

POLITECNICO DI TORINO

Degree course in Management Engineering
Path Innovation

Master Thesis

**Wearable HMI definition, development and validation
for collaborative environment in industrial processes in
the context of Industry 4.0**



Supervisor: Prof. Guido Albertengo
Company Tutor: Liudmila Dobriakova

Candidate:
Virginia Quartarone

April 2021

Abstract

The latest technological innovations, the changes in market conditions and the transformation of factories from digital to intelligent has led to a greater competition between companies. In fact, organizations increasingly show the need to satisfy a large number of consumers, approaching the so-called mass production. In addition, companies aim to reduce costs, improve product quality and create efficient work environments. This has produced the effect of an increasingly frequent introduction in the collaborative environments of factories of robots, that work in close contact with operators, giving them support in daily operations. From here the concept of Smart Factory develops, in which the predominant part is made up of the Cyber-Physical Systems, i.e. systems for which each physical object is associated with a digital one, thanks to which it is possible to collect, store and process data, and consequently to make the processes more efficient. In order to connect to the CPS, it's necessary to have a Human-Machine interface, which allows humans to communicate with machines and robots via the interface of a device.

The objective of the thesis is to develop a graphic interface for smartwatch that represents two of the Use-Cases of the ICOSAF project. The application is designed to support the interaction between humans and robots within a collaborative space of the intelligent factory for the execution of daily tasks.

The paper shows the various steps that have been performed for the analysis and definition, design and development of the best noninvasive HMI for the collaborative space based on the analysis of the state of the art for generic interfaces provided by university partners and methodological principles for its design; the final part regards the tools and technologies, the Android Studio platform in its Wear OS variant and, subsequently, the implementation of the project and the use of libraries for the development of the most important application features. In conclusion, the results achieved will be analyzed.

Table of contents

Introduction	1
1. ICOSAF Project.....	3
1.1. Santer Reply S.p.A.	3
1.2. Background of the project	4
1.3. Objectives	4
1.4. Technologies.....	6
1.5. Project structure: WPs	6
1.5.1. WP5 – Human Machine Interfaces.....	7
1.6. Impact of the project and expected results	8
2. Industry 4.0 and Smart Factory	9
2.1. Cyber-Physical System (CPS).....	13
2.2. Technologies of the Smart Factory.....	15
2.2.1. Internet of Things.....	15
2.2.2. Big Data	17
2.2.3. Cloud computing solutions	18
2.3. Wearable Computing.....	20
2.3.1. Smartwatches	21
2.3.2. Smart gloves	24
2.3.3. Smart glasses.....	26
2.4. Human Machine Interface (HMI).....	28
3. Tools and technologies	33
3.1. Android Studio Environment	33
3.2. Wear OS	34
3.2.1. Different models of Wear OS Smartwatches	35

3.2.2. TicWatch S2	38
4. Implementation and design of the app	40
4.1. Android Studio project	40
4.2. Server Sent Events	42
4.3. Http Connection.....	43
4.4. ICOSAF app	44
4.5. Use Case A	44
4.6. Use Case C.....	46
4.6.1. Login.....	47
4.6.2. Notifications and orders.....	48
4.6.3. Request for AGV assistance	57
4.6.4. Flow Diagrams.....	62
4.6.5. Testing of the ICOSAF app	64
4.6.6. Packaging and distribution of the app for partners	64
Conclusions	65
Appendix	67
References	69

Introduction

The job carried out during the 6-months internship at Santer Reply S.p.A. fits within the context of Smart Factory and Industry 4.0, whose aim is to create a new production system in which everything is interconnected and meet the needs of today's market, by achieving both horizontal and vertical integrations. In particular, the term "smart factory" refers to the process of digitalization typical of the manufacturing sector, that not only has impact on the way companies produce, but it also changes the nature of the organizations themselves. In fact, the production mode of manufacturing enterprises changed from digital to intelligent. This is mainly due to the introduction of robots in the collaborative spaces of intelligent factories, that has led to an ever-increasing need of automation and flexibility. In order to make humans in contact with robots for communicating, exchanging information and working together, it will be important to introduce the concept of Human Machine Interface, as the interface of a device with which the operator has to interact and, thus, it has to meet certain requirements.

The thesis work is structured in four chapters. The first one initially contains a section dedicated to the description of Santer Reply S.p.A. company. Then, the chapter is more focused on a detailed explanation of the ICOSAF project, in which Santer Reply is participating as a partner, together with other companies and universities. The chapter also contains an in-depth study of the objectives of the project, the technologies used and the expected results.

The second chapter gives an overview of the background of the concepts of Industry 4.0 and Smart Factory, with a focus on the Cyber-Physical Systems (systems for which each physical object is associated to a digital one) and on the most recent technologies related to this context. In this regard, this chapter provides an explanation of the concepts of Internet of Things, Big Data, Cloud Computing Solutions and Wearable Computing and their relationships to the Smart Factory. The section referred to the Wearable Computing describes the main

characteristics and features of the smartwatches, smart glasses and smart gloves technologies used in the factories. In addition, the last paragraph is focused on the Human Machine Interface, explaining its role and impact in the manufacturing sites.

The third chapter is related to the tools and technologies used in order to develop the ICOSAF application during the 6-months internship. In detail, it contains the explanation regarding the choice to use the Android Studio platform in its Wear OS variant, suitable for smartwatches. This chapter also includes an analysis between the different models of Android Wear smartwatches, highlighting the pros and cons of each of them it explains the reasons under the choice of a particular watch for the testing of the app.

The fourth and last chapter is entirely focused on the work carried out at Santer Reply during the internship. Initially, it explains in general how an Android Studio project is structured. Then, it contains a brief reference to the technologies used, such as the Server Sent Event protocol, and finally the most substantial part is dedicated to the development and implementation of the application, as well as, of course, the testing and distribution phases.

1. ICOSAF Project

This chapter is going to initially give a brief presentation of the company in which the internship was carried out. The other paragraphs will provide a discussion about the ICOSAF project, in which Santer Reply is involved, the objectives and the technologies used, as well as the project structure and its expected results on the companies and universities involved.

1.1. Santer Reply S.p.A.

Born on 2002, Santer Reply SpA is a Reply Group society, which provides Consulting, System Integration, Application Management and Business Process Outsourcing services. The Reply Group is specialized on the design and implementation of solutions, based on the new communication channels and digital media, in order to optimize the business processes, using innovative technologies. It contributes to develop new business models enabled by the new paradigms of Big Data, Cloud Computing, Digital Media and Internet of Things (IoTs). The Group includes more than 7000 employees and is made up of a network model of highly specialized companies and it works with the main European industrial groups in different sectors, such as Media, Industries, Services, Banks, Insurances and Public Administration.

During the years, Santer Reply has developed a relevant experience in industrial research in several technological areas, such as:

- Internet of Things
- Cloud Services and Platforms
- Cloud architectures
- Proximity and localization technologies

Moreover, the company also boasts an experience in hardware and software platform development for industrial customers and has a large capacity to

transform innovation into real products and services immediately available in the market.

1.2. Background of the project

The 30-months European project named ICOSAF stands for *Integrated Collaborative Systems for Smart Factory*, to which Santer Reply SpA takes part as a technology consulting firm, together with other entities, such as Italian Universities (Università di Napoli – Federico II, Università della Basilicata, etc.) and companies (Centro Ricerche Fiat SCpA, Adler Plastic SpA, etc.). In particular, one of Santer Reply's business units (BUs), Concept Reply, is in charge of the development strategies of the project. This BU is focused on the Internet of Things applied to the industry and services sectors, and it accelerates its commercial offering and the quality of innovation provided to customers in the field of Industry 4.0.

Today's manufacturing industry requires an ever-increasing need for flexibility, because it has to satisfy products' personalization and its short lifetime. As a consequence, requirements such as flexibility, scalability, modularity and re-configurability of the production sites are fundamental. Moreover, from a social point of view, it's fundamental to support operators with ergonomic technologies that reduce the negative impact of fatigue on their health. This system allows an efficient utilization of human intelligence, but at the same time it's possible to increase quality and productivity.

1.3. Objectives

The aim of this project is the development of the technologies and systems for a collaborative factory, taking into account the principles of Industry 4.0 (interconnected automation) and industry 5.0 (humanization and reuse of resources) and facilitating the integration of the operator in the factory, in order to establish a cooperative working environment between operators and machines.

While industry 4.0 has been the starting point for the advent of robots, the crucial part of industry 5.0 is constituted by cobots, that are robots but with the ability to work in a collaborative environment together with humans and other cobots, guaranteeing security and efficiency. Their importance is constituted by the fact that they are able to help humans for low added value tasks, leaving the operator to exploit his intelligence and flexibility for more complex operations. An example of the collaborative robots used in the ICOSAF projects are Franka, Kuka (that are robotic arms) and AGVs (Automated Guided Vehicles). The vision of a cooperative working space includes mobile and fixed robotic systems, quality and machinery active monitoring systems for operator's assistance and AGVs that interact with the operators and the environment, in order to achieve a higher factory flexibility level, maintaining the productivity requirements of the current factory. In addition, the integration of these systems into the smart factory leads to substantial improvements in productivity, quality, flexibility and ergonomics.

In detail, the project aims to:

- develop new models for collaborative factory
- enhance the effectiveness and efficiency in the integration and use of robotic collaboration
- develop an active and smart logistics with AGVs that monitor, transport or perform autonomous collaborative actions (autonomous kitting and sequencing, autonomous delivery up to the product line, cooperative actions)
- define quality control and device analysis systems designed to optimize the product/process quality and the predictive maintenance
- develop human-machine interfaces optimized for the cooperative factory
- develop methodologies for the digitalization and profitable integration of technologies developed into the shop floor of small, medium or large companies.

1.4. Technologies

The main technologies involved in the project are *Cooperative Robotics*, which assist the operators during manual operations, *Intelligent, robotic and autonomous AGVs* capable of performing coordinated operations, *Advanced Systems* to support maintenance and quality, *Advanced human-machine interface (HMI)* systems to optimize interaction and cooperation between man and machine and *Simulation models and augmented /virtual reality systems* that enable the improvement and validation of functionality and processes.

The reason why it was chosen to integrate the Collaborative Robots (also named as *cobots*) in the ICOSAF project is that these robots, differently from the traditional industrial robots, can work closely with a human operator, interacting with him during the whole execution of an operation. In such a scenario, it's clear that the protective barriers that separate the work space of man from the robot's one become useless. This involves a great saving of space but also increases productivity, as the robot can continue to work even when the operator is present in his work area. On one side, the Human-Robot Interaction (HRI) offers to the industries new and more flexible alternatives with respect to the traditional lines of production, on the other side, it also involves some problems, concerning the human security, because now the human and the robot are starting to work together and they need to comprehend each other in order to respect the security requirements and the system performances.

1.5. Project structure: WPs

The project is organized according to a structure (see Figure 1), that defines the different interactions among the Work Packages (that are realization goals) and the timeline in which they have to be performed and completed.

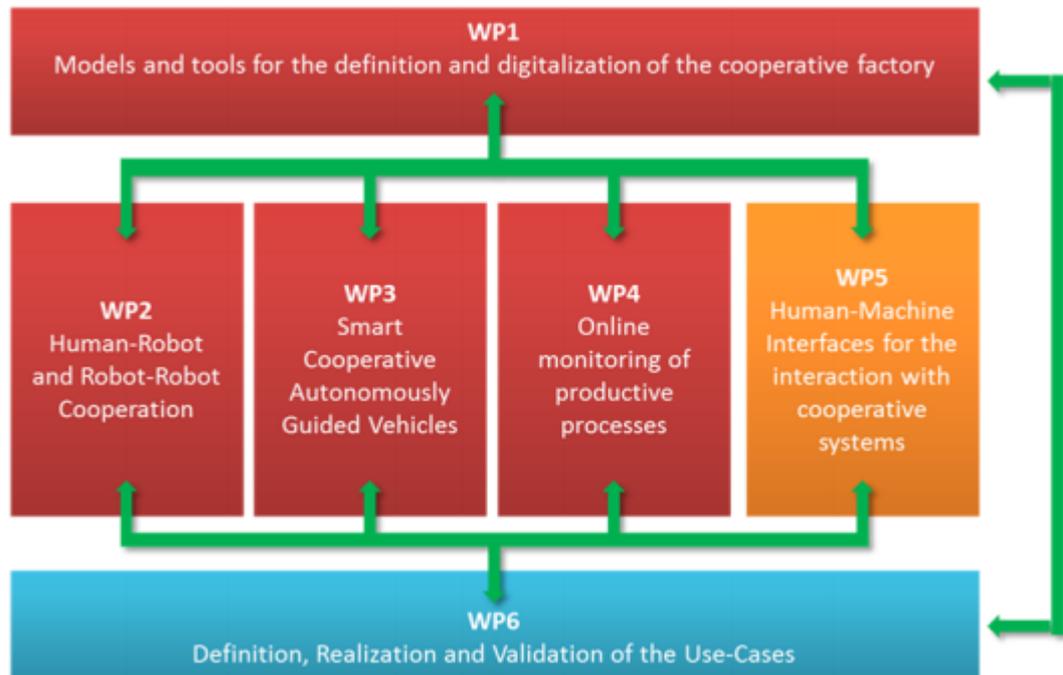


Figure 1: Structure and interactions between the Work Packages of the project

The Work Packages structure reflects the approach of the project of defining a cooperative factory, by providing a realization of all the necessary models, prototypes and technologies.

The first Work Package (WP1) follows the project in its whole duration, from the definition of the functions of the cooperative factory, to the implementation of simulation models, its digitalization and validation. The next four Work Packages (WP2, WP3, WP4 and WP5) are in charge of supporting the part related to the definition of devices and/or technologies to be applied in the normal operation of the factory. Finally, WP6 performs Use-Case identification cross activities, prototype and demonstration, testing and validation. The purpose of this Work Package is to set functional tests for prototypes in order to verify the proper functioning and to assure the integration with the HW and SW.

1.5.1. WP5 – Human Machine Interfaces

In particular, the WP5 is in charge of defining, testing, validating and then developing functional Human Machine Interfaces (HMI) for the interaction of

operators with the AGVs (Automated Guided Vehicles) in the logistics, paying attention to provide a simple, effective and safe interaction. Moreover, in this context, HMIs would be fundamental to allow a real-time visualization of logistics data. In its ending, this part of the project plans to develop graphical mock-ups for laboratory tests that emulate logistics applications with AGVs and cooperative manipulation systems.

1.6. Impact of the project and expected results

The expected impact of ICOSAF project on the companies and Universities involved will lead to improvements in terms of productivity, innovative know-how, product quality and working environment quality. In this way, it will increase the capacity of these firms to attract young and motivated people. Moreover, the availability and the knowledge of the new technologies will lead to the firms involved in the project to an increase in the level of visibility and attractiveness at local, national and international level.

In particular, the participation to ICOSAF project enables Santer Reply to:

- Increase the dimensions of the effort currently invested in innovation
- Accelerate prototyping and experimentation of software solutions for industry to be rapidly converted in market solutions
- Increase employment in the offices involved in project activities.

2. Industry 4.0 and Smart Factory

The following chapter provides an overview of the concept of Industry 4.0 and an in-depth understanding of how the Smart Factory has developed and the way it works. After that, it's fundamental to introduce the concept of Cyber Physical Systems and the main technologies used inside this “factory-of-things”, i.e. IoT, Big Data, Cloud Computing and Wearable Computing. Regarding to this last argument, the main features of smart glasses, smart gloves and smartwatches will be presented. For what concerns the smartwatches, a comparative analysis between the main manufacturers is provided, highlighting for each of them the pros and cons. The last paragraph of this chapter contains a study of the Human Machine Interfaces, their role, the main principles to respect in order to have a good HMI design and the impact and benefits for the companies.

The rapid development of electric and electronic technology, information technology and advanced manufacturing technology allowed the production mode of manufacturing enterprises to change from digital to intelligent (Wan et al., 2007). As a consequence, the advantages of traditional manufacturing technology have been gradually diminished in favor of the new challenges linked to the concept of Industry 4.0, whose aim is to integrate the production with the latest information and communication technologies. Industry 4.0 represents a smart manufacturing networking in which machines and products interact with each other without the need of a human control. For this reason, it requires a *horizontal integration* through the value networks, in order to facilitate inter-corporation collaboration, *vertical integration* of the hierarchical subsystems inside a factory and *end-to-end engineering integration* for all the entire value chain in order to support product customization.

The Figure 2 below describes the relationships between these three kinds of integration.

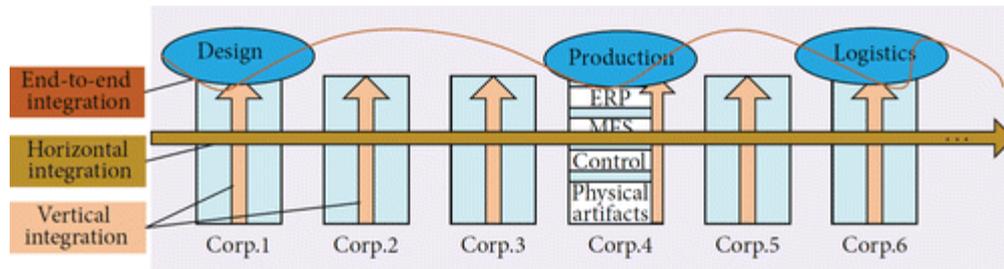


Figure 2: Relationships between vertical, horizontal and end-to-end integration

The manufacturing process have to be capable to rapidly adjust the product type and the production capacity in real time in order to be able to process small batches and multiple varieties (Chen et al., 2017).

Moreover, the industries have to become more reactive and fast in order to follow the changes at the market level and they have to be capable to adapt very quickly to the new customer requests and needs. In this context, the level of innovation and the technology life cycle is diminished, in favor of custom products at the cost of large-scale production. As a consequence, the big challenge for the companies is to adapt their process path to the new requirements and increase their productivity, also by improving the use of resources, and reduce the time-to-market to deliver products on the market. The new concept for the factories foresees just the production of the products effectively requested by consumers and an integration of all the production phases.

For this reason, such companies have to become “smart”, because they substantially have to adapt their production to the changes on the market.

The Smart Factory is at the heart of Industry 4.0. Basically, the Smart factory consists of networking machines and systems that can communicate with each other in an intelligent way and the work steps can be automatically coordinated with each other. In general, the “Smart Factory” concept is aligned with three other concepts: smart production, smart services and smart machines, as Figure 3 below shows.



Figure 3: The three concepts behind the “Smart factory” term

Smart production is referred to the fact that there exists an efficient collaboration between operators, machines and tools inside the factory. The services are considered “smart” because it is possible to monitor the production and to provide maintenance in real time, moreover the operators have the possibility to take advantage of the online services and of the virtual machines and tools (such as the smart glasses for the augmented reality). Even the machines are smart because systems are becoming more and more performing, thanks for example to the use of MRP, a technique to plan the production and purchase orders and thanks to the MES, an information system that has the function of manage and control the production of a factory.

The smart factory framework consists of four tangible resources, that is possible to see in Figure 4:

- Physical resources
- Industrial network
- Cloud
- Control terminal

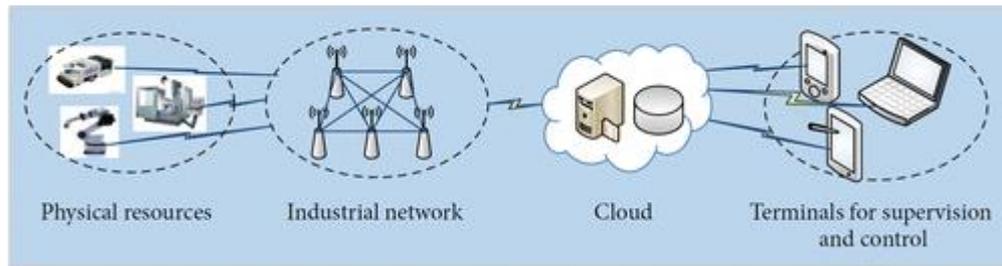


Figure 4: Example of the smart factory framework

The physical resources are intended to be objects (such as smart products, smart machines, smart conveyors, etc.) which can communicate with each other thanks to the industrial network. Instead, the cloud contains various information systems that can collect a massive amount of data from the physical resources and that can communicate and interact with people through the control terminals.

For a factory to become “smart”, it must increase flexibility, efficiency, security and speed, through the help of smart automation. In particular, the automation is useful to accelerate and optimize the production processes, by making the production process functions to be executed autonomously by artificial systems, such as machines, that have been previously programmed to do so. Therefore, this intelligent factory represents an engineering system that mainly consists of three aspects: interconnection, collaboration and execution.

The main goal of a Smart Factory is to create a networked, efficient and self-organized production environment and to optimize the production layouts and the connectivity between humans, software and machines, in order to personalize the products and to make the processes more flexible. Moreover, the Smart Factory aims at improving performance, quality, controllability, management and transparency of manufacturing processes.

Nevertheless, the human presence is a “must have” in this scenario. In fact, although people’s field of activity is progressively changing, due to the actual challenges, they must continue to be at the center of attention, because they still have to carry out controlling activities. In particular, within the context of the smart factory, it’s possible to assist for the first time to an optimal integration of the

human abilities with the ones of the *smart* resources, with which he communicates and collaborates during all the production phases.

In order to execute these tasks, people have to be connected to the Cyber-Physical System via a multimodal interface, also known as Human-Machine Interface (HMI).

2.1. Cyber-Physical System (CPS)

The Cyber-Physical System (CPS) forms the basis of the Smart Factory and of the Fourth Industrial Revolution and relies on the newest and foreseeable developments of Computer Science (CS), information and communication technologies (ICT) and manufacturing science and technology (MST). The CPS consists of hardware tools able to connect to the network and to interface with the latest technologies, because its aim is to enhance productivity and product quality.



Figure 5: Example of a Cyber-Physical system

This system includes the concept of “Digital Twin”, so it requires that physical objects are represented in a digital way, so that they can be integrated with tools

that are capable to do calculus, to memorize and process data and to be linked in the network together. Figure 5 is a clear example of a CPS: a real automobile and its virtual representation on a pc. The Cyber-Physical systems are intended to be one of the most innovative technologies in the context of Industry 4.0 because they give a great contribution to the digitalization of manufacturing in three aspects: the *smart product*, the *smart manufacturing* and the changes in the *business model* of the firms. In this sense, a Cyber-Physical System is a set of different technologies that generate an autonomous and intelligent system, that is capable to facilitate the integration between different objects, even distant from each other. For this reason, this kind of system is mainly in charge of three actions: it generates and acquires data, then it provides to aggregate these data and, at the end, it gives support at the decisional step.

The term “physical” is referred to the object and to the way it is perceived by our five senses, while the term “cyber” refers to the digital image (the Digital Twin) of that specific object, that gives a more detailed set of information regarding that object. As a consequence, thanks to the information available at the digital level, the single component becomes able to take decisions in an autonomous way and to communicate it to the other physical components. The not-centered intelligence and the double vision (a real and a digital one) allows the CPS to assess the situations in an independent way, to elaborate in real time the different possible choices, to support important decisions and to assure that the other cyber-physical systems execute the correct actions and take the correct decisions.

These systems contribute to give huge improvements for the factories and brings a lot of benefits. First of all, thanks to the “Digital Twin”, CPS are able to offer new business possibilities, because it allows the firms to be closer to the new requests and needs of customers. Secondly, these systems allow factories to generate and work with data and information on a real-time basis and this contributes to a higher level of flexibility and capability to react to the market changes. Moreover, there are also advantages linked to the digitalization of the product. In fact, in this context, the product is transformed in an intelligent object, that is able to communicate inside and outside of the factory and is able to share

and collect information at different levels. This kind of visibility brings benefits both to the factory, because it can understand if there exist some problems in the use of the products, but it has some advantages also from the user point of view. The reason is that the goal for the firms will be to create and manufacture ad-hoc products for the customers, paying a lot of attention to satisfy their requirements and reducing the products that do not have demand on the market. This last concept opens another point in favor of the CPS systems: a zero-waste supply chain.

While talking about the Cyber-Physical System, it's also important to mention the machine-to-machine communication (M2M communication). In the context of a Smart Factory, not only a human being can communicate with the machine, but even the machines can independently come into contact with each other or with other factory tools. In this way it's possible to record the data that come from machines and immediately obtain a real-time processing of information, known as Real Time Enterprise (RTE), that is a key element in a Smart Factory.

2.2. Technologies of the Smart Factory

Some enabling factors of industry 4.0, and consequently of the smart factory, are the emerging information technologies such as the Internet of Things, Augmented Reality, Big Data and Cloud computing, as well as of course the Artificial Intelligence technologies (AI). These technologies are fundamental for the growth and the sustainability of the factory because they are integrated with industrial automation, business and trade, so, as a consequence, they are able to make the industry to achieve huge improvements. In fact, this kind of deep and strong integration makes the production to operate in a more flexible, efficient and green way, but at the same time paying a lot of attention to maintain a high quality and low costs.

2.2.1. Internet of Things

The concept of *Internet of Things* (IoT) is the most essential and fundamental technology that enables the CPS and, in the field of manufacturing, it is referred to

the fact that physical objects, people and systems (things) are connected via the Internet and are able to interact with each other inside a network, where each of them has a proper and active role inside the system, in order to build services useful for manufacturing (N. Shariat Zadeh et al, 2016). In addition, those “things” are characterized by the fact that they are identifiable in a uniquely way and they receive a virtual representation, they become “intelligent” and capable to generate data and execute some actions. The principle of Internet of Things is very much aligned to the one of Smart Factory, as this last one is often defined as factory-of-things, since all the things inside the factory are connected with each other. The major application areas where IoT is used are the health care, the logistic, the agriculture and the robotics.

The IoT architecture consists of four major layers: the sensing layer, network layer, service layer and interface layer. The sensing layer makes use of various techniques (RFID, sensors, actuators, Bluetooth, Wi-Fi and etc.) in order to obtain information from the physical environment. These data will be transferred through the WLAN, Internet, Intranet, etc. to be connected with the database in the network layer. After the data clearing process, it’s possible to obtain knowledge by data mining in the service layer and at the end the result is presented in the interface layer.

Within the context of smart manufacturing, the data from sensors and machines are communicated to the Cloud by IoT connectivity solutions deployed at the factory level. Then, these data are analyzed and combined with contextual information and, at the end, shared with stakeholders. In this way, the Internet of Things technology enables this flow of data and provides the possibility to monitor and manage processes remotely, as well as modify production plans, when needed, in real-time. In addition, IoT technology has an important impact for the manufacturing outcomes. In fact, it contributes to reduce wastes, to speed the production and to improve the efficiency and the quality of goods produced. The reason is that connected smart products can feed information back to the factory, so that if there exist some quality issues in the products, these can be detected and solved during the manufacturing stage.

2.2.2. Big Data

Between all the different implementations available for the Smart Factory, the Manufacturing Execution System (MES) is one of the main tools for an efficient management and for an improvement of the production processes. In particular, the MES includes the *data analysis* function, that allows the diagnosis, prediction and optimization of the production through real-time analysis of the production site data, so that the decision-making process can rely on these data. Moreover, by revealing a causal relationship on a failure or on a problem, it is possible to achieve a performance improvement. The collection, processing and use of data are the core key of the *data analytics* (SJ. Shin et al., 2016). However, manufacturing data sometimes make it difficult to apply *data analytics* because of different reasons. First of all, manufacturing products composed of Man-Machine-Material create, use and store a large amount of data (Volume). Second, manufacturing data has a variety of time, sources by which data is generated and formats (Variety). Third, it should be made of fast data processing, delivering and response for real-time decision-making at the manufacturing site (Velocity). White (2015) has added the fourth dimension, Veracity, to highlight the importance of data reliability and the level of trust in a data source. Since it is not easy to satisfy the required performance of data processing with the existing relational database system, it is essential to include a distributed database system and various data processing technologies to solve this problem, so-called *big data infrastructure*. Therefore, data analytics fused with big data infrastructure leads to Big Data Analytics (BDA) that is one of the essential technologies for the Smart Factory, as it enables timely and accurate decision-making for an optimal production planning and control. Nowadays, the current information systems are being asked to take the responsibility of storing an increasingly large volume of data (Big Data), as well as supporting the real-time processing of this 'Big Data', by using advanced analytics. The progressive increase in the number of tools able to record measurements from physical environments and processes will lead to an exponential growth in the data production and so, as a consequence, in the need to accurately store these data (P. O'donovan et al., 2015).

Usually, factories must perform real-time control, in order to eventually detect some abnormal situations as soon as possible and to take quick decisions. In order to do that, organizations must be able to work with big data technologies to meet the demands of smart manufacturing. Therefore, it is necessary to develop a data infrastructure capable of handling such large-capacity of data in a short time. On the other hand, manufacturers want to easily access and use manufacturing data, regardless of where the manufacturing data is located or generated. In addition, manufacturers want to ensure data quality, so that each manufacturing data is correct and that it has been collected from trusted data sources. Therefore, it is necessary to develop a data infrastructure that assures data transparency, accessibility and reliability.

Big Data have a key role in the context of production to help companies to have a clearer picture of their manufacturing processes while automating data collection and data analysis. In addition to improvements in the manufacturing processes, Big Data Analytics can also be used to improve the safety of the work environment. In fact, the Key Performance Indicators (KPIs) collected for health and safety can reduce the risks related to injuries and incidents that can occur on the job. Big data Analytics is also essential because it helps manufacturing companies to make more informed business decisions, minimize production risks and improve their manufacturing processes by increasing efficiency, reducing waste and improving product quality.

The big challenge today has shifted from the difficulty in how to collect a sufficient quantity of data, to how to best use a vast amount of data available in order to take better business decisions. In particular, organizations have to transform the data into tangible ways that will improve efficiency, so that manufacturers could gain a competitive advantage.

2.2.3. Cloud computing solutions

The progressive adoption of Machine Learning methods improves the flow of materials through the manufacturing process and reduces to a minimum the downtime of production equipment. In this scenario, the *Cloud* becomes an enabler

and an essential factor that helps enterprises to accelerate the digital transformation process quickly and allows to achieve more flexibility, from the ability to respond to market demands, to cost control and an efficient management.

With the emerging trends of Smart Factory, Internet of Things, digital supply chains, vertical and horizontal integration, etc., the *Cloud Computing (CC)* has become an essential part of the production world, as it represents a sort of prerequisite for all these technologies.

Cloud computing is a relatively recent concept which combines technologies for resource management and provisioning with the ideas of mass deployment, elasticity and ease of use.

The idea behind the concept of “cloud” is in line with the one of “mobile”: in fact, in this way, information can be accessed by the users from any place and from any device. The major consequence falls on the production lines, in the sense that operators will be equipped with a smartphone, a tablet or even a smartwatch, with whom they will be able to communicate, send and receive necessary information, requests and notifications relative to tasks or to problems, so that they can intervene immediately. Always more often, the *Cloud* is integrated with the Artificial Intelligence solutions, that are able to suggest to the operators the possible solutions to adopt in every circumstance (D. Carbone, 2018).

To the enterprises, it is an interesting concept for several aspects: from internal applications to the possibility of sharing resources with other organizations or supplying their own resources as a service to others. One of the reasons why enterprises adopt the *CC* technologies is that it can allow them to have an easier management of their resources, a dynamic infrastructure and a support for various platforms and, in addition, it is essential to access quickly to every kind of information and to decrease costs. With *Cloud Computing*, companies will always be able to choose the best technological innovations, while respecting budget constraints and they will be able to increase scalability and agility. This technology allows to control all the processes across different locations and plants, even being far from those places, and, in this way it allows to eventually detect some problems and provide a resolution or optimization.

An organization equipped with *Cloud* solutions has an added value as suppliers and distributors become active partners for the reason that they are involved in all the share of information regarding the machines, their maintenance and their life cycle.

2.3. Wearable Computing

Before introducing the wearable devices and their features, it is important to briefly explain the context within which this new technology fits. Nowadays, an ever-increasing number of companies in the ICT (*Information and Communication Technology*) sector decides to invest resources in the Internet of Things and, in particular, in one of its emerging branches, which is the one of *Wearable Computing*, that is about the wearable technologies, usually of small dimensions and usable by the human being.

The Wearable Computing is the enabler of the interaction between man and machine because it allows the user to have an immediate access to the information he needs because he receives it directly on the device he is wearing. The user can interact with these smart objects, that will behave in a different way depending on the situation and the context in which they are involved, assuring to the operator to have a unique user experience because it is very similar to having a personal assistant. As a consequence of the Wearable Computing, the Wearable Technology (WT) has developed, in order to help and assist the human being in all his everyday tasks. One of the main characteristics of the WT is a hands-free function, that enables people and employees to have access to the data while performing their daily routine activities and job tasks. In addition, according to what the literature reports, many authors have written the characteristics that wearable technologies must have, such as to be integrated, seamless, comfortable, portable, multi-functional, reliable and practical. Moreover, there are several key attributes which play a significant role in the design of a WT, like the size and the dimension of the device, the power source, the heat, weight, durability and usability. On the other hand, it's important that the wearable devices have these main functions: *interface*

to transfer data between the wearer and the device, *communication* to transfer information via Bluetooth, wireless systems, etc., *data management* to store and process the data, *energy management* referred to the device's battery life and *integrated circuits* (Mesut Cicek, 2015). Over time, Wearable Technologies demonstrated to be really helpful for the human being and to have positively revolutionized many contexts, so that now they are widely spread in the healthcare (especially during COVID-19), fitness and in the production and logistic sector. Wearables combine the features of some of the most popular technologies and, thanks to their intrinsic properties of adapting to the human body, they can track useful data in a non-obtrusive way for the humans. Moreover, secretarial services are soon going to disappear, as wearable devices will be the worker's virtual assistant, capable to stay with the wearer 24/7, remind the necessary information, etc. and will also learn from him cumulatively and never forget any type of information.

2.3.1. Smartwatches



Figure 6: Picture of a common smartwatch

Even if the smartwatches seem to be one of the most recent and widespread technologies, in reality the initial prototypes date back to the 80s. However, just

during the recent years, the intelligent watches have faced an ever increasing diffusion on the market, thanks to its versatility and flexibility: it's used in the fitness, wellness and in the industry sector. Apple with *iWatch* and Google with *Wear OS* are the main producers of software and hardware tools related to the world of intelligent watches. *Wear OS* is nothing more than a version of Google's Android operating system but working for smartwatches. Wear OS, if connected with an Android phone or tablet, is able to integrate the features of the Google Assistant, the notifications from the device and the use of the apps on the smartwatch. Smartwatches substantially contain almost all the characteristics and features of a smartphone, but, undoubtedly, they allow for a better user experience, thanks to the fact that users can have access to the information that they need more immediately, thanks to the fact that such a device is wearable.

Typically, Smartwatches are equipped with a series of sensors, thanks to which the device is able to collect data. Usually, such sensors are divided into three categories:

1. Motion sensors: accelerometer, gyroscope, geomagnetic sensor, atmospheric pressure sensor.
2. Biosensors: includes sensors for detecting parameters such as glucose, blood pressure, ECG, EMG, body temperature.
3. Environmental sensors: temperature, humidity, gas, PH, ultraviolet, pressure.

Of course, these are just some of the sensors integrated into the technology of the smartwatches, which also include sensors for Wi-Fi and Bluetooth. It is clear, however, that the greater the number of sensors integrated into the device, the greater the number of contexts and situations in which it will be possible to use the smartwatch.

Most of the data transfers between a smartwatch and other devices commonly take place via Bluetooth, which is the most widely used wireless communication protocol, although more space is recently being given to the communication via Wi-Fi. This technology guarantees more flexibility in the use of the device, which is therefore able to manage the collection and exchange of data only if connected

to a Wi-Fi network. In this way the device is completely independent from the smartphone or tablet.

In general, among the most common features of the Smartwatches, there is the possibility of making and receiving calls, making mobile payments (with Google Pay and Samsung Pay for example), communicating and/or making requests to a virtual assistant, monitoring the frequency heart rate and daily steps taken (most Smartwatches on the market have a good basic Health platform, such as Google Fit, Samsung Health and Apple's Health app). As previously mentioned, Apple and Google are the largest Smartwatch manufacturers on the market, followed by Samsung and Mediatek. The Table 1 below shows the pros and cons of the intelligent watches for each of these companies.

Table 1: Main smartwatch producers on the market and their pros and cons

MANUFACTURING COMPANY	ADVANTAGES	DISADVANTAGES
APPLE - iWatch	<ul style="list-style-type: none"> • It is a more profitable sector than the competing platforms 	<ul style="list-style-type: none"> • It is uniquely addressed to users who already have an Apple device • It exists only one shape for the watch (the rectangular one) • It is expensive • The battery lasts for 24-36 hours
GOOGLE – WEAR OS (Android Wear)	<ul style="list-style-type: none"> • The Smartwatches Wear OS work also with the iOS system (iPhone and iPad) • There's a much larger user base rather than Apple • This system is compatible with different models of smartphones and tablet 	<ul style="list-style-type: none"> • For the moment, there's only a unique level of hardware Android Wear • The battery lasts for 24-36 hours

	<ul style="list-style-type: none"> • <i>Standalone Apps</i> • Two available shapes (rectangular and round) 	
SAMSUNG (Tizen OS is the developed operative system)	<ul style="list-style-type: none"> • It is equipped with the common features also found in other smartwatches (for ex. Heart rate monitoring, connection with GPS, ability to make/receive calls, etc.) 	<ul style="list-style-type: none"> • Inability to access resources that are external to the Samsung devices and to all the additional apps, which will be inevitably be downloaded from the Tizen store. • Only compatible with the Samsung devices.
MEDIATEK (alternative for the smartwatches)	<ul style="list-style-type: none"> • The battery lasts for 5-7 days • The devices are cheap • Great long-term growth potential 	<ul style="list-style-type: none"> • Limited functionalities • No user base currently on the market

2.3.2. Smart gloves



Figure 7: Example of smart gloves

The smart gloves are a particular type of gloves, made with a novel stretchable substrate material and embedded with flex sensors that are able to track finger

orientation, and that determine the degree of freedom of the gloves (DOF), and an Inertial Measurement Unit (IMU) in order to track hand movements in a 3-dimensional space. This combination between hardware and software allows to acquire the hand and fingers movements in real time and very accurately, even when the user holds an object. In fact, the values obtained from flex sensors and the IMU module are used to identify the position of the user's hands in the space, as the circuitry embedded inside the gloves communicates to an end device (B. O'Flynn et al, 2015). In order to detect the finger orientation, flex sensors are embedded behind the finger, which can execute four type of gestures (K. Bhaskaran et al., 2016). The current prototype of the smart glove system consists of 3 main components: a set of sensors, a data sampling unit (DSU) and a visualization and a computer software.

The objective of the smart gloves' technology is to measure the range of hand joint movements, in real time and in a quantitative manner. This includes taking accurate measurements of flexion, extension and adduction of the fingers.

Based on a study conducted by A. Akpa (2019), smart gloves are even widely used for fitness purposes in order to track the activities, for example by analyzing the time series of the pressure distribution in the hand palms observed during a fitness session. Apart from fitness purposes, the smart gloves technology demonstrates to be very useful in the medical and health sector. For example, one of its applications is for rehabilitating a hand after an accident or a health disease directly from home, without the need to go to a physiotherapist. The smart glove makes the user's hand to execute specific movements in order to progressively reacquire flexibility of the movement. Another application of this technology took place in Kenia, where a researcher invented a particular type of smart gloves that can translate sign languages into real speech, by using gesture recognition and sensors embedded into the gloves.

Bringing these gloves into mass production is for sure a challenge, but this technology constitutes, together with the smart glasses, the base for augmented reality (AR) and virtual reality (VR) experiences of the future, thanks to the

objectives that it has achieved: real-time monitoring, autonomous operation and ability to work in multiple environments.

2.3.3. Smart glasses



Figure 8: Example of smart glasses and augmented reality

The Smart glasses technology dates back to the 90's with the first and historically significant prototype, that was mainly a head-worn computer, but then this technology spread more during 2014 when Google introduced *Google Glass*, that is the first example of a real smart glass on the market. These glasses contain multiple sensors (camera, microphone, GPS, accelerometer, light sensor, etc.) and are equipped with a see-through optical display, positioned in the eye-line of the users, who can view contemporarily both the real-world environment both the virtual contents shown in the display. For this reason, today's smart glasses are regarded as the beginning of the principle of *augmented reality* on mobile devices. In this way, these glasses can superimpose the virtual content, such as text and images, into the user's field of view. While the major input for the other wearable devices is the touch screen, the smart glass wearer can perform inputs through various actions, such as head movements, hands gestures, voice inputs, etc. and the sensors embedded into the glasses will identify the actions of the user. These devices are compatible with the smartphones and can be connected with them via Bluetooth. Smart glasses have mainly been designed to support micro-interactions between the wearer and the glasses, such as map navigation, photo or video

capturing, receiving and reading a notification/message and even booking a taxi. Over the years, some additional versions of this technology have been developed. For example, a particular pair of smart glasses are also sunglasses, that protects the human eyes from UV radiations. Another ergonomic feature is the possibility to listen to the music directly from the glasses, without the need of earphones, but using the bone conduction headset, that is a particular tool that allows the sound to be propagated without passing through the eardrum but using the bones of the face. Inside the factories, the operator that wears the smart glasses can observe the reality that surrounds him, enriched with additional information about what he sees, and he has the possibility to share his view with a remote assistant.

In a paper named "*Mime: Compact low power 3D gesture sensing for interaction with head mounted displays*", A. Colaço et al. show the advantages of smart glasses compared with the most common smart devices. They sustain that a touch screen interface does not fully take advantage of human dexterity, it requires the user to touch a small screen repetitively and constantly on the device, thus this will occlude the user's sight of the display. Instead, smart glasses offer a better input approaches, as they make the interaction experience more intuitive and efficient, leaving the user's hands free and bringing the smart glasses to expand their limited usage of micro-interactions to daily usage.

However, this form of interaction is still problematic because the virtual content on the optical display is not touchable and, as a consequence, the operator cannot have a direct manipulation of this object. Moreover, according to L. Lee (2018), smart glasses are considered a rudimentary product because they provide a very small user interface, weak processors, limited computational power and, of course, a short battery life, so they result to be less flexible than the other smart devices (as the smartphones). As a consequence, even though the future of smart glasses is very promising, it's not clear if this technology will be adopted by users for daily usage in the same way as today's smartphones and smartwatches or it will just serve for industry operations and for specialized tasks.

2.4. Human Machine Interface (HMI)

A Human Machine Interface is the interface of a device or of a software application used by an operator in order to communicate with a machine, a robot, a system or a device. The most common HMIs involved in the industrial field are screens, touchscreens and keyboards, and are mostly used to control and automatize machineries and the lines of production.

Back in the '50s, the main kind of interaction between humans and machines was with the so-called *batch processing*, that required the user to specify all the details and sequences of an operation/task using a punch card, that was inserted into the machine, that then delivered the results. However, this technique brought with itself a high probability of error, that is why it was not considered as an efficient HMI. After the batch processing, during the '60s, the *command line interfaces* have been developed. This is a more interactive way for users to engage with machines and eventually direct commands to the machine and it is done by entering successive lines of text. The next phase of the human-machine engagement is made up by the *Graphical User Interfaces* (GUIs). These interfaces allow end users to engage with machines using graphical elements, such as windows, buttons and icons, and, successively, even the keyboards and the mouse. With the increase in the technology performance and in the use of computers, there was a need for more sophisticated levels of human-machine engagement. Thanks to that, touchscreens have been developed.

The basic HMI allows the operator in a plant to check typical parameters referred to the machines and their status. However, many modern industrial HMIs allows users to receive alerts and notifications about eventual problems in the machines or in the production plant directly on their smart devices (in fact, this is the case of the ICOSAF app for the operators). In addition, these advanced interfaces allow for remote control of machines and operations on multiple sites and guarantee a better efficiency and a very good time management of operators, that in this case do not need to go physically to a plant to detect problems.

The role of HMI in the manufacturing plant and sites is rapidly evolving as new technologies are continuously being integrated and as robots have started to gain more and more importance, especially in the collaborative environments.

By the way, after having introduced the importance and the roles of the collaborative robots inside an industrial plant in Chapter 1 (paragraph 1.3), it is fundamental to mention that the integration of robots have brought a substantial increase of the productivity and a decrease in the production costs. In fact, parallel to the technological progress and to the advent of Industry 4.0 and of the smart factory, robots became always more independent from the human operator and they acquired a deep sense of autonomy inside a work environment.

A common human machine interface consists of three elements:

- Operating elements. These elements allow to exchange and deliver information from the operator to the machine through, for example, a button.
- Displays. Displays are used to illustrate and transfer to the user information about the machine.
- Inner structure: it consists of hardware (e.g. electronic circuits) and software (e.g. computer programs).

Of course, an HMI should be based on natural motions that allow the user to interact with the machine and use it independently and in an easy, intuitive and smart way. For this reason, the usability is one of the key important features related to the human machine interfaces and, according to the International Engineering Consortium (IEC, 2007), the concept of usability is linked to its level of:

- effectiveness, such as the orientation towards the objectives
- efficiency, so the easiness for the operator to learn, remember and to minimize the errors during its usage
- satisfaction, that regards the compatibility with user's needs and the level of pleasure that derives after its usage.

In addition, another key success factor for a HMI is that its design must be user-centered, in order to meet operator's requirements and needs and in order to be easy for him to be used.

There are some principles for a good HMI design:

1. Principle of least astonishment

This principle can be applicable on the user interface and on the software design. This principle aims to exploit the pre-existing knowledge of users to reduce the learning curve to the minimum, for example by designing interfaces that are based on functionally similar programs so that users develop familiarity by using them.

2. Simplicity

The Principle of Simplicity aims to design a user interface that should be simple to learn and to use. In details, simplicity is based on the observation of the objectives and on the user experience, in this way the complexity of the system is reduced.

3. Human Memory Limitations

The human capacity of storing information, even if flexible, is very limited. For this reason, the activities designed through the definition of user interfaces must allow to organize short sequences of activities that are able to minimize the loads of working memory, by limiting the length of the activities sequences and the quantity of information to store. In particular, according to the limitations of the human memory capacity, the user interfaces should be integrated with systems able to:

- Provide suggestions and/or help to the user in order for him to understand in which phase of an operation/activity he is.
- Give reminders and/or alerts
- Provide feedbacks on what has happened, allowing users to recognize and remember the information more easily.

4. Cognitive Directness

It is based on how well the usage of a product corresponds to the cognitive capacities of the users. In this step, it is fundamental the knowledge about human perception, mental processing and memory activities.

5. Feedback

The feedback on the user interface mainly consists on the fact that the system reacts to the action of the user in the most appropriate location of the system.

6. System Messages

These are messages created by the system in order to be understood by the user. For example, a system can't show to the user "Execution error 159" but it has to show "There was a problem when copying your files". System messages that are not properly clear to the users but that are quite ambiguous must be re-written.

7. Attention

There are some principles, according to which it's possible to gain the user attention, such as for example by using flashing messages, bright colors, etc. Moreover, in order to display alarm and/or error messages, it's better to use the red color, instead the green color is more appropriate when showing messages such as "OK".

8. Display issues

The display should be organized in order to delete unnecessary information and to avoid to insert ambiguous messages.

The main benefit, in terms of investing in Advanced Human Machine Interfaces, is the simplification of factory processes and operations, because this system has the ability to monitor machines remotely. Moreover, another benefit is the capability to see real-time data, in such an easy way that it contributes to the reduction of complexity of the factory environment. In addition, in this way, factories are able to respond quickly to changing or challenging conditions, so the efficiency is improved and the downtime (time in which no activities are executed)

reduced. This system helps factories to have a significant cost reduction and less amount of wastes, and at the same time improve processes and factory profitability.

3. Tools and technologies

The following chapter is about the choice of using the Android Studio Environment to develop the ICOSAF app and it explains the benefits and the advantages of this tool. Moreover, it provides a presentation of the main characteristics of a Wear OS smartwatch and a comparative analysis between the main OS smartwatches available on the market. In conclusion, it presents the characteristics of the TicWatch S2, i.e. the smartwatch used as a prototype for testing the application and the reasons under this choice.

3.1. Android Studio Environment

Android Studio is the integrated development environment that has been used for the whole duration of the internship at Santer Reply. Android Studio is normally used for the creation and development of applications that will then be installed on the Android platform.

There are various advantages offered by this development environment:

- Editor of intelligent code. One of the integrated functionalities of Android Studio is an editor of intelligent code that makes it easier to write the code because it often offers suggestions on what to write.
- Emulator with many functionalities. With this service, Android Studio allows users to test their application on an emulator or even directly on a real smartphone, tablet or smartwatch.
- XML code. In Android Studio it is possible to directly edit the XML code, even if it is often automatically generated.
- Android Studio can be used for all the Android devices.

3.2. Wear OS

For the development of the application related to the ICOSAF project on the collaborative factory 4.0, it was decided to choose the Android operating system and, in particular, its Wear OS variant, suitable for wearable devices. Watches with Wear OS system guarantee, in fact, a high level of flexibility, thanks to the fact that they are compatible with both Android and iOS operating systems and boast of a very large user base.

There are various reasons why the smartwatch was chosen to be the wearable device used by the operators inside the factory. First of all, the fact that this device is particularly comfortable to wear as it is small in size, is not bulky and, therefore, allows the operator to perform his tasks correctly, having his hands free. Secondly, the smartwatch allows the integration of important features, such as vibration or sound to attract the attention of the wearer, and moreover, the fact that it is equipped with many sensors, that means that the operator can also communicate something to the watch through a simple gesture. Finally, this device supports Wi-Fi connection, therefore it does not need to connect to another device via Bluetooth but is capable to operate autonomously.

The most common features of a Wear OS Smartwatch that need to be taken into consideration when choosing such a watch are:

1. **The characteristics of the display and the shape of the smartwatch.** The display can be of the LCD type, which guarantees good readability under sunlight but has a greater impact on the autonomy of the device, or it can be of the OLED/AMOLED type, which guarantees good energy savings, since the black pixels are practically "off", resulting in a better rendering of black. However, these screens are less legible in the presence of natural light. Remaining on the autonomy subject, some smartwatches have an *always-on* screen, that allows to have a low consumption and that shows the time like a traditional clock, while other screens are activated only after a movement of the arm or in case of a touch on the screen. Instead, the shape of a smartwatch in the case of Wear OS devices can be either round or square/rectangular.

2. **Sensors and features.** Another important characteristic to take into consideration is the equipment of sensors and hardware chips, which naturally affect the set of features available. Some of the main ones are the speaker, the microphone, the Bluetooth, the Wi-Fi connectivity, and the GPS module to track the position, but for sportsmen, sensors such as the pedometer, the heart rate monitor, or the altimeter could also be important.

3. **SIM support.** This feature could be useful in case you need to make/receive calls completely independently from the smartphone. In this case, the device should also have support for 4G / LTE connectivity.

4. **Battery.** The battery is an essential feature to take into account when talking about smartwatches. It determines the amount of time in which it's possible to use the device and the longer it lasts, the better it is for the wearer, since such a device will soon be used as the smartphones are used today.

3.2.1. Different models of Wear OS Smartwatches

There are several Wear OS smartwatch models available on the market, sometimes even at competitive prices, as they are still smart devices. The following table (Table 2) contains an analysis on the main watch models of the moment, classified by price and by the advantages / disadvantages that their use entails.

Table 2: Main Smartwatch Wear OS models with their characteristics

SMARTWATCH	SHAPE	PRICE	ADVANTAGES	DISADVANTAGES
Fossil Sport	round	279 €	<ul style="list-style-type: none"> - Light design - Powerful 	<ul style="list-style-type: none"> - Lacks high-level feaures - Slow GPS
TicWatch Pro 3	round	300 €	<ul style="list-style-type: none"> - Very good battery that lasts up to 3 days - Fast and responsive processor 	<ul style="list-style-type: none"> - Sometimes poor interface - The sleep monitoring can be improved
TicWatch E2	round	159 €	<ul style="list-style-type: none"> - Price - Autonomy lasts up to 2 days 	<ul style="list-style-type: none"> - Lacks the NFC for the payments - Anonymous design

Fossil Gen 5	round	349 €	<ul style="list-style-type: none"> - Energy saving mode - Fast processor 	<ul style="list-style-type: none"> - More expensive than the other - Low-powered speaker
TicWatch S2	round	179 €	<ul style="list-style-type: none"> - Convenient - Waterproof - More robust structure because it is suitable to do sport - Equipped with the main sensors (ex. GPS) 	<ul style="list-style-type: none"> - Lacks the NFC (Near Field Communication) - Lacks the LTE variant
Misfit Vapor 2	round	143 €	<ul style="list-style-type: none"> - Convenient - NFC for Google Pay 	<ul style="list-style-type: none"> - Lacks the LTE variant - Fragile
TicWatch C2	round	188 €	<ul style="list-style-type: none"> - Cheap - Google Pay and GPS 	<ul style="list-style-type: none"> - Limited availability of straps - Outdated components
Huawei Watch 2	round	139 €	<ul style="list-style-type: none"> - Integrated GPS and NFC - Optional 4G 	<ul style="list-style-type: none"> - Small screen - Not always flowing screen
Polar M600	square	263 €	<ul style="list-style-type: none"> - It has WiFi and Bluetooth connectivity - Suitable for sports - Maximum autonomy of 2 days 	<ul style="list-style-type: none"> - The integrated speaker is missing
Skagen Falster 3 X	round	265 €	<ul style="list-style-type: none"> - High resistance to water - Equipped with many sensors (GPS, accelerometer, heart rate monitor, etc..) 	<ul style="list-style-type: none"> - Average battery life: 1 day
LG Watch W7	round	399 €	<ul style="list-style-type: none"> - It is a "hybrid" equipped with mechanical 	<ul style="list-style-type: none"> - It is expensive

			<p>hands capable of indicating the exact time even in the event of no battery</p> <ul style="list-style-type: none"> - The battery offers up to 36 hours of autonomy with active smart functions - There are many sensors, such as compass, barometer, altimeter, microphone, etc. 	
Suunto 7	round	479 €	<ul style="list-style-type: none"> - AMOLED display and stainless steel bezel - Withstands diving up to 50 meters underwater - It has the most important sensors - The battery can last up to 2 days in ordinary use conditions and up to 40 days in economy mode 	/
OPPO WATCH	square	205 €	<ul style="list-style-type: none"> - VOOC flash recharge, thanks to which the watch recharges in about 15 minutes 	<ul style="list-style-type: none"> - The battery lasts for a short period of time (less than 2 days)

3.2.2. TicWatch S2

Based on this comparative analysis on the main Wear OS smartwatch models, the device on which it was chosen to develop the application of the ICOSAF project is the TicWatch S2, very similar to the TicWatch E2, but different in terms of robustness and resistance, because the S2 model is particularly suitable for sports.



Figure 9: TicWatch S2 Smartwatch in white color

In 2019, an update was developed that improved many features of the TicWatch, making this watch even more performing. First of all, there was an improvement in terms of hardware, in fact now the processor is no longer *Mediatek*, but a more performing *Snapdragon Wear 2100*. The device display is 1.39 " diagonal in AMOLED technology, so that it is always bright and with an always-on to always show the time, just like classic watches, with low power consumption.

The main reason why it was chosen this particular smartwatch is because it is able to connect via the Wi-Fi, and so it does not need to be associated to a smartphone in order to work. This feature is very important in the industrial environment. The Wi-Fi connection has been considered more important than the 4G in this context

because since the pilot for this project is CRF (Centro Ricerche Fiat), it is better for them to use the factory's Wi-Fi connection rather than external networks, in order to avoid problems.

4. Implementation and design of the app

This chapter initially describes how the Android Studio platform organizes a project and it provides a brief explanation of the main parts of a common project. Secondly, it introduces the utility of the Server Sent Event protocol and how an Http connection is established in order to make API calls. By the way, the most consistent part is related to the implementation and design of the various user interfaces of the application. In addition, a subparagraph is dedicated to the explanation of the flow diagrams that represent the flow of the activities and the relationships between each other for the two Use Cases developed. In conclusion, the last two sections concern the testing and the distribution phases of the app.

4.1. Android Studio project

In order to start the creation of a new project on Android Studio, first of all the applicative context for the app has to be defined between Phone and Tablet, Wear OS, TV, Android Auto and Android Things. As a second step, it's necessary to define the app name, the package, its location and, at the end, the programming language to be used: Java or Kotlin. This project in particular has been developed only with the Java language.

A common Android Studio project (see Figure 10) is composed of three parts: a folder containing the Java code, a folder named `res`, that contains the resources realized in XML, so the correspondent layouts for each activity and a configuration file named *AndroidManifest.xml*, in which the user has to declare all the activities of the app, the user permissions (such as the `INTERNET.PERMISSION` and the `VIBRATION.PERMISSION`). The following picture shows how these elements are disposed in Android Studio.

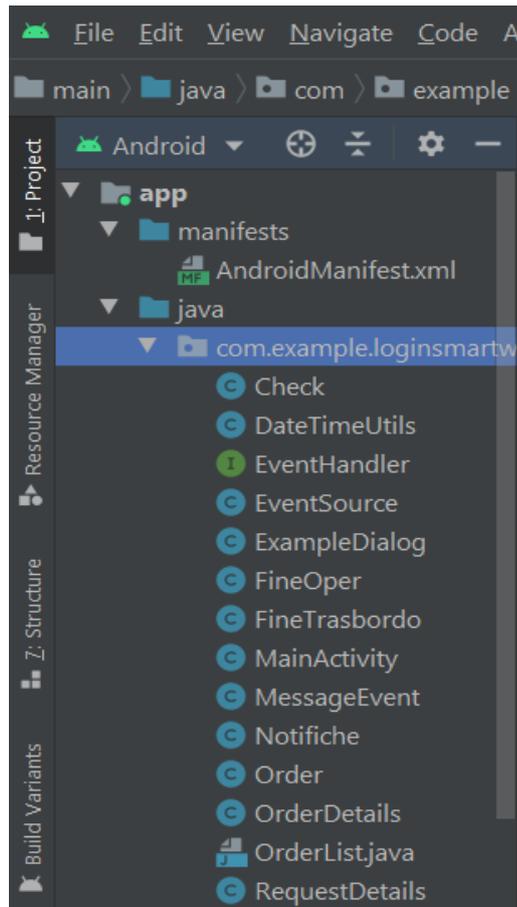


Figure 10: Structure of an Android Studio project

The entire project is contained in a folder named *app*, that is the default module and that contains the three parts of the project previously explained. After the *app* module, there's the section dedicated to the *Gradle Scripts*, that contains the build files that Gradle will use in order to transform the project in a functioning application. In particular, there are two build files: the first one is valid for the entire project and the second one just for the *app* module. This last is the one typically taken into account by the programmer because it needs some changes in order to have more functionalities for the application (this is the *dependencies* section).

The *Tools* menu contains the SDK Manager and AVD Manager sections that allow to respectively activate the Android SDK Manager and Android Virtual Device Manager. The first one is used to integrate the SDK with additional functionalities and updates, while the AVD Manager is useful in case it's not possible to run the

application directly on a real device, as it allows to create one or more emulators and choose their characteristics (shape of the device, dimensions, etc.).

The Android applications are created by putting together one or more components, each of them called *activity*. In detail, an activity is a single module relative to a functionality of the app and it is often correlated to a single user interface (UI). In the ICOSAF app, the main user interfaces are related to single activities, but there are also activities that do not match with any UI, because they are needed just to make the application to be executed correctly (for ex. the activities *EventSource.java*, *EventHandler.java*, *DateTimeUtils.java*, etc.).

Usually, the activities in an Android Studio project are independent of each other, but it's important to introduce the concept of *intent*. Intents are essentially the mechanisms that allow an activity to start another one, so they are useful in order to establish a flow that links the various activities of an application. An intent usually has the class in which the intent is created and the class that has to be opened as parameters. It is generally used together with `startActivity()` method in order to invoke a new activity to start.

4.2. Server Sent Events

The entire project is integrated with the Server Sent Events (SSE), that is a server push technology that enables a client to receive automatic updates from a server via HTTP connection. These events are commonly used to send message updates or continuous data streams to a browser client and designed to enhance cross-browser streaming through a JavaScript API called *EventSource* (it is an interface), through which a client requests a particular URL in order to receive an event stream. The *EventSource* interface has some properties (*onOpen*, *onMessage* and *onError*) that are an *EventHandler* that is called when a new event is received.

An example of how the event is received from the server side is shown in the picture below.

```
D/OPEN: Open
D/MESSAGE: Event: {"task_id":10,"agv_id":1,"area_id":1,"uc":"UC-C","delay":0,"status":"PEND"}
```

Figure 11: Example of a SSE event message

A Server Sent Event is an alternative technology to *WebSockets*, as both of them define how browsers and clients communicate with each other. WebSockets allows a real-time interactive bidirectional communication between the client browser and a server. So, in other words, this technology makes it possible for a client and a server to exchange data to transfer data in real time. The APIs of this technology allow to send messages to a server and receive event-driven responses without having to poll the server for a reply. This is useful, since the client and server can talk to each other without interruptions, because the connection remains open after a server response.

4.3. Http Connection

In order to access the network, the OkHttp library has been used. This library is useful to create request objects and make network calls. First of all, a client object of type OkHttpClient must be created:

```
OkHttpClient client = new OkHttpClient();
```

Secondly, it has to be defined the *url String* to pass and, eventually, even its parameters:

```
String url = HttpUrl.parse("https://...").newBuilder()  
.addQueryParameter("name", String.valueOf(...)).build().toString();
```

Then, a *request* object has to be created, passing as parameter the url just defined:

```
Request request = new Request.Builder().url(url).build();
```

In this case, these calls are asynchronous because when a web service is called, it's not necessary to wait until it returns, so it means that during this time the execution of the program is not blocked and the user can continue to interact with the page and/or program UI. In order to make an asynchronous call, it's necessary to create a *Call* object and use the *enqueue* method.

```
client.newCall(request).enqueue(new Callback() {
```

```
@Override public void onFailure(...) {...}
@Override public void onResponse(...) {...}});
```

The *onResponse()* method will contain what has to be done. Sometimes a Callback may start a thread in case it's necessary to update the UI of the application and a common way to achieve this is to call the Activity's *runOnUiThread()* method.

```
Activity.this.runOnUiThread(new Runnable() {
    @Override public void run() {...}});
```

4.4. ICOSAF app

The ICOSAF app has been developed with the purpose of representing the scenarios described in the Use Case A (UC-A) and Use Case C (UC-C) of the ICOSAF Project (see Table 3 in the Appendix). The application has been created to be used on Smartwatches and its scope is to support the operators inside the productive environments and support their collaboration both with remote operators and with the cobots in the picking area.

The Web interface for the project has been developed by using Angular, while the back-end has been made with Node.js, that was useful also to build the RESTful APIs for the project. A Swagger UI was developed to visualize and interact with the APIs resources and to do test calls to the back-end. The list of the API calls used for the project and mentioned in the following subparagraphs is available in the Appendix sector (Figure 29).

The pictures below are the result of screenshots taken using the Android Wear Square Smartwatch as an emulator for testing how the application works.

4.5. Use Case A

In the Use Case A, the operator is involved in two main tasks: control the “alberi” pieces and completing the “motore”. This Use case is way simpler than the other one, because the operator mainly interacts with a monitor that he has in front of

his workstation and, for little tasks, he interacts also with the smartwatch. In fact, he enters his credentials only on the monitor and, afterwards, the smartwatch will automatically recognize the operator id that has executed the access and explore the orders and tasks that he has to do. Because this use case provides that the user wears a pair of gloves for doing his job in the workstation, it was thought to completely delete an interaction with the smartwatch through the touch mode, but instead the operator can use wrist gestures in order to communicate something to the watch. This was possible because smartwatches are outfitted with accelerometers that, through some sensors, measure how movement changes over time.

The main interface on the smartwatch shown to the operator is the “Assemblaggio” one and, after he has done the login on the monitor, his username will appear.

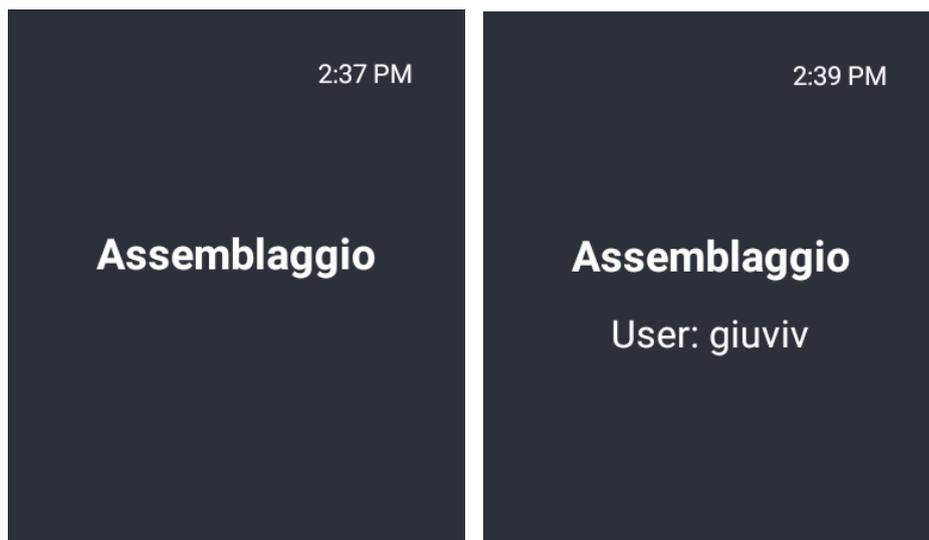


Figure 12: Main Activity pages

Meanwhile, the app takes the list of orders associated to that operator and, for each order, the list of tasks to do. There are certain tasks that require the operator to give particular attention. For example, as soon as the tasks regarding “Controlla Alberi” and “Completa Motore” are executed, a popup with a vibration is shown for each of them, in order to keep the operator’s attention and suggest him what he has to do. In this case, he has the possibility to close those popups with a wrist gesture.

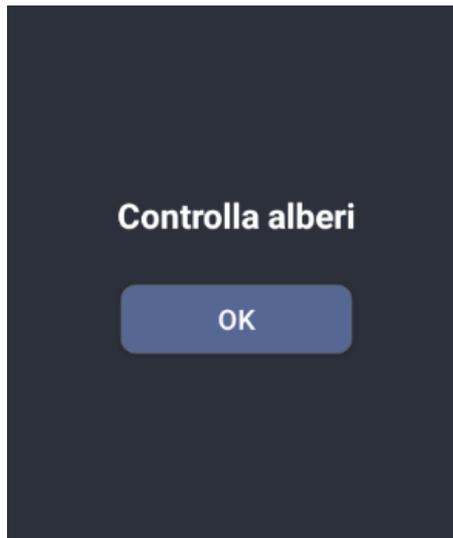


Figure 13: Popup "Controllo Alberi"

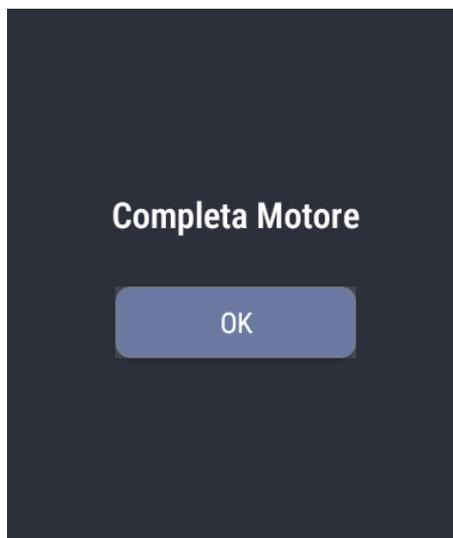


Figure 14: Popup "Completa Motore"

4.6. Use Case C

The Use Case C expects the operator to have more interactions with the smartwatch than the Use Case A. The main form of interaction is through the touch mode, but there's also the possibility for the operator to register a vocal message. In this Use Case, the user is able to control and monitor the status of the current order that he is executing through a progress bar. Moreover, he can see

notifications and can receive directly on the watch eventual requests for the AGV assistance.

4.6.1. Login

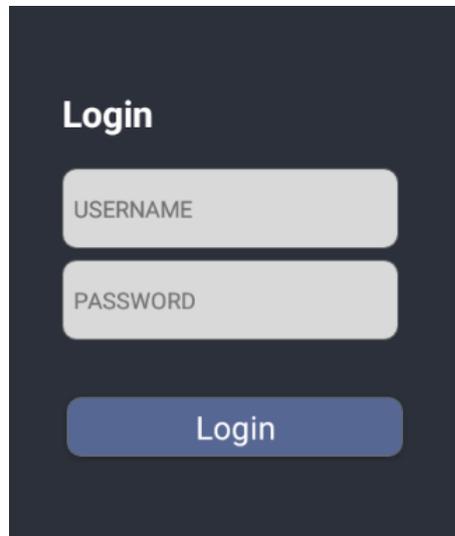


Figure 15: Screenshot of the Login page

The Login page (*MainActivity.java*) in Figure 12 allows the user/operator to access to his personal page (do the login) by inserting his credentials. According to the principles of the GDPR related to data protection, it's necessary to implement appropriate security measures in order to ensure a level of security appropriate to the risk and this is obtained by encrypting the user's personal data. If the operator's username and/or password do not match the right ones that he chose during the register phase, a transparent error message (Figure 13) will appear on the screen for just 3 seconds, then the operator has the possibility of re-inserting his username and/or password.

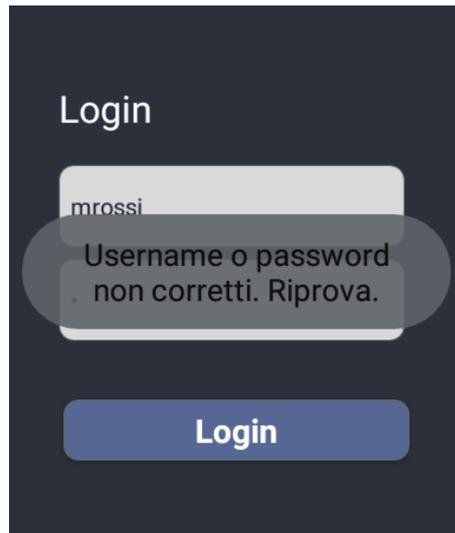


Figure 16: Screenshot of the Login page - error message

Once the user has written the correct username and password and has pushed the “login” button, a second activity opens automatically (*OrderDetails.java*).

4.6.2. Notifications and orders

The app, thanks to the reference in the code to an API call (*getOrderbyOper*) of type GET request, is able to use the username (ex. *mrossi*) inserted by the operator in order to extract a Json Array made of Json Objects that contains the list of orders that the operator has to do and, for each one (*order_id*), its correspondent status (*order_status_id*).

```
[{"operator_name":"Mario","operator_id":1,"order_id":3,"order_status_id":1},{"operator_name":"Mario","operator_id":1,"order_id":1,"order_status_id":6}]
```

In fact, each order can have six different *order_status_id*, based on the current status of that specific order:

- 1. Created
- 2. Started
- 3. In execution
- 4. Completed
- 5. Pending

- 6. Failed

Depending on the status of each order, the code has to select the first order that has to be executed by the operator (an order won't be proposed to the operator in case its status results to be "Pending" or "Failed") and has to insert the correspondent `order_id` into another API call (`getTaskListOper`), obtaining another Json Array made of various Json Objects, that contain the tasks to be done for that specific order and, for each task, the name of the task (`task_descr`) and a `task_status_id` that can be equal to 1 in case that task has not been completed yet and equal to 2, instead, if the task has already been executed. An example of such a Json object is the following:

```
{"task_id":17,"det_short_id":"11092141F","task_descr":"KIT42100","start_time":null,"error_time":null,"stop_time":null,"task_status_id":1}
```

In this way, the code, after having selected the first order to be executed, it is able to choose, for that order, the first task that has not been completed yet and displays it to the operator's interface in the exact moment in which this page (`OrderDetails.java`) is opened.

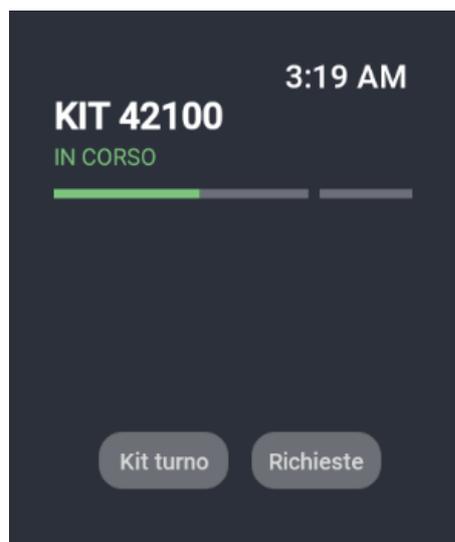


Figure 17: Screenshot of OrderDetails.java

The first green progress bar on the screen represents the percentage of completed tasks over the entire order, while the second progress bar is uniquely referred to the “trasbordo” phase and it will become green as soon as this activity is completed. As each task is completed, this page will refresh automatically after receiving a response from the backend, and so the progress bar will advance and a new task to do will appear on the screen. Actually, this system will function depending on the pick to light only in the real factory. In this case, for the moment it was only possible to simulate this process, by using a Swagger UI, through which it is possible to generate events, thanks to the use of API calls (in this case the *updateStatusOk* GET request). For this specific case, the app, during the whole running of the *OrderDetails.java* page, listen to events, that report when a task has been completed, in the form of a Json Object, such as the following:

```
D/MESSAGE: Event_task: {"task_id":70,"error_type_id":null,"start_time":"2021-02-03T15:37:05.450Z","agv_id":null,"oper_id":1,"area_id":1,"kit_name":"KIT 42100","uc":"UC-C","det_short_id":"11262020F","delay":0,"status":"OK"}
```

If the *task_id* contained into the Json Object coincides with the id of the current task shown on the screen to the operator, it means that the current task has been executed and, as a consequence, the page will refresh automatically and show the operator the immediate next task to execute. The java page is refreshed by using the *intent* and its *getIntent()* and *startActivity()* methods.

In this Java page, in order to show to the operator the task that has to be done, each time the code executes two calls, one to *getOrderbyOper* and another one to *getTaskListOper*. The two calls are asynchronous because they make use of the *enqueue()* method in order to make the Http requests. In detail, this method sends the request and notify callback of its response or if an error occurred when talking to the server when creating the request, or processing the response.

Since in a java activity it's not possible to make two asynchronous calls simultaneously, the two calls have been made nested, that is the second one is inserted inside the *onResponse* of the first one in the form of a method. In this

way, the two calls do not exist together in the `onCreate`, but only one is present inside the `onCreate`, the other one is out of it.

On the `OrderDetails.java` page, if the operator pushes the “**Richieste**” button, the code makes a query to the `getNotificationList` API call and, in this way, he will be able to see the list of notifications that he received with the correspondent time of receiving.



Figure 18: Screenshot of the Notifications page

Next to each notification there's a correspondent circle, that can be green in case the notification has been already completed or grey in case it has still to be done by the operator.

Instead, if in the `OrderDetails.java` page the operator pushes the “**Kit turno**” button, he opens the list of orders that has to be executed. Next to each order, there's circle that can be **green** in case the order has been completed, **yellow** for the current order, **grey** for the orders that still have to be executed, **orange** for the pending orders and **red** for the failed ones. The color of each order depends on the `order_status_id` previously mentioned.

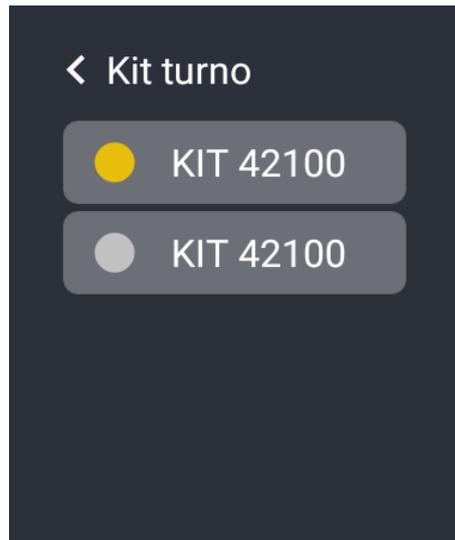


Figure 19: Screenshot of the list of orders to execute

The order that has a yellow circle next to its name is the current order that has to be executed and the operator is prohibited to execute some tasks of the next order that he has to do, unless he has finished all of the tasks of the current order.

By clicking on the order, the operator will open a new page, whose title is the name of the order clicked, containing the list of tasks referred to that specific order.

Of course, to each task corresponds a different color, in order for the operator to intuitively recognize the status of each order, according to the principles that explain the functioning of Human Machine Interfaces. The green circle represents a task that has already been executed, the yellow one is referred to the current task that the operator is doing and the grey ones are the ones that the operator still has to do.

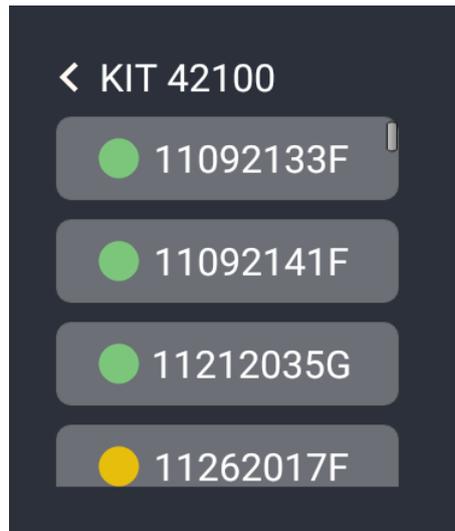


Figure 20: Screenshot of the list of kits assigned for an order

Even if the operator has already completed many orders, this page will show him just four orders at a time, with a focus on the current task that the operator is executing in order to show him the two previous completed orders, the current one and the next one to execute. The focus on this part of the list has been made with the `setSelection` method applied to the *ListView* referred to the list of tasks.

Both the activities represented in Figure 16 (list of orders) and Figure 17 (list of tasks), during their execution listen to events referred to the completion of a task. In particular, if this tasks comes in the form of a Json object and as `task_id` contains the current task and the `status` equals “OK”, then the app will automatically return to the `OrderDetails.java` page (Figure 14), showing the operator the next task that he has to do.

In this regard, in fact, the app is “intelligent” and remedies any operator distractions, such as leaving the screen with the order list or the task list open (static activities). In this case, as soon as the app receives an event containing the completion of an order, it automatically redirects the operator to the activity that shows him the next order to be executed. The reason of this choice is to avoid that the screen remains on a static page for a long time.

The last task to be done for each order is called “Trasbordo” and it requires that a popup appears to the user to advise him that this activity is going to start. The popup that appears on the screen is associated to a vibration of the device in order to call the user’s attention to the smartwatch. In order to make the device vibrate

under certain conditions, it has been necessary firstly to give the permissions in the AndroidManifest file:

```
<uses-permission android:name= "android.permission.VIBRATE"/>
```

Secondly, a *vibrator* object of type *Vibrator* has been created and a definition of the time to vibrate, expressed in milliseconds, has been set up:

```
final Vibrator vibrator = (Vibrator) getSystemService (VIBRATOR_SERVICE);  
vibrator.vibrate(500);
```

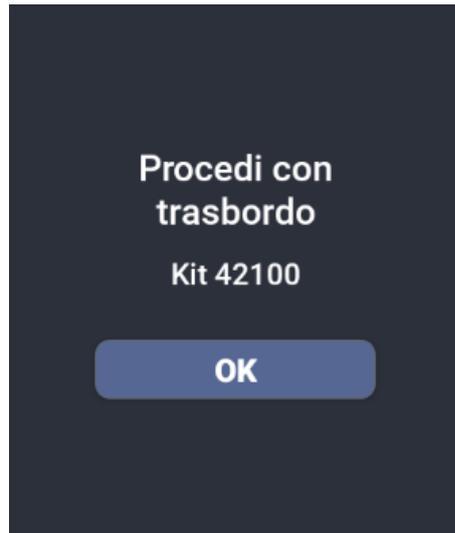


Figure 21: Popup for the ‘trasbordo’ start

The popup also shows to the operator the name of the order associated to the trasbordo task. This popup has been created by using the `AlertDialog` class and the `DialogFragment` as a container for that class.

When the operator is ready to begin that activity and after he pushes “OK”, he will be able to see on the screen the details of that activity and also the “Conferma fine trasbordo” button that until this moment was kept hidden, instead now it is clearly visible to the operator, by using the `setVisibility(View.VISIBLE)` method. He will click this button when finished, to conclude the activity.

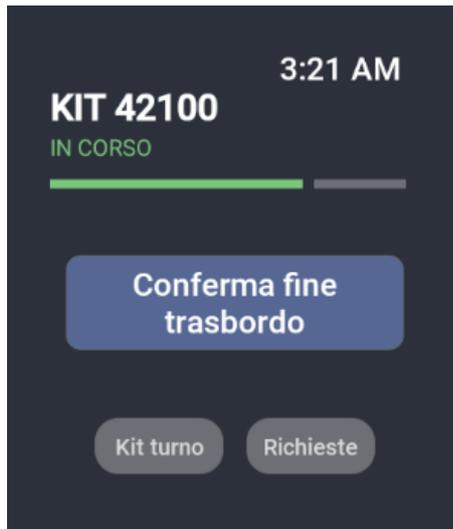


Figure 22: Activity during the ‘trasbordo’

As soon as the operator wants to conclude the activity, a popup will appear on the smartwatch screen to ask him if the operation is finished and, in positive case, he will receive a green check on the screen to confirm the positive outcome.

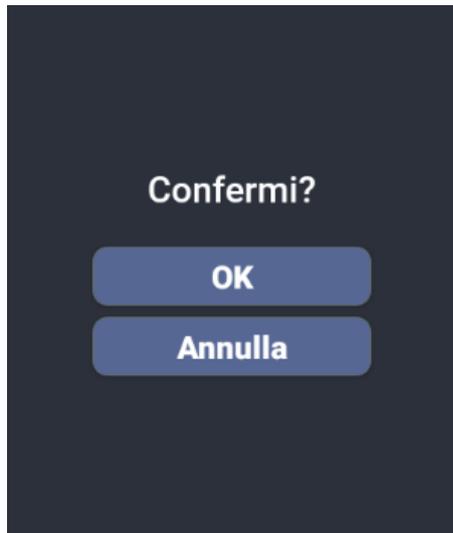


Figure 23: Screenshot of the FineOper page

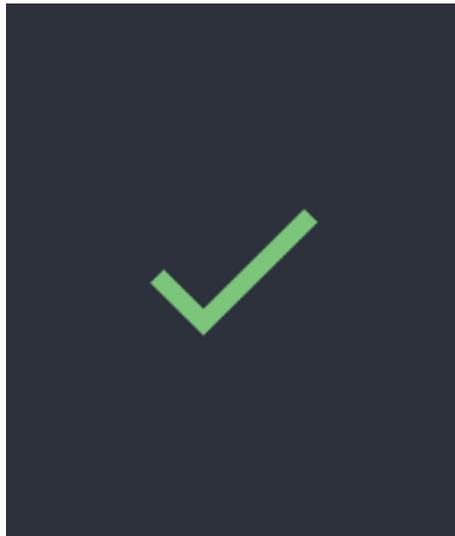


Figure 24: Check to confirm 'OK'

The check that appears on the screen is not the result of another Java activity, but it's made using the `Toast`, that is simply a view containing a quick little message or an image for the user and it is shown on the screen for just few seconds, then it will disappear. It is usually displayed as a transparent and floating view over the application and it doesn't occupy all of the space available in the screen, allowing the user to still see what's behind. However, in this case, this last requirement wasn't needed and so it was ordered to the `Toast` view to occupy the full screen. As concluding the "Trasbordo" activity also mean concluding an entire order (because the "Trasbordo" activity is the last task of the order), the code makes two API calls to the backend: the first one by calling `updateStatusOK` in order to update the `task_status_id` of the `Trasbordo` task equal to 2 (task completed) and the second one to `updateOrderStatus` in order to modify the `order_status_id` of the order equal to 4 (order completed).

Of course, after the `trasbordo` task of an order has been completed, the app will refresh and automatically show on the screen the immediate next order to execute (with its relative task).

4.6.3. Request for AGV assistance

Sometimes it may happen that the AGV needs assistance from one operator when executing a certain task. In that case, a remote operator recognizes which is the problem and through a popup, that will appear on the screen of the smartwatch and that makes the device vibrating in order to capture the user's attention, will show the operator in the factory what has to be done, the AGV id that needs assistance and the type of urgency. As before, the popup has been created by using the `AlertDialog` class.



Figure 25: Popup for AGV assistance

By clicking “Accetto”, the operator will be able to see the details of that request: what to do, the type of urgency, the AGV that needs assistance and the order of reference.



Figure 26: Details of the request



Figure 27: Screenshots of the Procedimento.java pages

If the operator in Figure 23 pushes “Aiuto”, a new page (Figure 25) will appear with three possible options to select. The last option provides the possibility of

recording a voice message by pushing the microphone icon on the screen and, when the “Invia” button is clicked, the option selected by the operator in the factory is sent to the remote operator. After registering a voice message, it will be directly sent to the backend via an Http Post request



Figure 28: Screenshots of the Aiuto.java pages

This is possible thanks to the fact that smartwatches are usually equipped with the microphone, but they don't have a speaker. Consequently, it was not possible to insert a function for which the operator can re-listen his voice message before sending it, as it commonly happens with smartphones. Anyway, to integrate the microphone tool in this page, it has firstly been necessary to give to app some permissions in the AndroidManifest file, such as:

```
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"
/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"
/>
```

Secondly, it has been created a *MediaRecorder* object, setting the *AudioSource*, the *OutputFormat*, the *AudioEncoder* and even the *OutputFile* where to put the registered file. Then the *MediaRecorder* object calls the methods `prepare()` and

start() in order to start recording the voice message and the stop() and release() method to finish recording. Moreover, in order to avoid loading the smartwatch's memory with all of the voice messages, every time a new record is created, it is automatically overwritten on the previous one.

Instead, if the operator clicks on “Fine” (in Figure 23), another page will open (Figure 26) and he has to confirm that the operation is finished and, in this case, if the result is positive, a green check will appear. As soon as the operator pushes “Ok”, the app makes an API call (*updateStatusOk*) to the server, in order to update the status_id of the task just executed to 2 (completed).

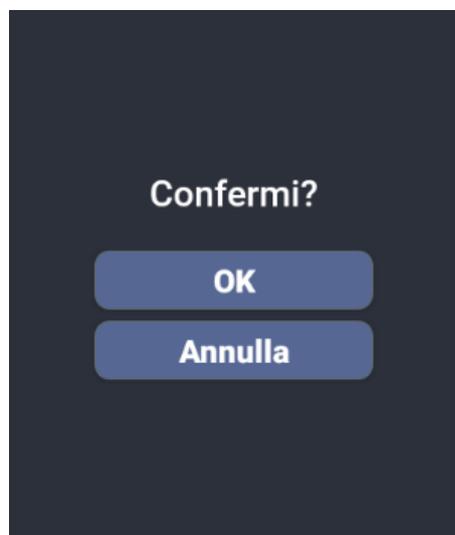


Figure 29: Screenshot of the FineOper.java page

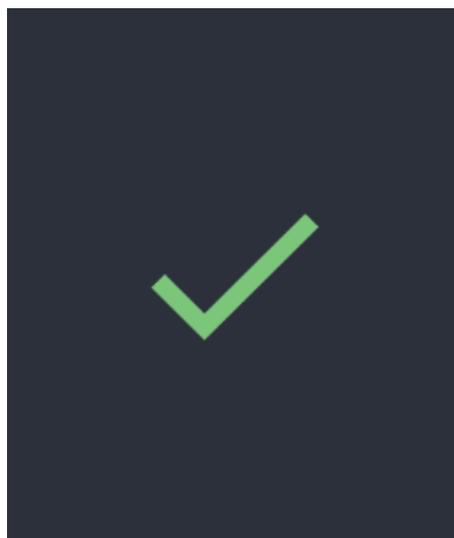


Figure 30: Check to confirm 'OK'

From the exact time in which the operator receives the new request (Figure 22) until the time he presses the OK button in the Figure 26, the code exploits the `DateTimeUtils` class to save the two times in two variables (in the code *TimeRicezione* and *TimeFinito*) and their difference gives as a result the *delay*, so the time it takes for the operator to complete the request. This data is not used to be shown on the user interface, but it will be useful for statistical purposes.

4.6.4. Flow Diagrams

UC-C

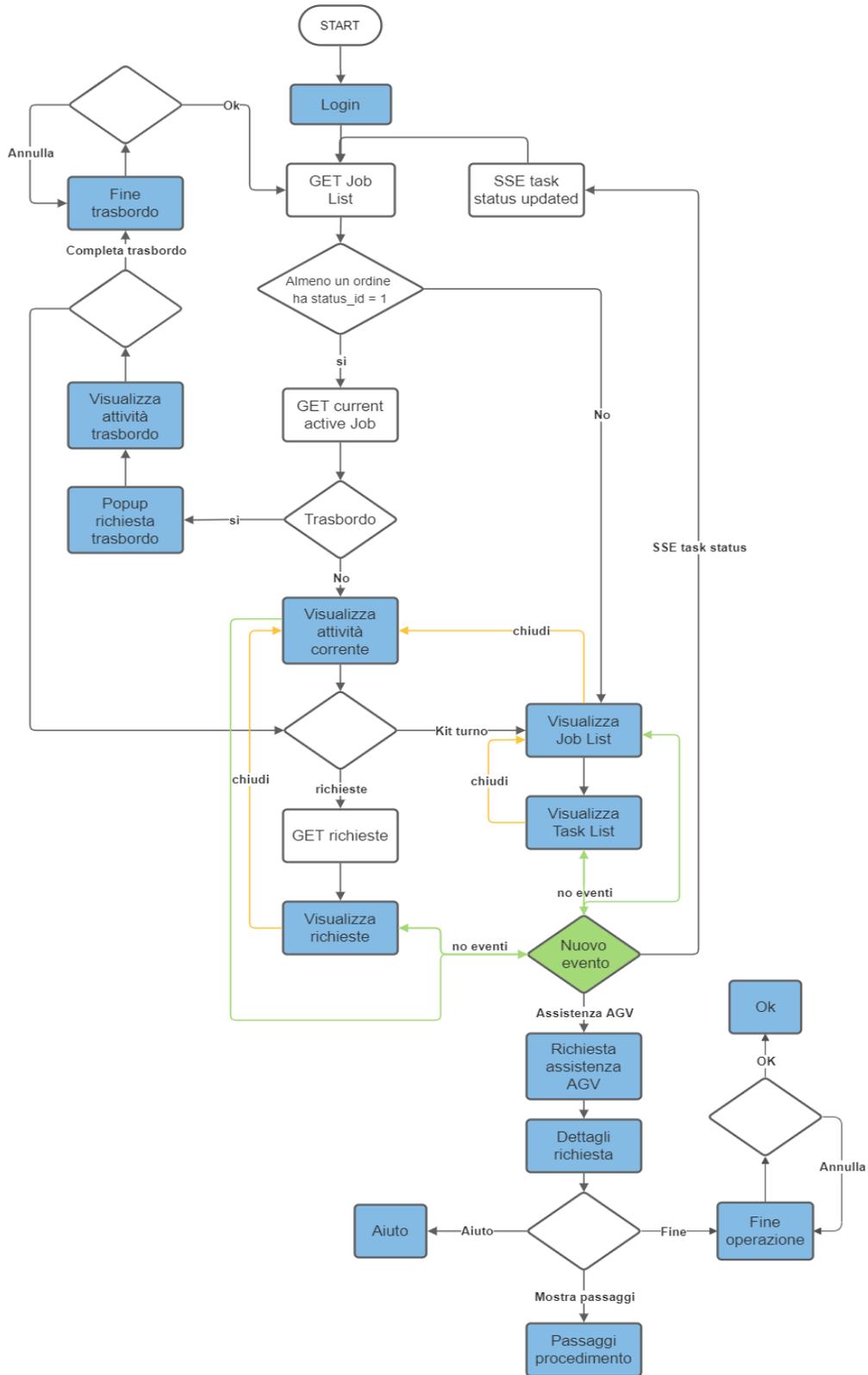


Figure 31: Flow diagram for use case UC-C

The above flow diagram represents the set of dynamic relationships between the activities of the application. In detail, the parts colored in light blue are referred to those activities that are shown in the user interface, while instead those in white are actions that the code perform in the background but that contribute to have an impact on what is shown in the interface. The rhombus in green is referred to the upcoming of a new event through the Server Sent Event technology.

UC-A

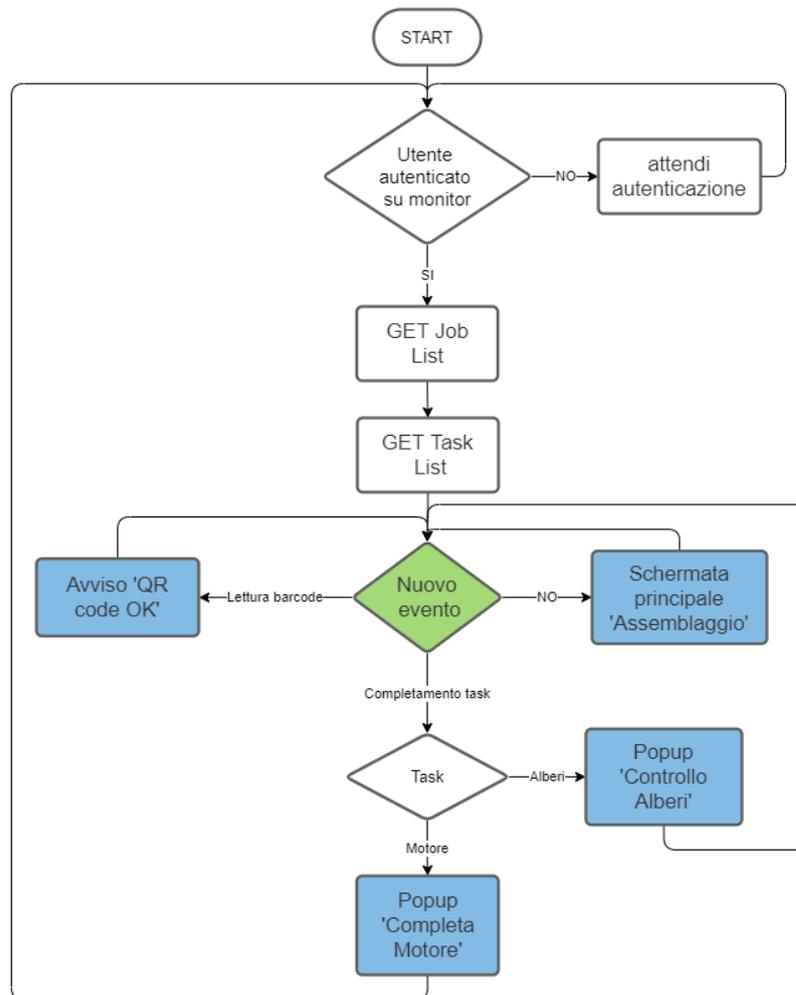


Figure 32: Flow diagram for Use Case UC-A

4.6.5. Testing of the ICOSAF app

In order to test the proper functioning of the app, during the whole development of the project, it was used an emulator directly available in Android Studio. Instead, when the app was almost on its end and a real smartwatch (TicWatch S2) was brought as a prototype, the app has also been tested there. In order to install an Android Studio app on a real device, it's necessary to link the device to the pc through the USB cable, then open the watch's *Settings*, then tap *System > About*, scroll to *Build number* and tap it 7 times, until a dialog won't appear on the screen confirming that you are now a developer. Afterwards, it's necessary to click on "Developer options" and enable "ADB debugging". Therefore, this device will appear on Android Studio as one of the available devices where to run the application. In this way it's possible to test the app on a real device.

4.6.6. Packaging and distribution of the app for partners

To make a Wear OS app appear in the on-watch Play Store and download the app directly on the watch, it is necessary to upload the watch APK in the Play Console. In addition, it's necessary to set up the targeting for a watch, so that the app is downloadable for watches. This is done by setting the `uses-feature` element to `android.hardware.type.watch` in the **Android Manifest** module, in this way:

```
<uses-feature android:name= "android.hardware.type.watch" />
```

Conclusions

Nowadays, the development of an application on a wearable device is still not as widespread as it is for smartphones or tablets. Because of this, it was initially not very clear whether certain features (such as the integration of the microphone and the ability to record a voice message) could be applied also to a wearable device, for the considerable limitations that this device presents in comparison to a smartwatch (for ex. it has smaller size, fewer integrated sensors and therefore less degrees of freedom and less flexibility). So, it can be said that this work was a challenge both from a personal and a professional point of view. The field of information systems and software development is not exactly my field of study, having attended the Engineering and Management course. Because of this, in the initial phase of the internship, I had to familiarize with the Android Studio Environment and with the Java programming language. Despite that, this work experience has allowed me to acquire and develop new skills also in this sector. It was interesting to take part in such a large and prestigious project and it was also very stimulating to participate in the project calls with partners, such as CRF, to discuss about the user interfaces.

The real potential of this application is the fact that it is very user friendly and that particular attention has been paid to try to avoid inserting too much information on each user interface. Furthermore, it was necessary to respect the principles that regulate the functioning of the HMIs, according to which the various activities with which the user interacts, need to be as intuitive as possible (so for example it was decided to use the green color to indicate completed operations, yellow for those in progress, etc.). A further strength of the application is the fact that it has been designed to be used on wearable devices, to make the operator wearing the device completely free to be able to carry out his work without impediments or distractions. In this regard, in fact, the app is “intelligent” and remedies any operator distractions, such as leaving the screen with the order list or the task list open. In this case, when the app receives an event containing the completion of an

order, automatically redirects the operator to the activity that shows him the next order to be executed.

Of course, some improvements may be done in the application and in the user interfaces as soon as the app will be addressed to be used by the operators inside a real factory, to meet the needs of a specific collaborative environment.

Appendix

Table 3: ICOSAF Use Cases proposed by CRF (Centro Ricerche Fiat)

ID	Partner Rif.	Short description
UC1	CRF	Station with collaborative robot working on two AGV lines (instead of conveyor) right and left
UC2	ADL	1 between: <ul style="list-style-type: none">- Picking and sequencing of laminated panels to produce CFRP body- Mounting of accessories to composite bodies (pins, studs, inserts)
UC3	CRF	Robotized AGV for kitting in cooperation with AGV conveyor (KIT, or KIT on trolley)
UC4	CRF	Thermocamera device for interior water infiltration
UC5	CRF	Cooperative robot for ultrasonic welding quality analysis integrated with Big Data System.

The image shows a Swagger UI interface for an HMI back-end. The title is "HMI Back-end for WMS and HMI development". Under the "default" tag, there are nine API endpoints listed, each with a method (GET or POST), a path, and a description:

- GET** `/details` Get all machine part
- POST** `/verifypwd` Authentication mechanism
- POST** `/verifypwdArea` Authentication mechanism with area_id
- GET** `/updateSolveActionOK` Update solve action to completed status
- GET** `/insertSolveAction` Insert solve action for the problem risen during process
- GET** `/getNotificationList` Get list of notification filtered by operator_id
- GET** `/getSolveActionDet` Get list of notification filtered by specific solve action id
- GET** `/createTaskOper` Create task for operator to solve problem risen during process
- GET** `/insertNewOrder` Insert new orders into DB

Figure 28: Swagger UI with API calls - 1st part

GET	<code>/getTaskDetails</code>	Get details for the task
GET	<code>/getTaskListAGV</code>	Get all tasks for specific order for specific AGV
GET	<code>/getOrderbyOper</code>	Get all orders for specific operator
GET	<code>/cleanOrder</code>	Reset all task_status_id to 1 and start_tim and error_time to NULL for spezific order_id
GET	<code>/updateOrderStatus</code>	Update order status to completed at the end of the simulation
GET	<code>/getLastActError</code>	Get last active error for specific task_id
GET	<code>/getTaskListOper</code>	Get all tasks for specific order and specific operator
GET	<code>/getTaskListOrder</code>	Get all tasks for specific order
GET	<code>/getTaskStatus</code>	verify task status
GET	<code>/getMappingErAct</code>	get mapping for error_type_id and solve_action
GET	<code>/updateStatusEr</code>	update task status to error

Figure 29: Swagger UI with API calls - 2nd part

GET	<code>/updateStatusOk</code>	update task status to completed
GET	<code>/updateOKEventOper</code>	update task status to completed
GET	<code>/getListOper</code>	Get operators list for specific order
GET	<code>/getSolveActList</code>	Get list of solve action for specific error
GET	<code>/getOrdListAGV</code>	Get order list for specific date
GET	<code>/getOrdList</code>	Get order list for specific UC
GET	<code>/getListAGV</code>	Get list of AGV for specific order
GET	<code>/getOrdListbyDate</code>	Get order list for specific date and specific Use Case

Figure 30: Swagger UI with API calls - 3rd part

References

- **Akpa, M. Fujiwara, H. Suwa, Y. Arakawa**, *A smart glove to track fitness exercises by reading hand palm*, 2019.
- **Carbone**, *Le aziende ed il Cloud*, 2018.
- **Chen B., Wan J., Shu, L., Li P., Mukherjee M., Yin B.**, *Smart factory of Industry 4.0: Key technologies, application case, and challenges. IEEE Access*, 2017.
- **Colaço et al.**, *Mime: compact, low power 3D gesture sensing for interaction with head mounted displays*, 2013.
- **F. Boschi, A. De Carolis, M. Taisch**, *Nel cuore dell'Industry 4.0: i Cyber-Physical Systems*, 2017.
- **Jiafu Wan et al.**, *Smart Factory of Industry 4.0: Key Technologies, Application Case and Challenges, IEEE Access*, 2017.
- **K. Bhaskaran, A. Nair, K. Ram**, *Smart gloves for hand gesture recognition: Sign language to speech conversion system*, 2016.
- **L. Francés, P. Morer, M. Rodriguez, C. Aitor**, *A review of wearable technology for smart gloves instrumentation technology*, 2019.
- **L. Lee, P. Hui**, *Interaction methods for smart glasses: a survey*, 2018.
- **M. White**, *Digital workplaces: vision and reality*, 2012.
- **Mesut Cicek**, *Wearable Technologies and its future applications*, 2015.
- **N. Shariat Zadeh et al.**, *Integration of Digital Factory with Smart Factory based on Internet of Things*, 2016.
- **O'Flynn, J. Torres Sanchez, S. Tedesco, K. Curran**, *Novel Smart Glove Technology as a Biomechanical monitoring tool*, 2015.
- **P. O'donovan, K. Leahy, K. Bruton, DTJ O'Sullivan**, *Big data in manufacturing: a systematic mapping study*, 2015.
- **SJ. Shin, J. Woo, W. Seo**, *Developing a Big Data Analytics Platform Architecture for Smart Factory*, 2016.
- **Z. Shi et al.**, *Smart Factory in Industry 4.0*, 2020.

- *Debug a Wear OS app*, in *Android Developers*,
<https://developer.android.com/training/wearables/apps/debugging>
- *How to take advantage of Cloud Computing for Industrie 4.0 and Enterprise 4.0*,
<https://www.reply.com/red-reply/en/Shared%20Documents/Red-Reply-How-to-take-advantage-of-cloud-for-Industrie4.0-and-enterprise4.0.pdf>
- *Package and distribute Wear apps*, in *Android Developers*,
<https://developer.android.com/training/wearables/apps/packaging>

List of figures

Figure 1: Structure and interactions between the Work Packages of the project ..	7
Figure 2: Relationships between vertical, horizontal and end-to-end integration	10
Figure 3: The three concepts behind the “Smart factory” term	11
Figure 4: Example of the smart factory framework.....	12
Figure 5: Example of a Cyber-Physical system	13
Figure 6: Picture of a common smartwatch	21
Figure 7: Example of smart gloves	24
Figure 8: Example of smart glasses and augmented reality	26
Figure 9: TicWatch S2 Smartwatch in white color.....	38
Figure 10: Structure of an Android Studio project	41
Figure 11: Example of a SSE event message.....	42
Figure 12: Main Activity.....	45
Figure 13: Popup "Controllo Alberi"	46
Figure 14: Popup "Completa Motore"	46
Figure 15: Screenshot of the Login page	47
Figure 16: Screenshot of the Login page - error message.....	48
Figure 17: Screenshot of OrderDetails.java	49
Figure 18: Screenshot of the Notifications page.....	51
Figure 19: Screenshot of the list of orders to execute.....	52
Figure 20: Screenshot of the list of kits assigned for an order.....	53
Figure 21: Popup for the ‘trasbordo’ start.....	54
Figure 22: Activity during the ‘trasbordo’	55
Figure 23: Screenshot of the FineOper page.....	55
Figure 24: Check to confirm ‘OK’	56
Figure 25: Popup for AGV assistance.....	57
Figure 26: Details of the request	58
Figure 27: Screenshot of the Procedimento.java page.....	58
Figure 28: Screenshot of the Aiuto.java page	59
Figure 29: Screenshot of the FineOper.java page	60

Figure 30: Check to confirm ‘OK’	60
Figure 31: Flow diagram for use case UC-C	62
Figure 32: Flow diagram for Use Case UC-A	63

List of tables

Table 1: Main smartwatch producers on the market and their pros and cons.....	23
Table 2: Main Smartwatch Wear OS models with their characteristics.....	39
Table 3: ICOSAF Use Cases proposed by CRF (Centro Ricerche Fiat).....	63