constant remote supervision also in case of specific diseases [31].

In retirement homes, one important field is the Human Activity Recognition (HAR) field, that plays an important role: an active lifestyle is the basis for a healthy life and HAR can assess users' lifestyle by monitoring the amount of daily activity. The data gathered from this monitoring can be used to build a behavioral model that can be useful for the early detection of anomalies relevant to well-being of the elderly people.

This type of systems, usually, includes some of the characteristics already mentioned for AAL systems mentioned in Chapter 2, such as, positioning monitoring, inside the home, for fall detection, wandering behaviour, number of visits to the bathroom, etc, and personal monitoring, for fall detection, for getting biophysical readings from the person. These two sensing capabilities can be combined: the house (motion sensors, cameras, ...) and the person (wearable device with sensing capabilities), to achieve the final objective of daily activity recognition.

Usually these systems notify someone (caregiver, relative, medical personnel) when an abnormal situation is detected, e.g., fall detected, too much time in a given room (bedroom, for instance), position at an unexpected location.

These systems can greatly improve the quality of life of the elders on the several different aspects AAL systems work on and, in respect of retirement homes, can free up human caregivers so they can be as efficient as potentially possible. This allows less caregivers managing more elders. However, there are some aspects that still have to be improved: these systems are only effective when the elders agree with its use and, sometimes, that is not the case, namely, when cameras are used to monitor the elders location [32]. Furthermore, issues can be raised about the systems accuracy, its privacy and the reduction of human interaction and hands-on care, worsening social isolation for older people [32]. The worst disadvantage, however, is the fact that this type of systems can, in fact, change the behaviour of the elders instead of only monitor it. To reduce alerts generation, there have been cases reported in which the elders purposely changed their routines, to be certain no alert would be sent to their relatives [32]. For example, people stopped sleeping in recliners afraid that would trigger an inactivity alert, or people were quicker in the bathroom to avoid an alert to be sent because they were there too much time.

3.2 Wearable Devices

The vast majority of elderly monitoring uses some kind of wearable device to collect data about the vital signals of the person. A wearable device can be described as a technological device that is worn ubiquitously and continuously in close contact by a person, usually having direct contact with the skin, to allow the necessary data gathering to be performed.

This type of devices, with this type of data gathering capabilities, represents one of the most important sources of data generation today, since they can record different types of data continuously, in the most diverse contexts. This allows the development of applications related to the health or fitness areas.

To be able to read data, these devices incorporate hardware with the most diverse sensing capabilities. Some of the most used sensors are:

- accelerometers
- gyroscopes
- GPS sensors
- skin temperature sensors
- photoplethysmography sensors
- heart pulse sensors
- other types of sensors [33]

An accelerometer is a device that measures the vibration or acceleration of motion of an object. The force caused by the vibration or the acceleration impacts the piezoelectric material of the sensor resulting in a generation of an electric charge, proportional to the applied force. Since $\vec{F} = m\vec{a}$ and the charge is proportional to the force, considering the constant mass of the piezoelectric material, the charge is also proportional to the acceleration. Accelerometers are used on a wide array of devices, being one of the most used sensors in handheld devices: for instance, nowadays, almost all smartphones have an accelerometer. In an industry setting, accelerometers help engineers analyze a machine's stability and enable them to monitor forces/vibrations the machine is subjected to.

There are two types of piezoelectric accelerometers: high and low impedance. In the first type, the piezoelectric crystal produces an electrical charge which is connected directly to the measurement instruments. The charge output requires special accommodations and instrumentation most commonly found in research facilities. This type of accelerometer is used in applications with high temperature requirements, $T > 120°C$.

A low impedance accelerometer has a charge accelerometer as its front end but also has a tiny built-in microcircuit and Field Effect Transistor (FET) that converts that charge into a low impedance voltage that can easily interface with standard instrumentation. This is the type of accelerometer that is most commonly used in industry.

A gyroscope sensor measures the angular velocity or tilt or lateral orientation of an object. Measured in degrees per second, angular velocity is the change in the rotational angle of the object per unit of time. These type of sensors are usually combined with accelerometers, to achieve a more robust/accurate sensing capability.

Gyroscopes can have multiple axes and depending on the direction, there are three types of angular rate measurements:

- Yaw — the horizontal rotation on a flat surface when looking at the object from above;
- Pitch — the vertical rotation when looking at the object from the front;
- Roll — the horizontal rotation when looking at the object from the front.

In order to measure one of these angular rates, the rotation rate of the sensor is converted into an electrical signal.

In terms of size, from small to large, gyroscope sensors can be defined as *Vibration gyroscopes*, *Fluid gyroscopes*, *Fiber-optic gyroscopes* and *Ring laser gyroscopes*. *Vibration gyroscopes* are easy to use and have the smallest size, making this type the most popular one. This sensor consists of an internal vibrating element made up of crystal material in the shape of a double **T** symmetrical structure. This structure includes a stationary part in the center, with a *sensing arm* attached to it and a *drive arm* on both sides.

When an alternating vibration electrical field is applied to the drive arms, continuous lateral vibrations are produced and, since the drive arms are symmetrical, when one arm moves to left the other moves to the right, canceling out the leaking vibrations. This keeps the stationary part at the center of the structure and sensing arm remains static. This is used when an external rotational force is applied to the sensor, causing vertical vibrations on the drive arms, creating a rotational force that acts on the stationary part in the center. The rotation of the stationary part leads to vertical vibrations in the sensing arms, measured as a change in electrical charge. This change is used to measure the external rotational force applied to the sensor as angular rotation.

The accuracy of vibration gyroscopes depends upon the stationary element material used in the sensor and structural differences. Thus, manufacturers are using different materials and structures to try to increase the accuracy of the sensor.

Almost all smartphones nowadays include a vibration gyroscope, that is able to detect, e.g., the tilt and orientation of the mobile phone and can also be used to perform motion sensing for the mobile phone.

A GPS sensor is a receiver that uses communication with a satellite-based navigation system, with a network of 32 (24 active at a time) satellites that circle the Earth twice a day in a precise orbit, to provide position and velocity of the object the sensor is attached to. It also provides timing information.

Each satellite broadcasts different signals which can be tracked by a GPS receiver on Earth, which are then analyzed by the GPS receiver to determine its precise location. To achieve that, trilateration is used and, it needs information from at least 4 satellites.

- first — spherical surface,
- second — the intersection of both spherical surfaces results in a circumference,
- third — two points in the circumference are identified,

- fourth — the final point is selected.

The amount of time it takes to receive a transmitted signal from each satellite is used by the GPS receiver to calculate the corresponding distances. With, at least, four available distances, trilateration is used and the position is determined. If more than four satellites are available, the trilateration accuracy increases. GPS can be used for different purposes: determining a position (location), getting from one location to another (navigation), monitoring object or personal movement (tracking), creating maps of the world (mapping) and taking precise time measures (timing).

The three sensors discussed so far, accelerometer, gyroscope and GPS are the most common sensors found in handheld devices. However, with technology advancements, new wearable devices started to appear more focused on health and fitness purposes. These devices include other type of sensing capabilities, that are now going to be discussed.

The temperature sensors, used in wearable devices to read the body temperature can be of different types:

- photoconductivity
- infrared
- thermistor
- optical — photodetector

The photoconductivity of a material can be expressed as $\sigma_{ph} = AI^m$, where A is constant, I is the light intensity and m is the light intensity exponent, proven to be temperature dependant. This is used to read the skin's temperature by using a known light intensity.

Infrared temperature sensors have the advantage that they don't need to have direct contact with the surface they are measuring, thus, they are a highly efficient solution to temperature measurement in all kinds of environments. These sensors emit an infrared energy beam, focused by a lens, onto a surface and use the reflected beam energy to determine the temperature of the surface.

A thermistor is a semiconductor, that possesses a resistance that varies with its temperature. This relationship is highly dependent upon the materials from which it's composed. To calculate the current temperature, the present thermistor resistance is converted in temperature, according with the equations defined for the material the thermistor is made of. Thermistor's can be Negative Temperature Coefficients (NTC) or Positive Temperature Coefficients (PTC) depending if the variation of resistance is not alligned (or alligned) with the variation of the temperature, respectively. Thermistors are accurate to approximately $\pm 0.2°C$ within their specified temperature range.

Photodiodes or phototransistors can be used as a photodetector and they can sense light. Semiconductor-based photodetectors have a *pn* junction that converts photons into current, provoking the creation of a reserve current, proportional to the light intensity being detected, through the *pn* junction. This current can be used to determine the temperature.

Photoplethysmography is an optical measurement method that is usually used for heart rate monitoring purposes. It can also be used for blood oxygen saturation level determination. It is a non-invasive technology that uses an infrared light source and a photodetector at the surface of skin to measure the volumetric variations of blood circulation. The reflected light is proportional to blood volume variations. However, one of the drawbacks of this technique is its inaccuracy in tracking the photoplethysmography signals during daily routine activities and light physical exercises, since its very susceptible to motion artifacts. Usually two types of Light Emmiting Diode (LED) are used: (1) infrared — to measure the flow of blood; (2) green LED — to calculate the absorption of oxygen in oxyhemoglobin (oxygenated blood) and deoxyhemoglobin (blood without oxygen present), since it can penetrate more deeply into the human tissue, thus, providing more accurate measures.

A pulse wave is the change in the volume of a blood vessel that occurs when the heart pumps blood, and a detector that monitors this volume change is called a pulse sensor. Reflection-type pulse sensors emit infrared, red, or green light into the body and measure the amount of light the body reflects (photodiode or phototransistor). Oxygenated hemoglobin present in the blood of the arteries has the characteristic of absorbing incident light, hence, by sensing the blood flow rate (change in blood vessel volume), that changes following heart contractions over time, we are able to measure the pulse wave signal.

Red or infrared light sensors are more suitable for indoor environment since they are affected by the infrared rays contained in sunlight. Green light sensors, has a high absorption rate in hemoglobin and less susceptibility to sunlight, thus, is preferred for outdoors usage.

3.3 FIWARE

In their own words [34], "FIWARE brings a curated framework of open source software platform components which can be assembled together and with other third-party components to build platforms that support the development of Smart Solutions faster, easier and cheaper.". In essence, FIWARE, defines a set of standards for context data management and the FIWARE Context Broker (CB) is the key element of the platform, enabling the gathering and management of contextual data: the current context state can be read and the data can be updated. This element is also the only mandatory element for a solution to be able to be described as a "Powered by FIWARE" solution. Moreover, FIWARE puts available a set of complementary open source elements that can be used in conjunction with the CB. These elements cover different areas of functionalities, as presented in Figure 3.1, devised around the CB management block, reinforcing the importance of the CB for this kind of implementation.

The Core Context management block has available the following generic enablers to be used:

- STH Comet — has the capability of storing a short-term history of context data (typically months) on MongoDB;
- Cygnus — allows context history management created as a data stream that can be injected into multiple data sinks, including some popular databases like PostgreSQL,

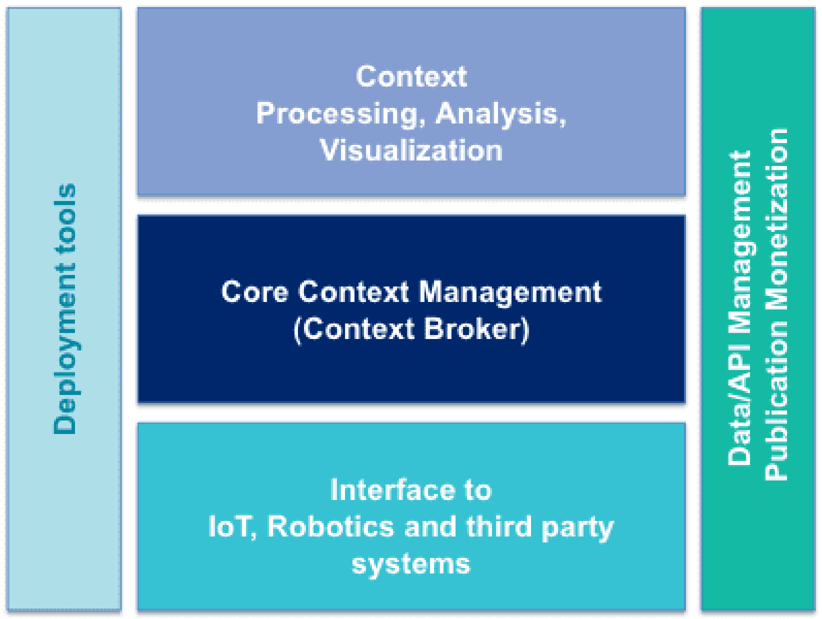

Deployment tools
Context Processing, Analysis, Visualization
Core Context Management (Context Broker)
Interface to IoT, Robotics and third party systems
Data/API Management Publication Monetization

Figure 3.1: Block diagram representing the functionality areas of the FIWARE framework.

MySQL, MongoDB or AWS DynamoDB as well as BigData platforms like Hadoop, Storm, Spark or Flink.

- Draco — an alternative for Cygnus. It is based on Apache NiFi and is a dataflow system, supporting powerful and scalable directed graphs of data routing, transformation, and system mediation logic. It also offers an intuitive graphical interface.
- Cosmos — permits simpler Big Data analysis over context integrated with popular Big Data platforms (Spark and Flink).
- QuantumLeap — implements storage of context data into time series database (CrateDB or Timescale).

The *Interface to Internet of Things (IoT), Robotics and third party systems* block possesses generic enablers of IoT agents capable of interfacing with the most used IoT protocols, such as, LWM2M over CoaP, JavaScript Object Notation (JSON) or UltraLight over HTTP/MQTT, OPC-UA, Sigfox or LoRaWAN. The objective is to allow an easier retrieval of the context information from the IoT devices or to trigger them based on context updates. These IoT agents that FIWARE developed act as bridges between the IoT corresponding communication protocol with the NGSI protocol used by the FIWARE elements. Currently, several extra generic enablers are being developed to support more protocols such as OneM2M, ROS2, DDS, EPCIS and others.

As the name implies, the next block, *Context Processing, Analysis and Visualisation* has available generic enablers to ease the process of processing, analysing and visualising context information. The generic enablers are the:

- Wirecloud — makes it easier to develop web operational dashboards, highly customizable by end users.

- Kurento — alows real-time processing of media streams supporting both the transformation of video cameras into sensors and the incorporation of advanced video application functions.

- FogFlow — it is a distributed execution framework that supports dynamic processing flows over cloud and edges.

Two extra generic enablers are currently under development:

- Perseo — Complex Event Processing that enables events firing capable to send Hypertext Transfer Protocol (HTTP) requests, emails, tweets, Short Message Service (SMS), messages, etc.

- OpenVidu — abstraction layer for Kurento, to make media processing easier to program.

The block *Context Data/API Management, Publication and Monetization* deals with different aspects. One of them is the security aspect of an implementation of a FIWARE solution. For that, it introduces generic enablers, such as, the:

- Keyrock — it is an identity manager that establishes support to secure and private OAuth2-based authentication of users and devices, user profile management, privacy-preserving disposition of personal data, Single Sign-On and Identity Federation across multiple administration domains.

- Wilma — supports proxy functions within OAuth2-based authentication schemas. It also implements PEP functions within an XACML-based access control schema.

- AuthZForce PDP/PAP — supports PDP/PAP functions within an access control schema based on the XACML standard.

Other aspects this blocks deals with is the publication and monetization of context data resources aspect. In this case the available generic enablers are:

- CKAN extensions — brings a number of add-ons enabling to extend current capabilities of the world-leading CKAN Open Data publication platform to allow publication of datasets matching right-time context data, the assignment of access terms and policies to those datasets and the assignment of pricing and pay-per-use schemas to datasets.

- Biz Framework — implements backend support to context Application Programming Interface (API)/Data monetization based on open TM Forum Business APIs.

Current work is being done in extra generic enablers to allow the discovery of open datasets, to execute API management, to facilitate API access, monitoring and monetization, to implement independent FIWARE components security, to manage Users, Groups, and Roles using the SCIM v1.1 standard and to enable extra security checks over user passwords.

The final block is the *Deployment Tools* block. FIWARE modules can be deployed through standard containerization techniques. Furthermore, the majority of the FIWARE components are directly available as Docker images, hence, Docker is a preferred containerization solution to implement a FIWARE-based system.

Docker is an open source platform that allows the containerization of applications and uses images and containers to do so. An image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings [35]. A container is a standardized unit of software, and includes the code as well as the necessary dependencies for it to run efficiently, regardless of the computing environment where is being run [35]. Images turn into containers at runtime. The advantage that applications developed using this framework have is that they will always run the same way, regardless of the used infrastructure to run them.

Docker's technology objective is to separate application dependencies from infrastructure, based on the requirements of developers and system operators. It is similar to a virtual machine (VM), however, the VM virtualizes the hardware and Docker the Operating System, making it more portable and flexible.

FIWARE uses the FIWARE NGSI RESTful API [36] to support data transmission between components as well as context information update or consumption. The usage of this open source RESTful API simplifies the process of adding extra functionalities through the usage of one, or more, complementary FIWARE components previously introduced or third-party components. In this work, the version used was the NGSI-v2.

Chapter 4

System Architecture

The development of the system was based on an open source platform, FIWARE [37], to deal with the context information, thus, ready-to-use modules of this platform are an integrant part of the system's architecture, namely, the CB, the Cygnus and the security modules (Keyrock, Wilma). As a consequence, communication between modules is performed through REST requests, following the FIWARE NGSI RESTful API. This data transmission between modules entails both context information update and consumption, hence, writing into the system or reading from it. For the system to achieve the proposed objectives, different modules had to be implemented and coupled together with the FIWARE modules to create the final version of the system's architecture. These modules are the health data provider, responsible for acquiring the health data and sending it to the CB, and the data analysis module, responsible for reading data from the system and produce the forecasts based on that data. Implementation-wise, the health data provider was implemented in Typescript [38] and the data analysis module in Python [39].

In the remainder of this chapter, Section 4.1 presents the final system architecture and the flow of information within the system. Sections 4.2 – 4.5 discuss the architecture's individual modules in more detail.

4.1 Final Architecture

The system architecture, presented in Figure 4.1, includes both ready-to-use FIWARE modules as well as the implemented health data provider and data analysis modules, complemented with the indication of the communication ports used on the different modules to communicate between one another. The architecture also shows, for completion reasons, the FIWARE ready-to-use security modules (*Keyrock* and *Wilma*), configured by Riskivector, and the mobile app module (Flutter Dashboard), also developed by Riskivector, to be used by the HTQ personnel to interact with the system. These modules, however, are out of the scope of this work. All modules are inside the same network, thus, are able to communicate directly with each other.

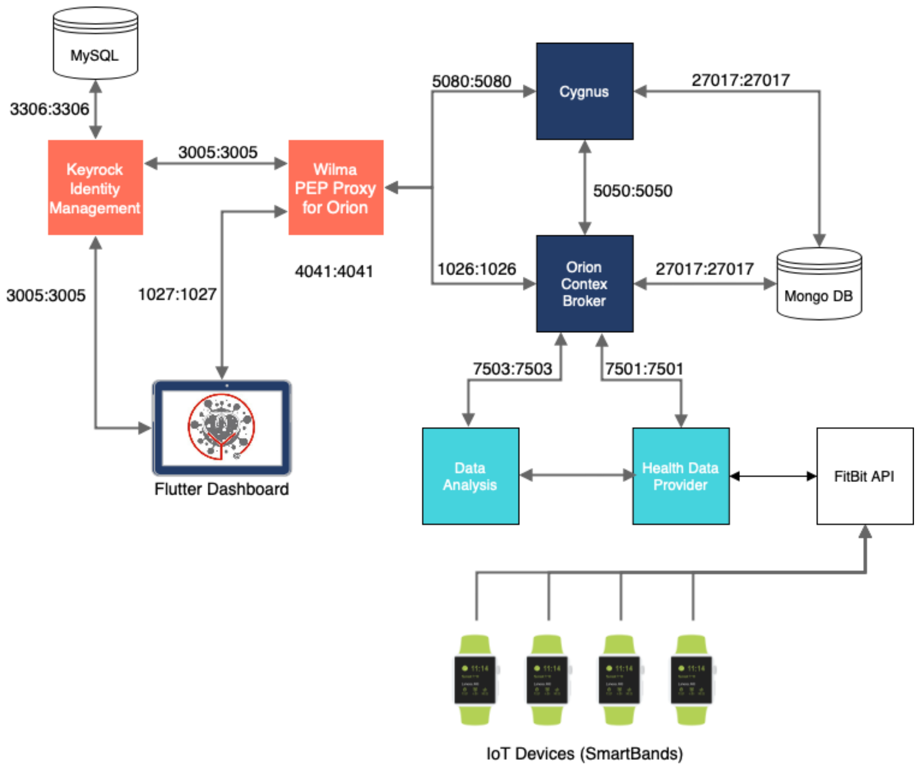


Figure 4.1: Complete system architecture.

Different types of interactions exist between the architecture modules, for different functionalities the system must have. For instance, Figure 4.2 shows the sequence diagram for adding users into the system.

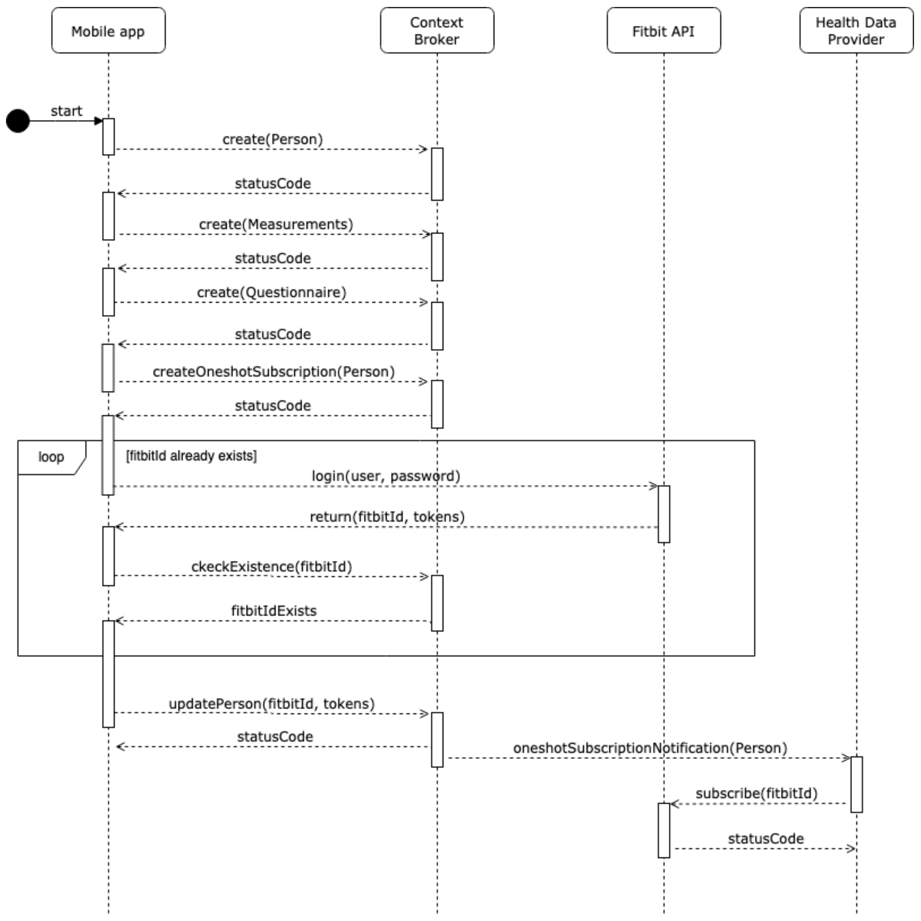


Figure 4.2: Sequence diagram for creating a new user in the system.

As can be seen, this operation requires collaborative work between four modules of the system:

- Mobile app — responsible for collecting and sending the necessary data, for the person creation, to the CB,
- CB — to include the data of the new person, and related entities, into the system,
- Fitbit API — to allow the user to log in into his Fitbit account, associating this account with the person in the system and to provide access to their data.
- Health data provider — to subscribe new data notifications, from Fitbit, whenever new data is detected at the Fitbit servers.

Before introducing a new user into the system, a check should be performed to ensure that there is only a single user in the system with a given set of Fitbit login credentials, thus, guaranteeing that a Fitbit account can only be associated with a single person in the system.

There are two sources of data: (1) mobile app, for manual input data, and (2) health data provider, for data from Fitbit. Considering the data source, there are slight differences in the methodology to get the data into the system. For the manual input case, the sequence diagram that shows the flow of messages used to write data into the system is presented in Figure 4.3.

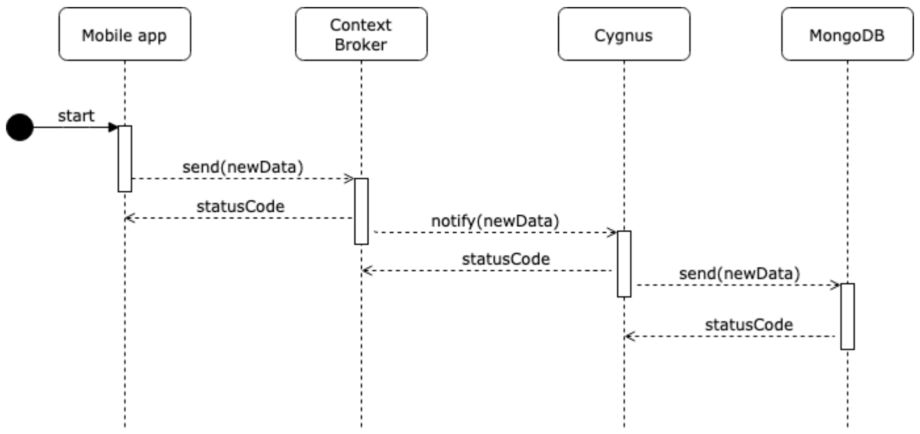


Figure 4.3: Sequence diagram for data fetching from the mobile app.

After the mobile app has sent the data to the CB, this module identifies that data update is occurring and deploys the corresponding notitifications to *Cygnus*. These notifications include in its body the data related to the entity that was updated, hence, *Cygnus* reads that data from the notification and persists it in the *MongoDB*.

The second data source possibility — Fitbit — requires a different approach. When Fitbit detects new data availability, the Fitbit API notifies the health data provider about this, identifying the *fitbitId* that has the new data available. This allows the system to identify the person and the corresponding measurement entity, to know what should be updated in the system. Before data fetching can be executed, a token validity action has to be performed to verify that data retrieval does not fails due to an invalid token. This operation is executed by using the current token in the system to fetch the Fitbit user profile associated with the identified *fitbitId*. If invalid, new tokens have to be requested from the Fitbit API. The next step, implemented by the health data provider, is to fetch the new data from the Fitbit servers through the Fitbit API. Upon data reception, data parsing is done to convert the data received from Fitbit to the data model in use by the system. This allows data to be sent to the CB, that, in turn, notifies Cygnus about context changes and Cygnus persists the data in the MongoDB. Figure 4.4.

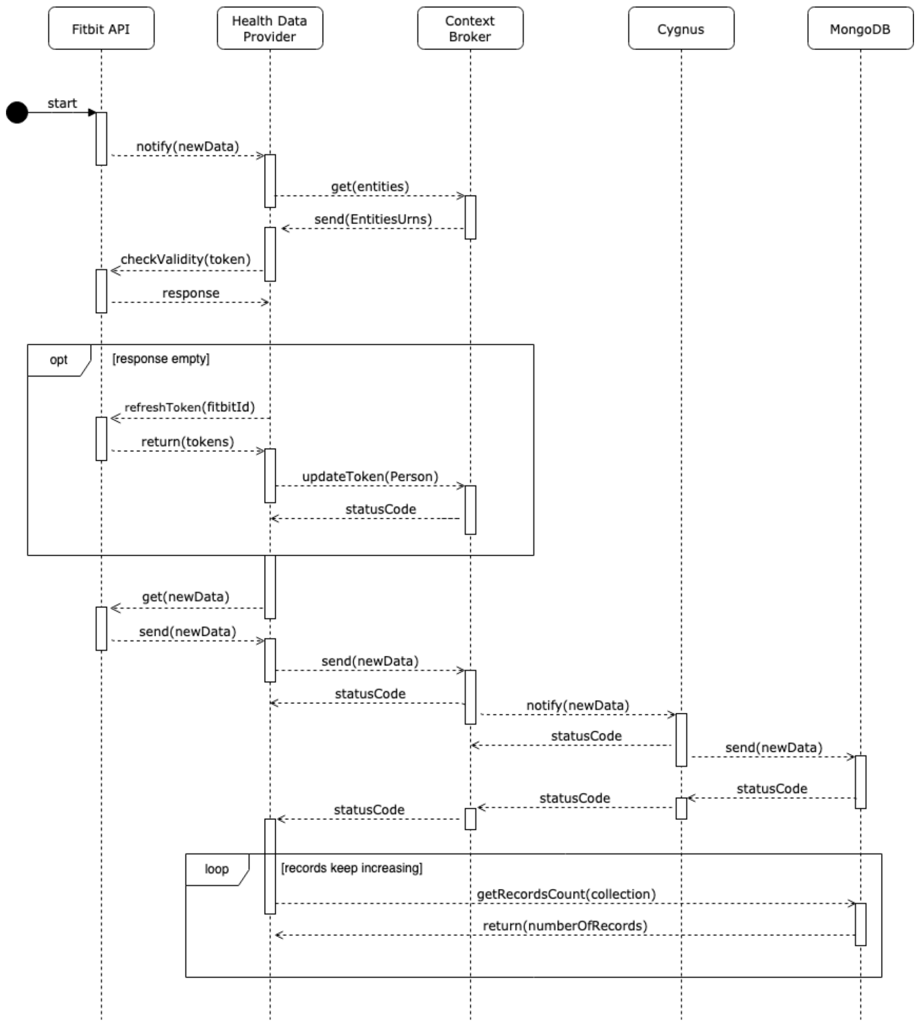


Figure 4.4: Sequence diagram for data fetching from Fitbit servers.

Having in consideration that there are two data sources, with different data fetching methods associated, and that new data should imply a new forecast should be calculated, the data analysis module can be triggered to create a new forecast from two different perspectives: new data arrived from manual input or new data arrived from the Fitbit servers.

In the first perspective, new data is sent to the CB and the relevant data, in terms of forecast execution, is the last observed value. This data can be read from the CB directly, hence, in order to trigger the forecast execution, when new data arrives at the CB, this module notifies the data analysis module. The person's entity *id* can be read from the body of this notification and the data analysis module uses this to discover all associated entities. Having this information, it can request the necessary data, from the CB and the MongoDB, to perform the forecast.

After forecast execution, the results are sent to the CB. If its the first forecast being made to that person, it creates the associated Forecast entity, otherwise, it updates the already existing Forecast entity. Figure 4.5.

In the second perspective, new data arrives from Fitbit and is sent to the MongoDB through the health data provider. When this module finishes sending the data to the database, it sends a notification to the data analysis module to inform it that new data was received from the Fitbit servers. This notification includes the person's entity *id*, to allow related entities discovery, from the CB, in order to allow data fetching from both the CB and the MongoDB. After collection the data, the module makes the new forecast and follows the same procedure of the first perspective to create the forecast in the CB. The steps for this operation can be seen in 4.6

4.2 Context Broker

Each module contributes for the system implementation by performing a set of given functionalities. The CB is responsible for:

1. keeping within itself the last state of the context information of the entire system;

2. sending notifications to the Cygnus module, whenever there's an update in the

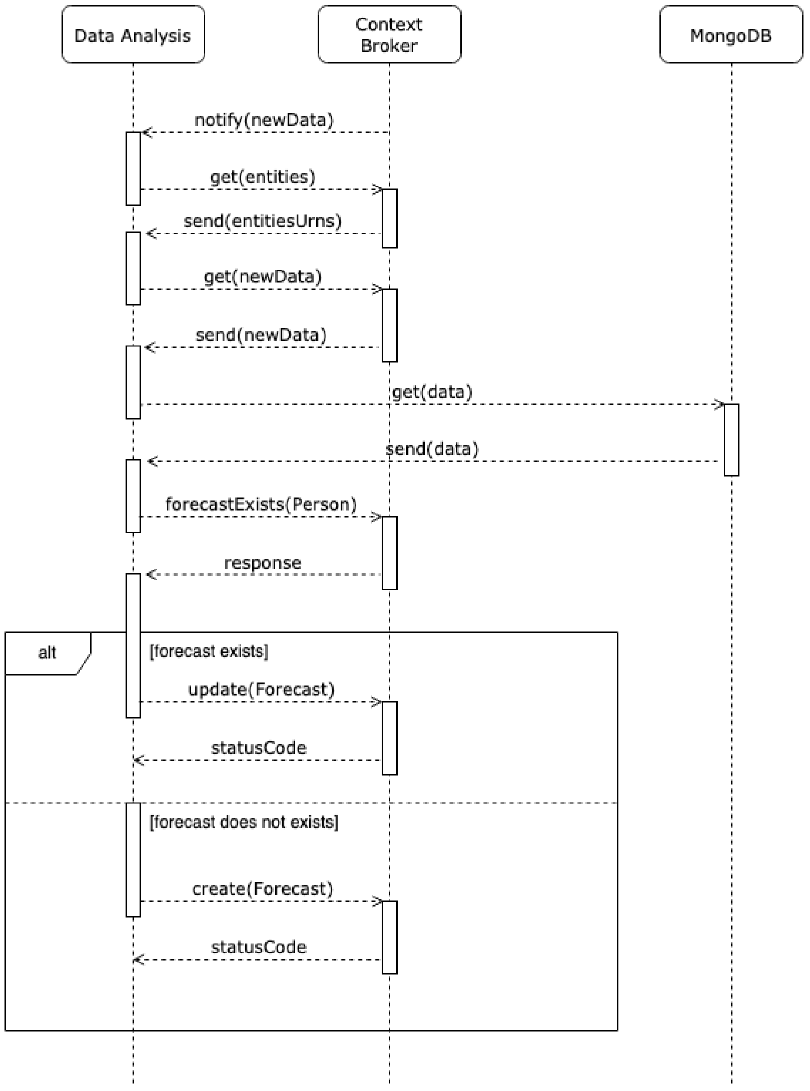


Figure 4.5: Sequence diagram for forecast execution triggered by manual input.

contextual information, to allow this module to perform data persistence in the MongoDB.

3. sending notifications to the health data provider, whenever a new person is inserted into the system, to allow this module to subscribe new data notifications with Fitbit.

4. sending notifications to the data analysis module, whenever there's data update in the system, to allow this module to execute a new forecast.

To be able to execute these actions, the CB needs to communicate with the Cygnus,

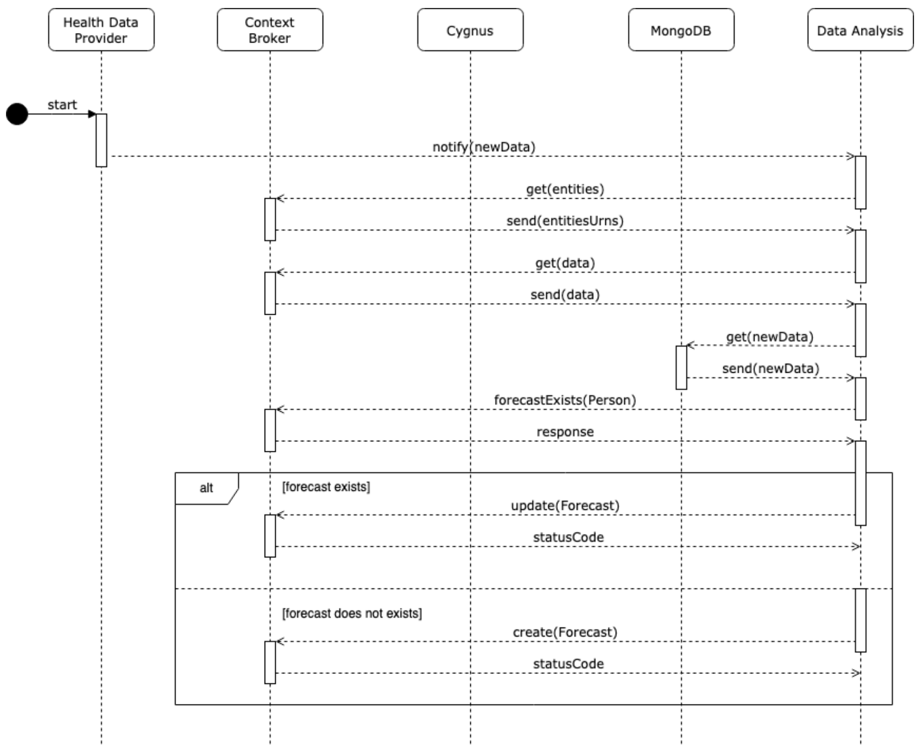
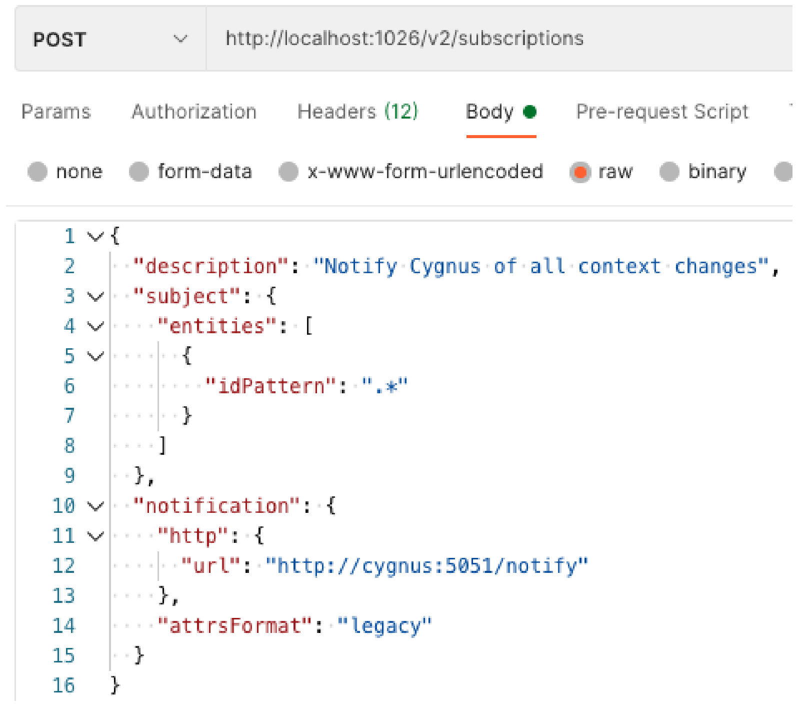


Figure 4.6: Sequence diagram for forecast execution triggered by Fitbit input.

the health data provider and the data analysis modules. To perform the aforementioned actions, it is necessary to create in the CB:

- a Cygnus subscription for all data changes, regardless of the entity those changes are related to, for action 2;
- a health data provider endpoint subscription for the person's Fitbit subscription execution, for action 3. Since the Fitbit subscription is only done once, per person, this type of subscription in the CB is a special subscription that can only be triggered one time — oneshot subscription —, created for a certain Person entity.
- a data analysis endpoint subscription of data update related to the variables used in the forecast execution. Since data has different sources, different types of subscriptions have to be executed.

FIWARE defines a subscription endpoint that is used to create all the necessary subscriptions through POST requests sent to this endpoint. Figure 4.7 shows the structure of the body of one request of this type, namely, the Cygnus subscription for data persistence.



```
http://localhost:1026/v2/subscriptions
```

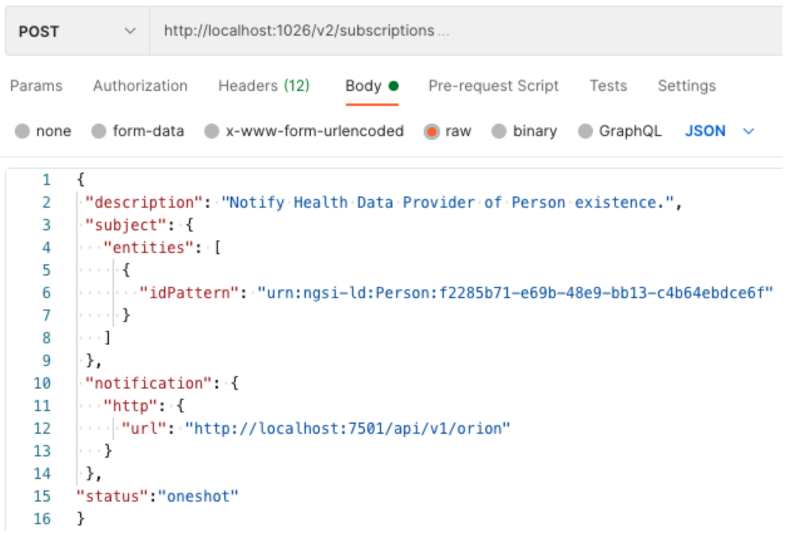
```
{
  "description": "Notify Cygnus of all context changes",
  "subject": {
    "entities": [
      {
        "idPattern": ".*"
      }
    ]
  },
  "notification": {
    "http": {
      "url": "http://cygnus:5051/notify"
    },
    "attrsFormat": "legacy"
  }
}
```

Figure 4.7: POST request body for the creation of the Cygnus subscription within the CB.

In the request, *idPattern* identifies the entity(ies) that the CB will look for data update, to trigger the notification that will be sent to the endpoint identified in the *url* sub-attribute of the *notification* attribute of the body of the request. Since Cygnus is a FIWARE module, using the default configuration, this endpoint is already defined as http://cygnus:5051/notify. This specific subscription, since identifies all entities, is responsible for the creation of the data persistence in the database, since it will notify the Cygnus module of all data updates the CB identifies.

The health data provider subscription request can be seen in Figure 4.8. This type of subscription is triggered only when it detects the first update of a person's data.

For all manual input variables, sent to the system from the mobile application, a different type of subscription is created within the CB. These subscriptions are triggered when a variable associated with them is updated in the CB and a notification is sent to the data analysis module so that a forecast based on the new data can be calculated.



http://localhost:1026/v2/subscriptions...

```
{
 "description": "Notify Health Data Provider of Person existence.",
 "subject": {
   "entities": [
     {
       "idPattern": "urn:ngsi-ld:Person:f2285b71-e69b-48e9-bb13-c4b64ebdce6f"
     }
   ]
 },
 "notification": {
   "http": {
     "url": "http://localhost:7501/api/v1/orion"
   }
 },
"status":"oneshot"
}
```

Figure 4.8: Oneshot subscription example.

4.3 Cygnus

The Cygnus module is a ready-to-use FIWARE module that uses a subscription based notification system to send data to a given database. The notification, triggered by the subscription, sent by the CB, includes in its body the subscriptionId and the entity that was updated, thus, Cygnus needs to grab this data from the notification and send it to the database. This means that the Cygnus module needs to be connected to the CB, to receive the data update notifications, and to the database, to send the data to it.

4.4 Health Data Provider

The health data provider is responsible for all aspects related to data acquisition from the wearable device. These include:

1. subscribing a new user in the system with the Fitbit services;

2. setting up a notification endpoint, to be used by the Fitbit servers, to deliver notifications about new data existence. The notification Fitbit sends to this endpoint identifies in its body the Fitbit user(s) that have new data available;

3. fetching new data, from the Fitbit servers, after availability of data is signalled by the Fitbit services;
4. sending fetched data into the CB;
5. calling the data analysis module when the fetched data is already in the system, in the MongoDB.

Actions 1 and 2 are part of the workflow to establish the subscriptions with Fitbit. Actions 3 – 5 represent activities about the normal operation of the system in data acquisition and management.

The interaction with Fitbit is done by the implementation of the notification endpoint, referred in 2, that will act as a callback URL, where near real-time updates are sent. Figure 4.9 shows the workflow to establish the subscription endpoint with the Fitbit servers. It is necessary to:

- create/configure a notification endpoint;
- make sure the endpoint is accessible to the Fitbit servers;
- verify the endpoint;
- authenticate users in the implemented mobile application;
- add data subscriptions, per user.

After the implementation of the notification endpoint, the URL needs to be disclosed to Fitbit, in the Fitbit application configuration, of the developer account that registered the application with Fitbit. For the verification of the endpoint, Fitbit sends two GET requests to the created endpoint, as shown in Figure 4.10 and, depending on the received replies, defines the endpoint as a valid/invalid subscription endpoint. For the endpoint to be defined as a valid one, it needs to reply with a 204 code to the first GET and with a 404 code to the second request. If these replies do not occur, the endpoint is signalled by Fitbit as an invalid one.

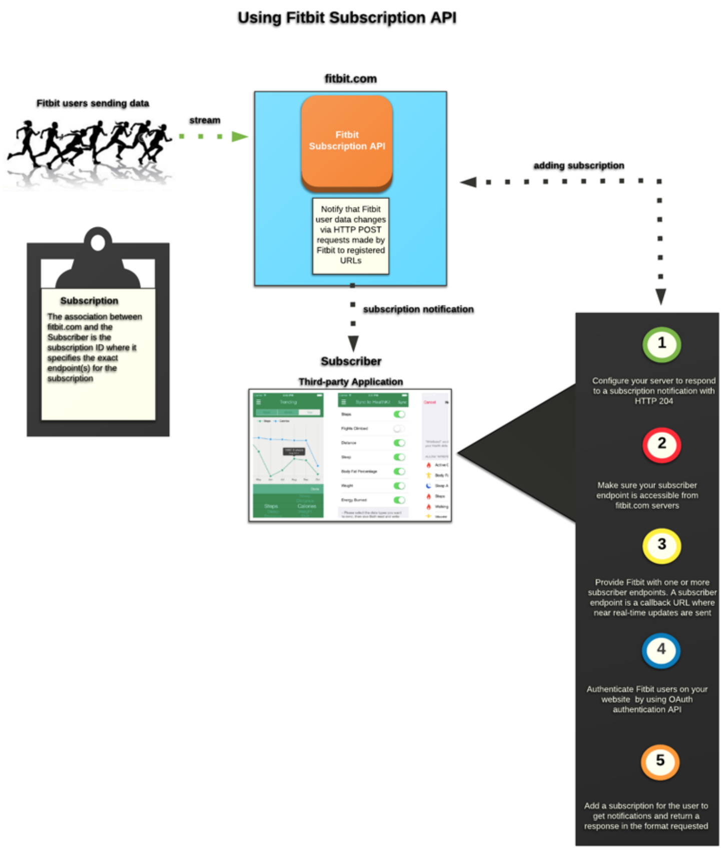

Using Fitbit Subscription API
Fitbit users sending data
stream
fitbit.com
Fitbit Subscription API
Notify that Fitbit user data changes via HTTP POST requests made by Fitbit to registered URLs
adding subscription
subscription notification
Subscription
The association between fitbit.com and the Subscriber is the subscription ID where it specifies the exact endpoint(s) for the subscription
Subscriber
Third-party Application
1
Configure your server to respond to a subscription notification with HTTP 204
2
Make sure your subscriber endpoint is accessible from fitbit.com servers
3
Provide Fitbit with one or more subscriber endpoints. A subscriber endpoint is a callback URL where near real-time updates are sent
4
Authenticate Fitbit users on your website by using OAuth authentication API
5
Add a subscription for the user to get notifications and return a response in the format requested

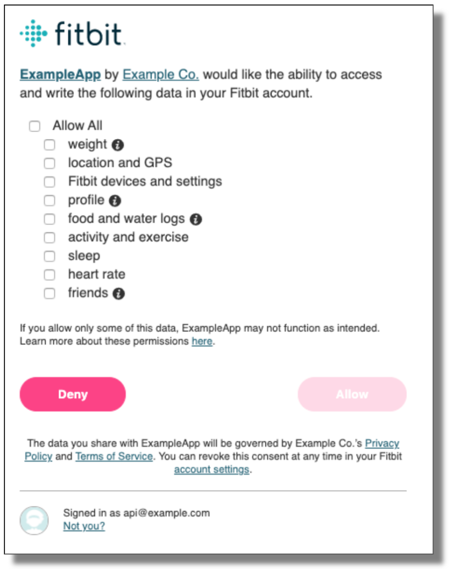
Figure 4.9: Fitbit subscription workflow, from [40].

Upon having a valid subscription endpoint classification, the endpoint can be used by Fitbit to signal new data availability. For this to happen, the user must give consent for the application to be able to access his data. This is done in the authorization stage: the user logs in with his Fitbit account details and is presented with a screen as the one of Figure 4.11. In this screen, the user can choose which scopes he authorizes the mobile app to collect data.

```
GET https://yourapp.com/fitbit/webhook?verify=correctVerificationCode
```

```
GET https://yourapp.com/fitbit/webhook?verify=incorrectVerificationCode
```

Figure 4.10: Fitbit subscription endpoint validation, adapted from [40].


fitbit

ExampleApp by Example Co. would like the ability to access and write the following data in your Fitbit account.

- [ ] Allow All
  - [ ] weight
  - [ ] location and GPS
  - [ ] Fitbit devices and settings
  - [ ] profile
  - [ ] food and water logs
  - [ ] activity and exercise
  - [ ] sleep
  - [ ] heart rate
  - [ ] friends

If you allow only some of this data, ExampleApp may not function as intended. Learn more about these permissions here.

Deny Allow

The data you share with ExampleApp will be governed by Example Co.'s Privacy Policy and Terms of Service. You can revoke this consent at any time in your Fitbit account settings.

Signed in as api@example.com
Not you?

Figure 4.11: Data scope authorization to provide data access for the application [41].

The access token generated in the authorization stage provides an association between user and application, by one side, and, by other side, it is a data access token, valid for the allowed scopes. This fact is used for the new data subscriptions creation with Fitbit. To create a subscription with Fitbit, for a new user, through the Fitbit API, a POST request is sent, without body, using the syntax present in Figure 4.12. In this syntax, *[collection-path]* identifies the data collection to receive notifications about. In this work, the collection of interest is *activities*. The *[subscription-id]* value has to be unique and provides a way to associate an update with an user — when a notification is sent about this user, the *[subscription-id]* is an attribute of the JSON notification body. This allows for data fetching to be controlled through the *fitbitId* or through the *subscription-id*. We control it using the *fitbitId*.

```
POST https://api.fitbit.com/1/user/-/[collection-path]/apiSubscriptions/[subscription-id].json
```

Figure 4.12: REST request syntax for subscription creation [40].

User association is performed in the request headers by the inclusion of the access token in one of the headers that have to be sent in the request. In fact, there are three mandatory headers:

- Accept: 'application/json'
- Authorization: 'Bearer *accessToken*'
- Content-Length: 0

The *Authorization* header includes the user's access token and, despite the fact that the request has no body, the *Content-Length* header has to be explicitly set to 0, otherwise, the Fitbit servers won't create the subscription.

After subscription creation, Fitbit sends notifications, to the configured endpoint, identifying new data availability, so that the system can respond accordingly, by fetching the new data from the Fitbit servers. The process of data fetching will be discussed in a more detail in Section 5.1.

4.5 Data Analysis

This module is responsible for reading updated data from the system and to produce forecasts based on that data. The data is stored both in the CB and in the MongoDB, thus, this module needs to connect to both modules. Furthermore, the health data provider must signal the data analysis unit when new data is collected from the Fitbit servers. This means that connectivity to the health data provider is also necessary. In conclusion, the data analysis module connects to the CB, the MongoDB and the health data provider.

This module is activated based on subscriptions that trigger notifications whenever a data update is detected on the attributes identified at the subscription creation. The subscriptions used to trigger the data analysis module identify changes in the relevant variables to the forecast, thus, there are different subscriptions aligned with the different data sources:

- CB:
  - *Questionnaire* entity, for the relevant attributes $\rightarrow$ anosmia, contactCovid, cough and dyspnoea

  - *Measurement* entity, using the *measurementType* value to implement different subscriptions: blood pressure, diarrhea, oxygen saturation and body temperature.

- health data provider:

  - notification identifying new heart rate data and the person associated with that data.

Upon notification reception, this module performs the necessary steps to read the data from the system, creates a new forecast, based on this new data, and sends the recent calculated forecast mark to the CB.

Chapter 5

Data Model and Analysis

The FIWARE NGSI-v2 API is used to deal with data representation in the system. This API establishes rules for entity definition and, per those rules [42], an entity has to have mandatory attributes, namely, the *id* and the *type* attributes. The *id* is created according with the format *"urn:ngsi-ld:<entity-type>:<entity-id>"* and represents an unique Uniform Resource Name (URN). In this work, the entity's *type* is identified in each entity definition, as can be seen in the data model presented in Fig. 5.1. Hence, the used entity types are *Person*, *Questionnaire*, *Measurement*, *Organization* and *Forecast*.

Given the fact that each *id* has to be unique, the *<entity-id>* part of the URN was created by using a Universally Unique IDentifier (UUID) version 4 generator. Using these rules, a person's entity *id* may be, for instance:

urn:ngsi-ld:Person:4bc3262c-e8bd-4b33-b1ed-431da932fa5

These rules, logically, are applied to all different entities in the system.

To define the data model, extra attributes were used, in each entity, to convey the necessary information between modules. Attribute naming conforms to the NGSI-v2 API guidelines [42]. The *Person* entity has attributes to specify the user characteristics and to associate the person with his Fitbit account. It also identifies the medical facility the patient is associated with. The *Measurement* entity represents the acquired data values,

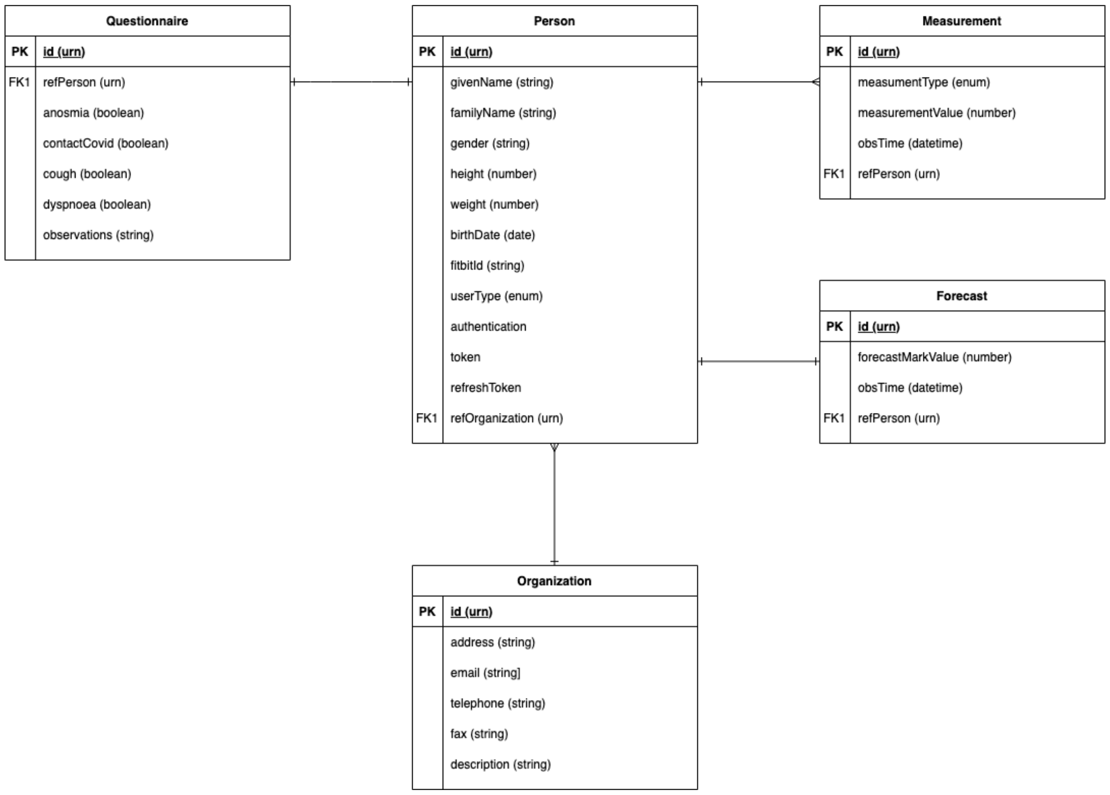


Figure 5.1: Data Model

its corresponding time of observation and the identification of the person those measurements belong to. This entity is used to specify information regarding the temperature, oxygen saturation, blood pressure, heart rate and daily number of diarrhea occurrences. The *measurementType* attribute is used to indicate what type of variable is being used on that specific entity. To ensure maximum interoperability possible, the Fast Healthcare Interoperability Resources (FHIR) specification of Health Level Seven International (HL7) [43] was used to define terms, whenever possible. This resulted on the following terms being used:

- temperature: *bodytemp*
- oxygen saturation: *oxygensat*
- blood pressure: *bp*
- heart rate: *heartrate*

- number of dayly diarrhea occurrences: *diarrhea* (FHIR non-compliant)

The *Questionnaire* entity specifies data that has to be acquired through the response of a questionnaire, by the user, through the mobile application. It identifies variables that are read manually into the system and provides an *observations* attribute, to allow the medical staff to enter notes that are relevant to the medical condition of the patient. Finally, it also has a relationship attribute to associate the *Questionnaire* entity with a *Person* entity.

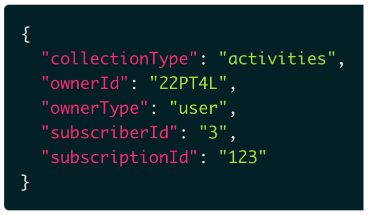
The *Organization* entity identifies the medical facility that the person is associated with, thus, has usual identification attributes used for an institution description. The *Forecast* entity provides the forecast mark value, calculated according with the classification table defined by the HTQ personnel, and the corresponding time of calculation. It also relates the *Forecast* entity with a *Person* through the relationship attribute *refPerson*.

5.1 Data Acquisition

Some aspects of data acquisition were already discussed in Section 4.4, such as, the fact that there are two different data sources that have to be taken into account.

The mobile application is responsible for manual data gathering, sending this data to the CB through POST requests. For instance, data related to the attributes of the *Questionnaire* entity is sent to the CB using this methodology. Fetching data from this source implies entity identification, using the NGSI-v2 API queries to that effect, based on the entity's *id* that had a data update, and entity's content reading, from the CB, to get the data.

Considering the second data source, Fitbit provides an API for data acquisition that, when new data is uploaded to the Fitbit servers, sends notifications to a configured/verified subscriber endpoint. These notifications are similar to the one represented in Fig. 5.2 and its reception allow the system to start the data fetching process, from the Fitbit servers. As can be seen in Figure 5.2, the notification identifies both the *fitbitId* (*ownerID*) of the person that has new data available and the *subscriptionId* (defined in the subscription creation stage), thus, allowing the system to fetch data from the identified user.



```
{
  "collectionType": "activities",
  "ownerId": "22PT4L",
  "ownerType": "user",
  "subscriberId": "3",
  "subscriptionId": "123"
}
```

Figure 5.2: Example of a notification Fitbit sends to notify the application about new data availability in the Fitbit servers [40].

This results in the following workflow, to get the data into the system:

1. create a Person entity;

2. associate Fitbit account with that person;

3. make a oneshot subscription (it is only triggered one time — oneshot) for that Person entity, notifying the Fitbit subscription endpoint;

4. update the Person entity with the Fitbit details;

5. create the fitbit subscription upon the oneshot notification reception;

6. get new data from Fitbit web API after receiving a notification from Fitbit: collect the *fitbitId* from the notification and use it to get the data from the Fitbit servers, from the last data update instant up until the present time.

The reception of such notification at the health data provider, triggers the endpoint responsible for acquiring new data from the Fitbit servers. The content of the notification is read to get the *fitbitId* associated with the new available data, thus, allowing the identification of the person who generated that data. The next step is to fetch the data, and, to that end, the operations identified in the workflow depicted in Figure 5.3 are executed in the presented order.

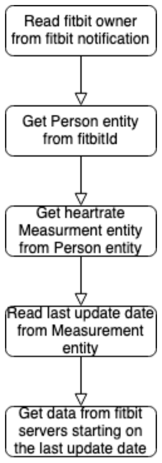
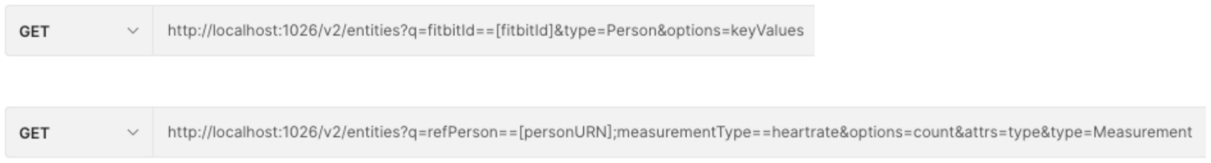

Read fitbit owner from fitbit notification
Get Person entity from fitbitId
Get heartrate Measurment entity from Person entity
Read last update date from Measurement entity
Get data from fitbit servers starting on the last update date

Figure 5.3: Data fetching workflow.

In Figure 5.3, the operations related to entities identification are done through the NGSI-v2 FIWARE API, by making queries to the CB. The first query, to get the Person entity, is on a one-to-one relationship, while the second, to get the *Measurement* entity, is on a one-to-many relationship and the idea is to read information from parent to child entity, resulting in the syntax presented in Figure 5.4.



```
GET  http://localhost:1026/v2/entities?q=fitbitId==[fitbitId]&type=Person&options=keyValues

GET  http://localhost:1026/v2/entities?q=refPerson==[personURN];measurementType==heartrate&options=count&attrs=type&type=Measurement
```

Figure 5.4: GET requests syntax for *Person* and *Measurement* entities identification. Adapted from [44].

The last action of the workflow, getting the data from the Fitbit Web API, involves in itself a set of steps that have to be followed to ensure correct data retrieval. These steps are shown in Figure 5.5.

As can be seen:

- the access token is read from the previously identified Person entity;
- this token is verified by using it to request the corresponding Fitbit profile of the person. If successful, token is valid, otherwise it is invalid.

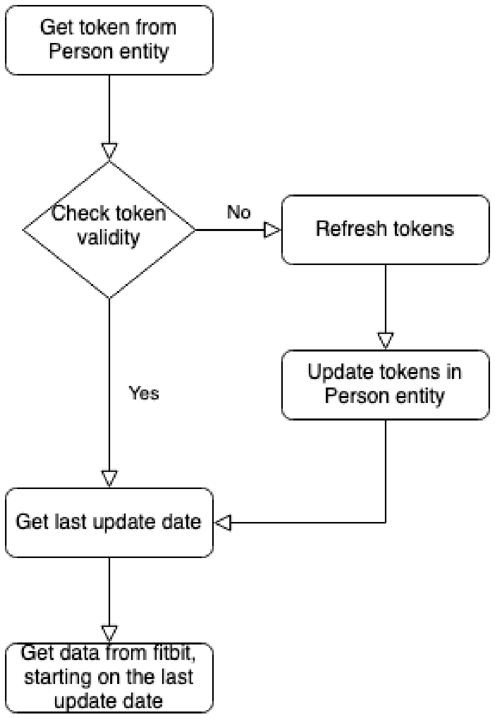
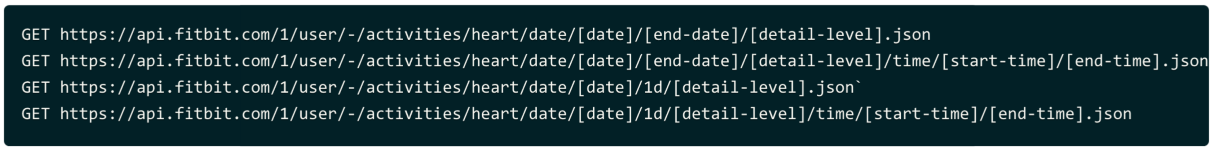

Get token from Person entity
Check token validity
No
Refresh tokens
Update tokens in Person entity
Yes
Get last update date
Get data from fitbit, starting on the last update date

Figure 5.5: Workflow to get new data from Fitbit servers.

- if the token is invalid, new access and refresh tokens are requested from Fitbit and updated within the system. This is done by sending a specific REST request to the Fitbit servers defined in the Fitbit API.

- after guaranteeing that the tokens are valid, the last update date (date up until there's already data in the system) is read from the *Measurement* entity and is used to read new data from that point in time up until the present instant.

The process of reading data from Fitbit conforms to the Fitbit Web API, that establishes four possible syntaxes to get data from their servers. These syntaxes are the ones presented in Figure 5.6.



```
GET https://api.fitbit.com/1/user/-/activities/heart/date/[date]/[end-date]/[detail-level].json
GET https://api.fitbit.com/1/user/-/activities/heart/date/[date]/[end-date]/[detail-level]/time/[start-time]/[end-time].json
GET https://api.fitbit.com/1/user/-/activities/heart/date/[date]/1d/[detail-level].json`
GET https://api.fitbit.com/1/user/-/activities/heart/date/[date]/1d/[detail-level]/time/[start-time]/[end-time].json
```

Figure 5.6: GET syntax to fetch heart rate data from Fitbit servers [45].

In these syntaxes, *date/end-date* is in the format **yyyy-MM-dd** or **today**, *detail-level* represents the sampling period, *start-time/end-time* is in the format **HH:mm**.

A period of observation can be defined using these variables, to establish the interval to get data from the Fitbit servers. After some tests, we took the decision to use a five minute sampling period (*detail-level* takes the value *5min*) and to get the data in daily batches. To achieve that purpose, data fetch methods were implemented to deal with the different possible intervals that may appear for data synchronization:

- last day:
  - start of synchronization is on a previous day → on the present day, data has to be collected from 00:00 up until the present instant.
  - start of synchronization is on the same day → data is collected from last synchronization time up until the present instant.
- multiple day interval
  - two days interval → collect data from last synchronization time up until 23:59 of the start day and collect the present day data.
  - more than two days interval → repeat the process of two days, but collecting in between the data from each intermediary day (00:00 → 23:59).

Data is received in a JSON, placed in a *dataset* array inside an *activities-heart-intraday* attribute. An example of such response is presented in Figure 5.7, where the first samples of the response are displayed. A data parser was implemented to read this data from the JSON response from Fitbit and forward this data to the CB, updating the corresponding *Measurement* entity value by value. Before executing this data forwarding, the data parser needs to do some pre-processing on the data, namely, on the observation time, because Fitbit provides the date and the time separately and the data model represents it jointly, according with ISO 8601 [46] in the format *YYYY-MM-dd*T*HH:mm:ss.mmm*Z, e.g., 2021-06-25T11:35:45.000Z.

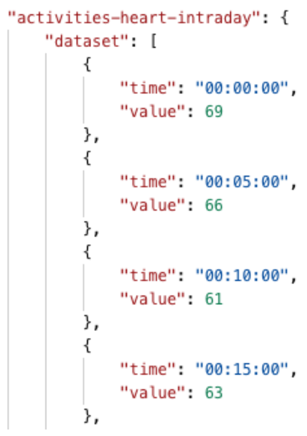

```
"activities-heart-intraday": {
    "dataset": [
        {
            "time": "00:00:00",
            "value": 69
        },
        {
            "time": "00:05:00",
            "value": 66
        },
        {
            "time": "00:10:00",
            "value": 61
        },
        {
            "time": "00:15:00",
            "value": 63
        },
```

Figure 5.7: Fitbit response example of fetching heart rate data.

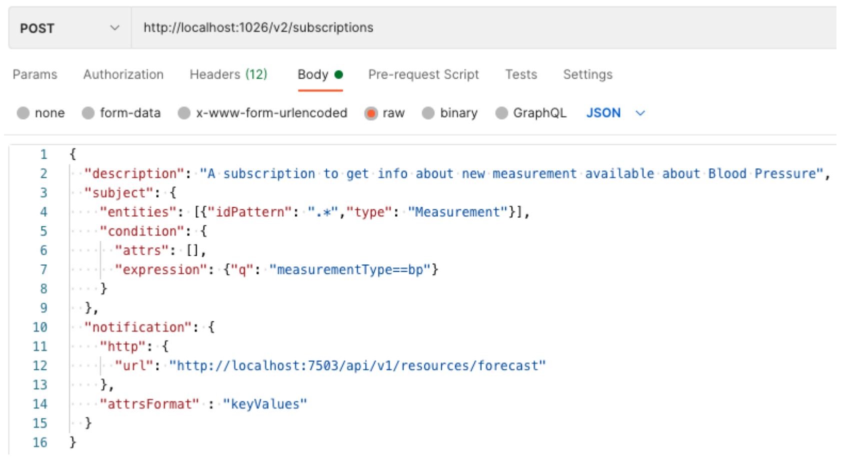
After parsing all data, the health data provider notifies the data analysis module about the existence of new data, to allow it to produce a new forecast, based on this new data. The notification's body include two attributes:

- *subscriptionId*: identification of the source of the data $\rightarrow$ *fitbit*;
- person's entity *id*: identification of the person the data belongs to by sending the corresponding person's entity *id URN*.

5.2 Data Analysis and Forecast Execution

When new data arrives into the system, the data analysis module is called to perform a forecast, based on the new available data. Given the data sources at hand, there are two methods to call this module: (1) subscription based from the CB or (2) notification from the health data provider.

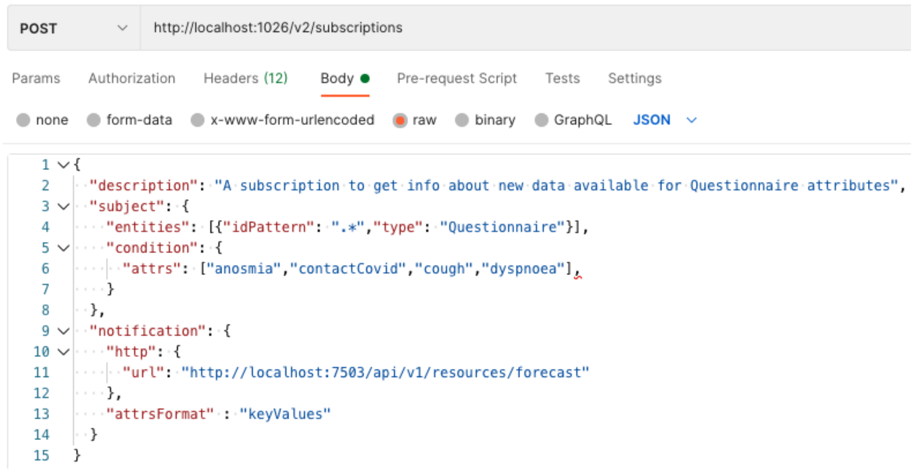
In the first method, subscriptions are created within the CB to notify the data analysis module about data change. This method is used with the *Questionnaire* and *Measurement* entities. For the latter, individual subscriptions have to be created for each possible *measurementType* value: oxygen saturation (*oxysat*), blood pressure (*bp*), number of daily diarrhea occurrences (*diarrhea*) and temperature (*bodytemp*). With this methodology, every time new data arrives at the CB about any of these variables, the subscription is triggered and the CB sends the notification to the forecast endpoint, identifying the *subscriptionId*, which this module uses to perform a correct identification of the entities it needs to look after, to collect the data, and the content of the entity that triggered the subscription. These subscriptions only need to be created once, at system start up. Examples of subscription creation, at the CB, for the *Measurement* (blood pressure) and *Questionnaire* entities are shown in Figure 5.8.



```
http://localhost:1026/v2/subscriptions
```

```
{
  "description": "A subscription to get info about new measurement available about Blood Pressure",
  "subject": {
    "entities": [{"idPattern": ".*","type": "Measurement"}],
    "condition": {
      "attrs": [],
      "expression": {"q": "measurementType==bp"}
    }
  },
  "notification": {
    "http": {
      "url": "http://localhost:7503/api/v1/resources/forecast"
    },
    "attrsFormat" : "keyValues"
  }
}
```

(a) Measurement entity (blood pressure example).



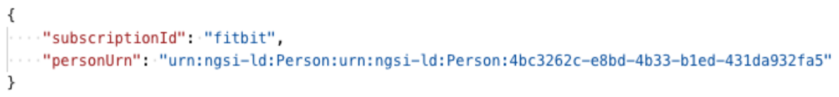
```
http://localhost:1026/v2/subscriptions
```

```
{
  "description": "A subscription to get info about new data available for Questionnaire attributes",
  "subject": {
    "entities": [{"idPattern": ".*","type": "Questionnaire"}],
    "condition": {
      "attrs": ["anosmia","contactCovid","cough","dyspnoea"],
    }
  },
  "notification": {
    "http": {
      "url": "http://localhost:7503/api/v1/resources/forecast"
    },
    "attrsFormat" : "keyValues"
  }
}
```

(b) Questionnaire entity.

Figure 5.8: CB subscription, to send notifications to the data analysis module about context changes in the Measurement and Questionnaire entities.

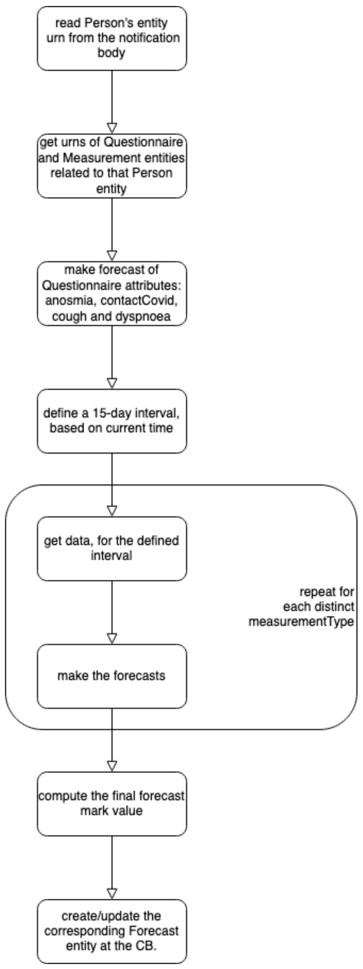
The second method is used when there's new data from the Fitbit servers. Upon data collection from the Fitbit servers, the data is sent to the CB, being persisted by *Cygnus* in the database. Afterwards, the data analysis endpoint is called with a notification sent by the health data provider, as stated in Section 4.4, that includes a static *subscriptionId* defined as *fitbit* and the person's entity *id* URN, as depicted in Figure 5.9.



```
{
    "subscriptionId": "fitbit",
    "personUrn": "urn:ngsi-ld:Person:urn:ngsi-ld:Person:4bc3262c-e8bd-4b33-b1ed-431da932fa5"
}
```

Figure 5.9: Notification sent by the health data provider to the data analysis module.

Considering the body of the notification received by the data analysis module, the flowchart presented in Figure 5.10 is executed to perform the forecast.



read Person's entity urn from the notification body
get urns of Questionnaire and Measurement entities related to that Person entity
make forecast of Questionnaire attributes: anosmia, contactCovid, cough and dyspnoea
define a 15-day interval, based on current time
get data, for the defined interval
make the forecasts
repeat for each distinct measurementType
compute the final forecast mark value
create/update the corresponding Forecast entity at the CB.

Figure 5.10: Flowchart of the operations the data analysis module executes to create forecasts.

After reading the Person's URN, queries to the CB are implemented to get the URNs of all *Measurement* and *Questionnaire* entities related with that person, to allow data collection from the system about all variables: (1) anosmia, (2) contactCovid, (3) cough, (4) dyspnoea, (5) blood pressure, (6) numberDailyDiarrhea, (7) oxygen saturation, (8) temperature and (9) heart rate.

Data history of a certain variable is used to forecast that variable:

- variables $1-4$ are boolean variables that tend to maintain the value in consecutive observations, however, they may have sudden and unpredictable changes in value. This type of behaviour configures the random walk model, for which the advised forecast consists in the last observed value [47], that can be read from the CB. This implies the forecast is the one presented in Equation 5.1:

$$\hat{y}_{t+1} = y_t \tag{5.1}$$

- variables $5-8$ are variables with slow variation and small dynamic range. In terms of forecasting, the trend is a regular, slowly evolving change in the series level and can be modeled by low-order polynomials. Given these two facts, the forecast used for these variables was a linear trend estimator, based on least squares estimation [47]. The least squares principle provides a way of choosing the coefficients effectively by minimising the sum of the squared errors, identified in Equation 5.2:

$$\sum_{t=1}^{T} \varepsilon_t^2 = \sum_{t=1}^{T} (y_t - \beta_0 - \beta_1 x_t)^2 \tag{5.2}$$

  The resulting coefficients, $\hat{\beta}_0$ and $\hat{\beta}_1$, are then used to make the forecast according with Equation 5.3:

$$\hat{y}_{t+h} = \hat{\beta}_0 + \hat{\beta}_1 x_{t+h} \tag{5.3}$$

- variable 9 has a seasonal variation that can be observed on a daily basis, thus, a forecast algorithm that can take seasonal effects into consideration had to be used. To choose one method, it was considered the usage of a neural networks, AutoRegressive Integrated Moving Average (ARIMA) models and the additive Holt-Winters method. Based on the results accuracy and computation time, the additive Holt-Winters forecast method [47] presented in Equations 5.4 – 5.5.

$$\begin{aligned}\ell_t &= \alpha(y_t - s_{t-m}) + (1-\alpha)(\ell_{t-1} + b_{t-1})\\ b_t &= \beta^*(\ell_t - \ell_{t-1}) + (1-\beta^*)b_{t-1}\\ s_t &= \gamma(y_t - \ell_{t-1} - b_{t-1}) + (1-\gamma)s_{t-m},\end{aligned} \tag{5.4}$$

In Equation 5.4, ℓ_t represents the level, b_t the trend and s_t the seasonality components. The forecast is performed using these values and is illustrated in Equation 5.5.

$$\hat{y}_{t+h|t} = \ell_t + hb_t + s_{t+h-m(k+1)} \tag{5.5}$$

The next step in the flowchart is to read the last observed values of the Questionnaire attribute values from the CB and use them as the forecast for those variables.

An observation interval of 15 days is set, based on the current time. Next, the process of forecasting, that includes reading the data from the MongoDB and forecast execution based on that data, is repeated for each numerical variable in play.

In this step, when forecasting the heart rate, the Holt-Winters method needs to be fed with periodic data in order to make the forecast and has to have at least data related to two seasons (two days). This means that data pre-processing has to be executed to ensure that this requirement is met. For instance, when the wearable device is being charged, it cannot, of course, read the patient's data, creating gaps that have to be processed before passing the data to the forecast algorithm. Hence, when a new forecast is executed, data is collected from the database, the data is verified to determine if there are gaps in the data and, if the answer is positive, forward and backward interpolation are used to fill those gaps. The final data, without gaps, is, afterwards, passed to the forecast method. Holt-Winters parameter optimization is executed first, using, to that end, the last known hour data, coupled with a Mean Absolute Error (MAE) metric. The optimized parameters

Table 5.1: Classification metrics for COVID-19 level identification provided by HTQ.

Variable	**0 Points**	**1 Points**	**2 Points**
Systolic Blood Pressure (mmHg)	$120 \leq x \leq 140$	$100 \leq x \leq 119$	$x \leq 99$
Heart Rate (bpm)	$56 \leq x \leq 85$	$86 \leq x \leq 100$ or $50 \leq x \leq 55$	$x \geq 101$ or $x \leq 49$
Oxygen Saturation (%)	$x \geq 94$	$90 \leq x \leq 93$	$x \leq 89$
Body Temperature (ºC)	$x \leq 36.9$	$37 \leq x \leq 39.9$	$x > 39.9$
Dyspnoea	No	N/A	Yes
Cough	No	N/A	Yes
Anosmia	No	N/A	Yes
Contact COVID+	No	N/A	Yes
Diarrhea	No	N/A	Yes (minimum of 5 times/day

are then used to make the forecast for the defined forecast horizon.

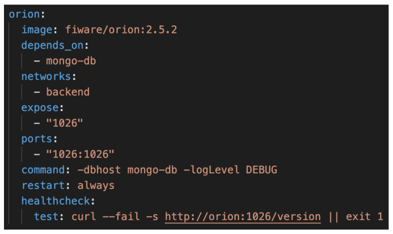
The final forecast mark is computed by adding all individual marks, calculated according with the rules specified in Table 5.1, defined by the medical personnel of HTQ [4]. This value is used with the following classification, to generate alerts:

- $value \leq 4$ points $\rightarrow$ unlikely COVID-19 $\rightarrow$ 24 hour vigilance;
- $value \geq 5$ points points $\rightarrow$ likely COVID-19 $\rightarrow$ patient should be directed quickly into the emergency room.

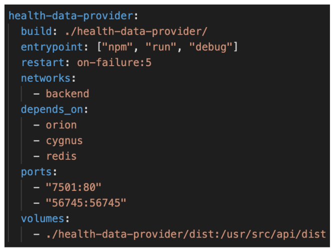
Chapter 6

Implementation and Results

Docker is used to turn the system online using a *docker-compose.yml* configuration file in which services are defined for each architecture module. As an example, Figure 6.1 shows the service definition for a FIWARE ready-to-use module, the context broker, *Orion*, and for an implemented module, the health data provider. This methodology has the advantage that allows the system to implement modularity, thus, allowing functionality separation between modules. As a consequence of this modularity, the system is easily scalable: to put more sources into the system there's only need to implement other health data provider modules, e.g., to add other brand of wearable devices.



```
orion:
  image: fiware/orion:2.5.2
  depends_on:
    - mongo-db
  networks:
    - backend
  expose:
    - "1026"
  ports:
    - "1026:1026"
  command: -dbhost mongo-db -logLevel DEBUG
  restart: always
  healthcheck:
    test: curl --fail -s http://orion:1026/version || exit 1
```



```
health-data-provider:
  build: ./health-data-provider/
  entrypoint: ["npm", "run", "debug"]
  restart: on-failure:5
  networks:
    - backend
  depends_on:
    - orion
    - cygnus
    - redis
  ports:
    - "7501:80"
    - "56745:56745"
  volumes:
    - ./health-data-provider/dist:/usr/src/api/dist
```

Figure 6.1: Service definition for docker set up to run the system.

For the ready-to-use modules, the service identifies the base image the module should use. For instance, in the case presented in Figure 6.1, the FIWARE orion image, version 2.5.2 is being used. For the implemented modules, the source code path is identified as well as the necessary commands to build that code. Both services identify the ports in use for that module.

Two modules were implemented:

- health data provider
- data analysis

Health data provider module is responsible for fetching data from the Fitbit API, thus, methods were implemented in this module that coupled together can achieve this objective:

- security methods for data access: get tokens; refresh tokens when they are invalid;
- data fetching methods: subscription creation with Fitbit; endpoint creation to receive new data notifications from Fitbit servers; data fetching for a given interval (daily-based fetching).

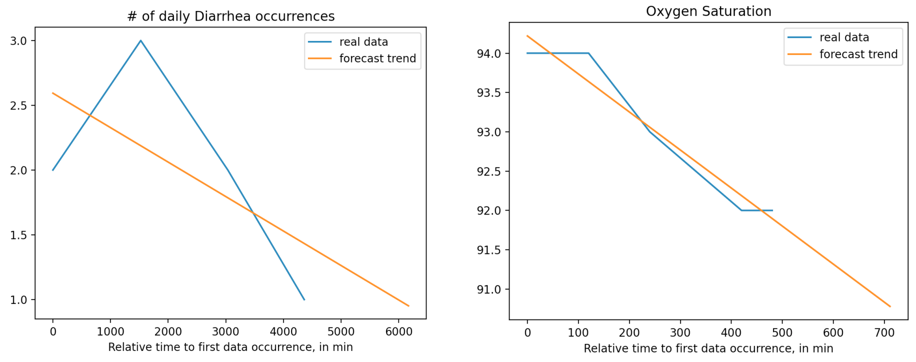
Data analysis module is responsible for creating forecasts. To that end, it was necessary to implement methods for:

- identifying the entities related to the necessary data;
- fetching data from the context broker and the MongoDB;
- implementing the forecast for the individual variables;
- implementing the final forecast based on the individual forecasts;
- sending the calculated forecast in the context broker.

In terms of forecasting results, there are three different types of forecast being executed, for the different type of variables at play:

- boolean variables with a random walk behaviour. As explained previously in Section 5.2, the forecast for this type of variables is the last observed value. This type of forecast is used with the following variables: anosmia, contactCOVID+, cough and dysponea;

- variables with slow variation and/or small dynamic range. Given these characteristics, a linear trend estimator was used with this type of variables, namely, the number occurrences of daily diarrhea, the body temperature, the blood pressure and the oxygen saturation level. Figure 6.2(a), that showcases the number of daily diarrhea occurrences example, in which the forecast value determined is for the next day. Other variable that also uses this type of forecast, but with a different forecast horizon (two hours), is the oxygen saturation level. An example is shown in Figure 6.2(b).



(a) Daily diarrhea example. (b) Oxygen Saturation example.

Figure 6.2: Trend estimator forecasting.

- variables with seasonal variation, namely, heart rate. The Holt-Winters additive method was used and needs parameter optimization.

For the last scenario, the Holt-Winters forecasting of the heart rate, two different metrics were used for the parameters optimization: MAE and Root Mean Square Error (RMSE) values. Table 6.1 presents the obtained results for the values of these metrics, for different data (days) of the same person.

Different simulations present different days/hours and data fetching situations: data is almost complete for the analysis scenario or data has considerable gaps (simulations four and five). MAE and RMSE values remain almost the same, for both metrics, and produce the same type of variation: an increase in MAE corresponds to an increase in RMSE. The exception is simulation four, that has the lowest MAE but does not have

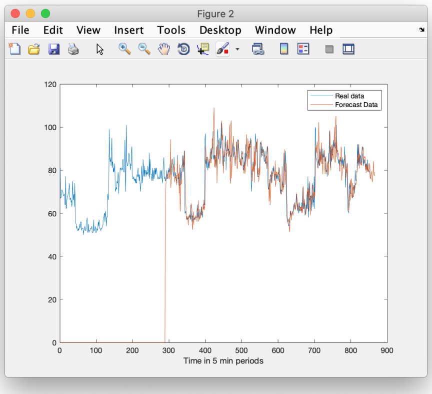
Table 6.1: Simulations of different heart rate data input for Holt-Winters parameters optimization using different metrics.

Simulation	**MAE metric**			**RMSE Metric**		
	$[\alpha^*, \beta^*, \gamma^*]$	**MAE***	**RMSE**	$[\alpha^*, \beta^*, \gamma^*]$	**RMSE***	**MAE**
1	$[0.6114, 0.0019, 0.6505]$	4.1224	5.5443	$[0.5859, 0, 0.7246]$	5.5311	4.1327
2	$[0.4790, 0, 0.7860]$	4.3682	5.8456	$[0.4866, 0, 0.7770]$	5.8454	4.3685
3	$[0.6236, 0, 0.2667]$	4.7648	6.7430	$[0.7457, 0, 0.7352]$	6.7067	4.7732
4	$[0.6841, 0.0005, 1]$	3.2325	5.2630	$[0.6508, 0.0004, 1]$	5.2498	3.2343
5	$[0.6167, 0, 0.7068]$	3.6561	5.1768	$[0.6950, 0, 0.686]$	5.1426	3.6715
6	$[0.7084, 0, 0.8121]$	3.9354	5.5081	$[0.6799, 0, 0.6823]$	5.5010	3.9408
7	$[0.7039, 0, 0.6790]$	3.6291	4.9786	$[0.7538, 0, 0.9987]$	4.9662	3.6328

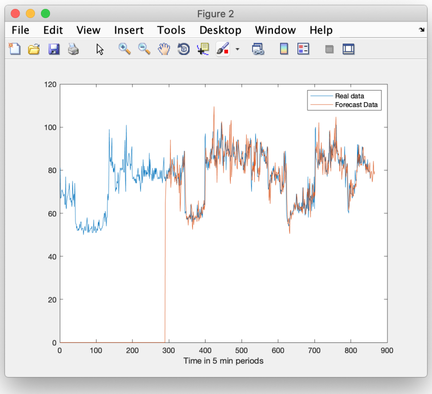
the lowest RMSE. This can be explained by the fact that this is a simulation in which the data retrieval interval includes a somewhat long data gap, indicating that in this type of situations, choosing the MAE metric can yield better results. Table 6.1 also shows that the difference in MAE and RMSE values, for both metrics, is negligible, with the minimum values occurring in the corresponding metric. Looking into the optimized parameters, most of the simulations have similar values using both metrics, however, there are simulations with different values, for each optimization metric. Despite this, however, the characterization of the MAE and RMSE for all these cases is still in accordance with the other simulations. Furthermore, from the simulation graphs presented in Figure 6.3, one can verify that there isn't a significative difference between the usage of both metrics, thus, the MAE metric was chosen to be used to define the optimization of the input parameters of the additive Holt-Winters method.

In order to achieve the final classification, as defined by the medical personnel of HTQ, the worst case scenario is the one that was considered. This means that minimum and maximum forecasted values are used together with the classification metrics defined in Table 5.1 to achieve the individual forecast marks, for each variable.

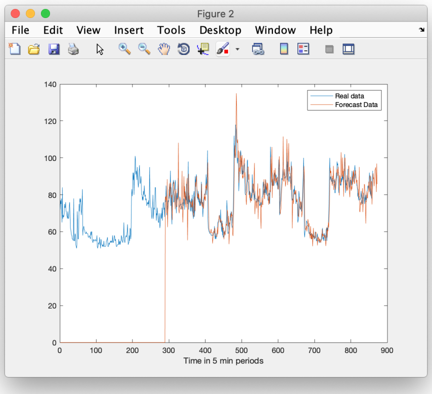
These marks are added to define the final mark and predict, according with the medical personnel criteria presented in Section 5.2, if the person is likely infected, or not, with COVID-19.



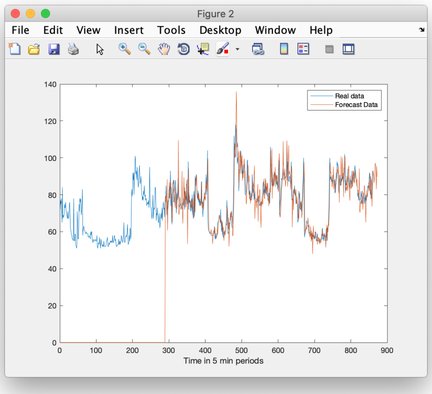
(a) Simulation 1 — MAE.



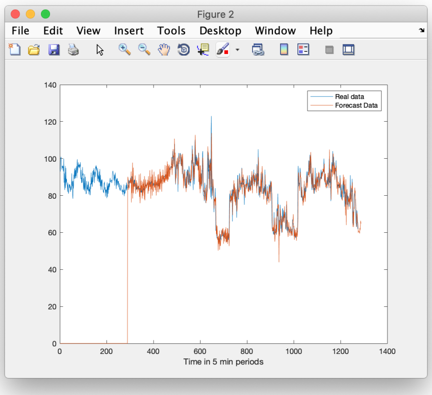
(b) Simulation 1 — RMSE.



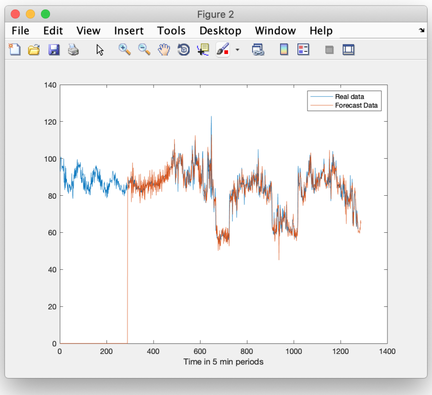
(c) Simulation 3 — MAE.



(d) Simulation 3 — RMSE.



(e) Simulation 6 — MAE.



(f) Simulation 6 — RMSE.

Figure 6.3: Forecast of heart rate for different scenarios.

Chapter 7

Conclusions

The technology development in wearable devices and its corresponding sensing capabilities facilitates the usage of this type of devices for elderly monitoring. Furthermore, there is nowadays a global pandemic in effect, with dire consequences for the elderly people. Within this context, this work presents the development of a system to monitor persons that may be, or not, infected with COVID-19, in order to forecast the situation that each person is in as soon as possible.

FIWARE platform modules were used, namely, the context broker and cygnus combined with a MongoDB and two implemented modules: health data provider and data analysis. The health data provider is responsible for getting data from the Fitbit servers and the data analysis is responsible for reading data from the system and use that data to create health status forecasts. To create the system, all these modules were containerized using Docker.

Three type of forecasts were implemented, considering the different type of variables that have to be taken in consideration: boolean variables, with a random walk behaviour are forecasted through the last observation; slow variation and/or small dynamic range variables are forecasted using linear trend estimators; variables with seasonal variation are forecasted by the additive Holt-Winters method, with input parameter optimization performed using a MAE metric.

The forecasts for the individual variables are taken into consideration to, in conjunction with the classification defined by the medical personnel, determine individual classification marks. These marks are added to calculate the final classification mark that defines if the person is likely to be, or not, infected by COVID-19.

The determination of the individual marks is based on a worst case scenario. This means that maximum and minimum values are taken (depending from the variable) from the forecasted values to apply the classification. By one side, this is the methodology to identify the most COVID-19 cases. However, since it is the worst case scenario, it is the methodology that is more prone to identify false positive cases.

The obtained results so far are encouraging, however, future work can be done to improve the system:

- the modularity of the system provides scalability to it. Work can be done to take advantage of this system characteristic, for instance, increasing the existing data sources of the system — only a new health data provider has to be implemented for a new data source, e.g., another wearable device with other sensing capabilities.
- other forecast methods can be taken into consideration to compare them with the current methods in use at the moment, to verify if accuracy can be improved.
- other final classification methods, other than the worst case scenario, should be investigated/implemented to check if it is possible to improve the results in terms of correct identification vs false positives.

Acknowledgements

This work was been supported by SMACovid-19 – Autonomous Monitoring System for Covid-19 (70078 – SMACovid-19).

Bibliography

[1] *Projetos – smacovid-19 – more colab*, `https://morecolab.pt/en/projetos-smacovid-19-en/`, (Accessed on 06/29/2021).

[2] *More colab – laboratório colaborativo montanhas de investigação - associação*, `https://morecolab.pt/`, (Accessed on 06/29/2021).

[3] *Riskivector*, `https://riskivector.com/`, (Accessed on 06/29/2021).

[4] *Hospital terra quente - mirandela*, `https://htq.pt/`, (Accessed on 06/29/2021).

[5] *Pordata - população residente: Total e por grandes grupos etários (%)*, `https://www.pordata.pt/Portugal/Popula%C3%A7%C3%A3o+residente+total+e+por+grandes+grupos+et%C3%A1rios+(percentagem)-3018-253437`, (Accessed on 07/01/2021).

[6] *Statistics portugal - web portal*, `https://www.ine.pt/xportal/xmain?xpid=INE&xpgid=ine_publicacoes&PUBLICACOESpub_boui=71882686&PUBLICACOESmodo=2`, (Accessed on 07/01/2021).

[7] B. David Chung Hu, H. Fahmi, L. Yuhao, C. C. Kiong, and A. Harun, "Internet of things (iot) monitoring system for elderly," in *2018 International Conference on Intelligent and Advanced System (ICIAS)*, 2018, pp. 1–6. DOI: `10.1109/ICIAS.2018.8540567`.

[8] T. F. Bernadus, L. B. Subekti, and Y. Bandung, "Iot-based fall detection and heart rate monitoring system for elderly care," in *2019 International Conference on ICT*

for Smart Society (ICISS), vol. 7, 2019, pp. 1–6. DOI: 10.1109/ICISS48059.2019.8969845.

[9] H. Liu, J. Huang, C. Lu, Z. Lan, and Q. Wang, "Indoor monitoring system for elderly based on zigbee network," in *2016 International Symposium on Micro-NanoMechatronics and Human Science (MHS)*, 2016, pp. 1–7. DOI: 10.1109/MHS.2016.7824172.

[10] K. E. Ansefine, Muzakki, Sanudin, E. Anggadjaja, and H. Santoso, "Smart and wearable technology approach for elderly monitoring in nursing home," in *2017 IEEE 3rd International Conference on Engineering Technologies and Social Sciences (ICETSS)*, 2017, pp. 1–6. DOI: 10.1109/ICETSS.2017.8324159.

[11] A. Gupta, R. Srivastava, H. Gupta, and B. Kumar, "Iot based fall detection monitoring and alarm system for elderly," in *2020 IEEE 7th Uttar Pradesh Section International Conference on Electrical, Electronics and Computer Engineering (UPCON)*, 2020, pp. 1–5. DOI: 10.1109/UPCON50219.2020.9376569.

[12] T. Elakkiya, "Wearable safety wristband device for elderly health monitoring with fall detect and heart attack alarm," in *2017 Third International Conference on Science Technology Engineering Management (ICONSTEM)*, 2017, pp. 1018–1022. DOI: 10.1109/ICONSTEM.2017.8261318.

[13] S. A. Waheed and P. Sheik Abdul Khader, "A novel approach for smart and cost effective iot based elderly fall detection system using pi camera," in *2017 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC)*, 2017, pp. 1–4. DOI: 10.1109/ICCIC.2017.8524486.

[14] M. Bundele, H. Sharma, M. Gupta, and P. S. Sisodia, "An elderly fall detection system using depth images," in *2020 5th IEEE International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, 2020, pp. 1–4. DOI: 10.1109/ICRAIE51050.2020.9358330.

[15] M. Frydrysiak and L. Tesiorowski, "Health monitoring system for protecting elderly people," in *2016 International Multidisciplinary Conference on Computer and Energy Science (SpliTech)*, 2016, pp. 1–6. DOI: `10.1109/SpliTech.2016.7555935`.

[16] P. Jiang, J. Winkley, C. Zhao, R. Munnoch, G. Min, and L. T. Yang, "An intelligent information forwarder for healthcare big data systems with distributed wearable sensors," *IEEE Systems Journal*, vol. 10, no. 3, pp. 1147–1159, 2016. DOI: `10.1109/JSYST.2014.2308324`.

[17] F. Tahavori, E. Stack, V. Agarwal, M. Burnett, A. Ashburn, S. A. Hoseinitabatabaei, and W. Harwin, "Physical activity recognition of elderly people and people with parkinson's (pwp) during standard mobility tests using wearable sensors," in *2017 International Smart Cities Conference (ISC2)*, 2017, pp. 1–4. DOI: `10.1109/ISC2.2017.8090858`.

[18] M. Al-khafajiy, T. Baker, C. Chalmers, M. Asim, H. Kolivand, M. Fahim, and A. Waraich, "Remote health monitoring of elderly through wearable sensors," *Multimedia Tools and Applications*, vol. 78, pp. 1–26, Sep. 2019. DOI: `10.1007/s11042-018-7134-7`.

[19] M. Hamim, S. Paul, S. I. Hoque, M. N. Rahman, and I.-A. Baqee, "Iot based remote health monitoring system for patients and elderly people," in *2019 International Conference on Robotics,Electrical and Signal Processing Techniques (ICREST)*, 2019, pp. 533–538. DOI: `10.1109/ICREST.2019.8644514`.

[20] R. Aburukba, A. Sagahyroon, L. T. Kamel, A. M. Al-Shamsi, H. Surti, and E. Sajwani, "Remote monitoring framework for elderly care home centers in uae," in *2020 IEEE International Conference on E-health Networking, Application Services (HEALTHCOM)*, 2021, pp. 1–6. DOI: `10.1109/HEALTHCOM49281.2021.9398921`.

[21] J. Wang, Z. Zhang, B. Li, S. Lee, and R. S. Sherratt, "An enhanced fall detection system for elderly person monitoring using consumer home networks," *IEEE Transactions on Consumer Electronics*, vol. 60, no. 1, pp. 23–29, 2014. DOI: `10.1109/TCE.2014.6780921`.

[22] C. Chesta, L. Corcella, S. Kroll, M. Manca, J. Nuss, F. Paternò, and C. Santoro, "Enabling personalisation of remote elderly assistant applications," in *Proceedings of the 12th Biannual Conference on Italian SIGCHI Chapter*, ser. CHItaly '17, Cagliari, Italy: Association for Computing Machinery, 2017, ISBN: 9781450352376. DOI: 10.1145/3125571.3125587. [Online]. Available: https://doi.org/10.1145/3125571.3125587.

[23] A. C. Bicharra Garcia, A. S. Vivacqua, N. Sánchez-Pi, L. Martí, and J. M. Molina, "Crowd-based ambient assisted living to monitor the elderly's health outdoors," *IEEE Software*, vol. 34, no. 6, pp. 53–57, 2017. DOI: 10.1109/MS.2017.4121217.

[24] C.-T. Yen, J.-X. Liao, and Y.-K. Huang, "Human daily activity recognition performed using wearable inertial sensors combined with deep learning algorithms," *IEEE Access*, vol. 8, pp. 174 105–174 114, 2020. DOI: 10.1109/ACCESS.2020.3025938.

[25] A. Candelieri, W. Zhang, E. Messina, and F. Archetti, "Automated rehabilitation exercises assessment in wearable sensor data streams," in *2018 IEEE International Conference on Big Data (Big Data)*, 2018, pp. 5302–5304. DOI: 10.1109/BigData.2018.8621958.

[26] V. Hahanov and V. Miz, "Big data driven healthcare services and wearables," in *The Experience of Designing and Application of CAD Systems in Microelectronics*, 2015, pp. 310–312. DOI: 10.1109/CADSM.2015.7230864.

[27] S.-Y. Lee, Y.-W. Hung, Y.-T. Chang, C.-C. Lin, and G.-S. Shieh, "Risc-v cnn coprocessor for real-time epilepsy detection in wearable application," *IEEE Transactions on Biomedical Circuits and Systems*, pp. 1–1, 2021. DOI: 10.1109/TBCAS.2021.3092744.

[28] M. Ijaz, A. U. Rehman, and A. Bermak, "Prediction of heart rate and blood oxygen from physiological signals," in *2021 4th International Conference on Circuits, Systems and Simulation (ICCSS)*, 2021, pp. 244–248. DOI: 10.1109/ICCSS51193.2021.9464221.

[29] M. Agius, C. Seguna, J. Attard, K. Scicluna, and J. Scerri, "A wearable wireless sensing system for capturing human arm motion," in *2021 IEEE 12th Latin America Symposium on Circuits and System (LASCAS)*, 2021, pp. 1–4. DOI: 10.1109/LASCAS51355.2021.9459180.

[30] V. Bianchi, M. Bassoli, G. Lombardo, P. Fornacciari, M. Mordonini, and I. De Munari, "Iot wearable sensor and deep learning: An integrated approach for personalized human activity recognition in a smart home environment," *IEEE Internet of Things Journal*, vol. 6, no. 5, pp. 8553–8562, 2019. DOI: 10.1109/JIOT.2019.2920283.

[31] A. Quarto, D. Soldo, S. Gemmano, R. Dario, V. Di Lecce, C. Guaragnella, A. Cardellicchio, and A. Lombardi, "Iot and cps applications based on wearable devices. a case study: Monitoring of elderly and infirm patients," in *2017 IEEE Workshop on Environmental, Energy, and Structural Monitoring Systems (EESMS)*, 2017, pp. 1–6. DOI: 10.1109/EESMS.2017.8052698.

[32] *The future of elder care is here – and it's artificial intelligence | us news | the guardian*, https://www.theguardian.com/us-news/2021/jun/03/elder-care-artificial-intelligence-software, (Accessed on 07/12/2021).

[33] A. Kamišalić, I. Fister, M. Turkanović, and S. Karakatič, "Sensors and functionalities of non-invasive wrist-wearable devices: A review," *Sensors (Basel, Switzerland)*, vol. 18, no. 6, p. 1714, 2018. DOI: 10.3390/s18061714.

[34] *Developers - fiware*, https://www.fiware.org/developers, (Accessed on 06/28/2021).

[35] *What is a container? | app containerization | docker*, https://www.docker.com/resources/what-container, (Accessed on 06/28/2021).

[36] *Fiware-ngsiv2-2.0-2018_09_15*, http://fiware.github.io/specifications/ngsiv2/stable/, (Accessed on 06/28/2021).

[37] *The open source platform for our smart digital future - fiware*, https://www.fiware.org/, (Accessed on 06/28/2021).

[38] *Typescript: Typed javascript at any scale.* https://www.typescriptlang.org/, (Accessed on 06/28/2021).

[39] *Welcome to python.org*, https://www.python.org/, (Accessed on 06/28/2021).

[40] *Subscriptions*, https://dev.fitbit.com/build/reference/web-api/subscriptions/, (Accessed on 06/28/2021).

[41] *Using oauth 2.0*, https://dev.fitbit.com/build/reference/web-api/oauth2/, (Accessed on 06/28/2021).

[42] *Getting started with ngsi-v2 - step-by-step for ngsi-v2*, https://fiware-tutorials.readthedocs.io/en/latest/getting-started/index.html, (Accessed on 06/28/2021).

[43] *Index - fhir v4.0.1*, http://www.hl7.org/fhir/R4/, (Accessed on 06/28/2021).

[44] *Entity relationships - step-by-step for ngsi-v2*, https://fiware-tutorials.readthedocs.io/en/latest/entity-relationships/index.html, (Accessed on 06/28/2021).

[45] *Heart rate*, https://dev.fitbit.com/build/reference/web-api/heart-rate/, (Accessed on 06/28/2021).

[46] *Iso - iso 8601 — date and time format*, https://www.iso.org/iso-8601-date-and-time-format.html, (Accessed on 06/28/2021).

[47] R. Hyndman and G. Athanasopoulos, *Forecasting: Principles and Practice*, 2nd. Melbourne, Australia: OTexts, 2018.