POLITECNICO DI TORINO

Faculty of Information Technology Engineering

Master of Science in Computer Engineering

Master Degree Thesis

# Digital Twin for collaborative spaces and Web Applications development in the context of Industry 4.0

**Supervisor:**
Prof. Paolo Garza

**Candidate:**
Walter Forcignanò

**Company Supervisor**
**Santer Reply SPA**
Dr. Liudmila Dobriakova

Academic year 2019-2020

*Alla mia famiglia*

# Summary

Over the past, our society has taken part to important transformations from the technological point of view. New technologies have been introduced and old ones have been modernized in a continuous process that brought our society where it is today.

The digitalization can be considered as one of the mainstays of this process, bringing companies to stand out by becoming leaders in new sectors, and others to collapse. But digitalization can be seen as a portion of a wider trend: Industry 4.0.

The "Fourth Industrial Revolution", an important milestone in the technology evolution has combined advanced manufacturing techniques with the IoT, in order to be able to use analyses and information to take important decisions in the digital manufacturing environment and basically forcing companies to rethink about their organization.
This trend relies on the employment of important technologies like Cloud computing, IoT (Internet of Things), Big Data, Artificial Intelligence (AI) with the aim of creating the so called smart factories.

They are called smart because with these technologies, products and tools in the production process get networked enabling in this way communication, sharing of relevant data, new ways of production and real-time optimization.

In a scenario where the cooperation among the different components of a company communicate with each other sharing data, a technology which is gaining more and more importance is: Digital Twin

The Digital Twin, one of the most promising technologies for smart manufacturing and Industry 4.0, consists of creating a virtual representation of a physical resource like a production process, a product or a robot by means of the data collected about it, with the purpose of recreating all the functionalities, interaction and communication interfaces for optimization, diagnosis, validation and prediction purposes.

The Digital Twin can be seen as a connected sets of simulation systems and models capable of exploiting data to obtain an effective response to the business requirements.
The reason for which it is becoming wide spread nowadays is that it can rely upon key enabling technologies like high speed wired and wireless network, reliable sensors, data storage and computer with an higher level of maturity and with a relatively lower cost.
In order to interact with such a virtual twin, it is necessary to introduce a layer able to create an interface between the human and the robot: a Human Machine Interface.
The HMI include all the elements that a person will touch, see, hear or use to perform control functions and receive feedback on such actions.
The definition of HMI in cooperative systems is fundamental to obtain advanced functionalities in terms of control and management of the activities but also for monitoring tasks.
Different kind of applications can be implemented to fulfil the task of providing all the functionalities deemed necessary to control a digital twin, but thanks to the spread of internet, almost all the users have already familiarized with web applications and since in an environment full of stimuli, as it can be a collaborative space in a manufacturing industry, such a technology could be helpful in facilitating users in performing their work routines.
In the context of Industry 4.0 one of the simplest and effective solution, even though it's not the only one, is indeed the internet application, which is implemented for taking care of many objectives: to visualize, control, monitor processes, machines or, generally speaking, resources.
Entire infrastructures of sensors are used to collect data about specific aspects, problems and features but in order to visualize or simply take advantage of such knowledge an application to the final user must be provided.
In this thesis work, solutions to specific problems related to the presented technological scope have been proposed, in the context of two different projects:

- ICOSAF
- DiPreTreat

The work for both the projects was in collaboration with the partner Reply S.p.A., that provided a lot of documentation and support to the realization of this thesis work.
The ICOSAF project, whose name stands for Integrated COllaborative Systems for smArt Factory where its leader is CRF (Centro Ricerche Fiat) is collocated in the specialization area of smart factory.

The DiPreTreat project instead, whose name stands for Diagnosis, Prognosis and Treatment of technological pathology affecting European cities is collocated in the specialization area of Innovation and Research specifically in CTS (Digital cities) segment.

The focus of the first is to develop and integrate technologies related to the world of systems and tools for collaborative factories, with increasing operator integration according to the concepts of industry 4.0 (interconnected automation) and industry 5.0 (humanization and reuse of resources).

The key players of industry 5.0 are cobots, and their main difference with respect to robots is their ability to work in collaborative spaces with both humans and other cobots in a secure and efficient way.
The ultimate aim of the project is the operator enhancement, because in order to exploit the workers intelligence and flexibility in tasks where the human factor is more important, cobots are used in collaborative spaces for low added value tasks. More specifically the activity performed is about representing a specific process and the cobots involved, by means of the Digital Twin, to simulate, predict, optimize and eventually to integrate the developed system with a dedicated HMI which is used instead for reflecting its status, for controlling it and for visualizing meaningful insights obtained with the collected data.

The process considered in this thesis work is the one depicted in the use case A.
This process is about the kit preparation for counter-rotating shafts in which several cobots are involved, going from the AGVs (Automated Guided Vehicles) to the cobots of Franka and Kuka which are two robotics arms with their own characteristics and peculiarities.

According to the demands of CRF and the partners involved in the project, a virtualization of a collaborative work area has been realized, where all the involved actors perform their own routines, synchronizing when necessary.

In parallel, it was developed an HMI which allows, with a bidirectional communication, to control and monitor the simulated process.
This HMI has been developed in such a way to be intuitive and effective in showing relevant information such as the current status of the different working areas and cobots involved in the process.
From the integration of the two systems by means of the definition of a RESTful application, significant insight have been produced, indeed it was possible to evaluate the feasibility, the efficiency and also possible improvements to be introduced for the proposed solution.

The aim in the DiPreTreat project instead, consists in building a functional and robust software platform able to address a problem that afflicts concrete buildings, in particular the corrosion of reinforcing steel.
Thanks to implementation of a dedicated software application indeed, specific functionalities have been provided such as the possibility of accessing the information gathered from the sensors installed in the different facilities, the possibility of producing a data presentation able to give a timely insight on the problem and furthermore the possibility to process those data with AI-based algorithms aiming in supporting corrosion issues predictive maintenance and forecast.

A Single Page Application has been realized which allows the view of long term measurements on the concrete deterioration parameters.
The implemented solution is only a link in the chain of components of a wide infrastructure, by means of which the final user is able to monitor the status of the different facilities and eventually take important decisions to solve the problem.

This application perfectly fits in the context of Industry 4.0.
Behind the web application that has been developed for this thesis work, an entire network of sensors collects the data that after some analysis and processing are used for reflecting the state of the various facilities but also as input for producing forecasts based on AI-based algorithms which eventually are shown to the final user after the sharing of those data with a client server communication, in order to take advantage of this information and apply prevention actions.

# Contents

# List of Figures

# Nomenclature

| **Acronym** | **Meaning** |
|---|---|
| DT | Digital Twin |
| HMI | Human Machine Interface |
| AGV | Automated Guided Vehicle |
| WMS | Warehouse Management System |
| MES | Manufacturing Execution System |
| AI | Artificial Intelligence |
| ML | Machine Learning |
| JS | JavaScript |
| IoT | Internet of Things |
| IIoT | Industrial Internet of Things |
| UC | Use Case |
| Cobot | Collaborative Robot |
| SPA | Single Page Application |
| SSE | Server Sent Events |
| MMI | Man Machine Interface |
| OT | Operator Terminal |
| GUI | Graphical User Interface |
| RFID | Radio-Frequency Identification |
| PLC | Programming Logic Controller |
| PLM | Product Lifecycle Management |
| EDF | Électricité de France S.A |
| MCU | Micro Controller Unit |
| HTML | Hyper Text Markup Language |
| CSS | Cascading Style Sheets |
| OPC UA | Open Platform Communications Unified Architecture |

# Introduction

Over the past, our society has taken part to important transformations from the technological point of view.
New technologies have been created and old ones have been modernized in a continuous process that brought our society where it is today.

The Digitalization can be considered as one of the mainstays of this process, bringing companies to stand out by becoming leaders in new sectors, and others to collapse and one of the most important aspects of those that survived was their ability in reacting to the changes that the competition brings.
But the Digitalization can be seen as a portion of a wider trend: Industry 4.0.
The "Fourth Industrial Revolution", an important milestone in the technology evolution has combined advanced manufacturing techniques with the IoT, in order to be able to use analyses and information to take important decisions in the digital manufacturing environment and basically forcing companies to rethink about their organization.
This trend relies on the employment of important technologies like Cloud computing, IoT (Internet of Things), Big Data, Artificial Intelligence (AI) with the aim of creating the so called smart factories. This growing process has not stopped and in the last 15 years a new concept has been introduced: the Digital Twin, one of the most promising technologies for smart manufacturing.

But what is really a Digital Twin?
The term Digital Twin (DT) appears for the first time in a publication by the National Aeronautics and Space Administration (NASA) in 2010. This praises it as: "An ultra-realistic, highly scalable simulation, which uses the best available physical models, sensor data and historical data to mirror one or more real systems".
The DT can be seen as a connected sets of simulation systems and models capable of exploiting data to obtain an effective response to the business requirements.
One of the main reason for which the DT is now reaching its peak, is that it can rely on key enabling technologies with a level of maturity able to support the main companies' processes. At first the cost in building digital twins was one of the main

reasons of their limited use, but today the situation is changing rapidly and their cost has almost been knocked down.
Those technologies include availability of robust, high speed wired and wireless networks, reliable sensors, data storage and computer power.

In order to interact with a simulation, system or industrial process, it seems worthy to introduce the concept of Human Machine Interface (HMI). It can be seen as a middle layer able to make the final user able to interact with a system showing him relevant data, by providing the possibility to take action on the system by changing its status, always taking into account important parameters like cognitive load, tools and other factors deemed important by the companies.

This work of thesis has been drawn up during a collaboration with Reply S.p.A. for two main projects:

- ICOSAF
- DiPreTreat

The Use Cases [16] proposed by CRF are reported in table 1.

| **ID** | **Title** | **Partner** | **Headquarter** | **Brief Description** |
|---|---|---|---|---|
| UC-A | *"Postazione di assemblaggio alberi controrotanti"* | CRF | CRF Melfi | *"Postazione con robot collaborativi integrata su due linee basate su AGV in sostituzione dei conveyor"* |
| UC-B | *"Assemblaggio collaborativo di inserti su scocca CFC"* | Adler | Adler Airola | *"Montaggio e incollaggio di inserti metallici sulla scocca in carbonio"* |
| UC-C | *"AMR (Autonomous Mobile Robots) collaborativi per logistica flessibile"* | CRF | CRF Melfi | *"Postazione di magazzino logistico in-plant ottimizzato per automazione flessibile."* |
| UC-D | *"Qualità Proattiva su punti di saldatura RSW tramite analisi "Big Data" e postazioni collaborative di controllo"* | CRF | CRF Melfi | *"Sistema di controllo della saldatura a punti basato su una prima selezione dei punti a "rischio" basato su un'analisi con approccio "Big Data" dei dati dalle pinze di saldatura unito con un dispositivo di controllo automatico-collaborativo della qualità dei punti di RSW."* |

Table 1: Use Cases

In particular for ICOSAF (Integrated COllaborative Systems for smArt Factory) the focus was towards technologies and systems for a collaborative factory with increasing operator integration according to the concepts of industry 4.0 (interconnected automation) and industry 5.0 (humanization and reuse of resources). With respect to Industry 4.0, that introduced robots, in Industry 5.0 the main actors are cobots. One of the main difference between the two actors is that they are able to work together with humans in shared areas, in a secure and efficient way.
Examples of this collaborative robots are Franka, Kuka (robotic arms), AGVs (Automated Guided Vehicles), smart autonomous robots capable of coping with coordinated operations. One of the focuses of ICOSAF is the operator enhancement, indeed in order to exploit the workers intelligence and flexibility in tasks where the human factor is more important, cobots are used in collaborative spaces for low added value tasks.
The activity conducted is about the development of a digital twin in a manufacturing environment and the development of an HMI interacting with it, following the principles just presented. In particular, the work will talk about Use Case A, both from the simulation point of view and from the interaction with the HMI.

Regarding DiPreTreat's project instead, it is developed in collaboration of:

- Santer Reply
- U Trento
- PoliTo
- Pipple

It is a project that can be classified in the area of Innovation and Research and in the CTS (Digital cities) segment.
The DiPreTreat's project whose name stands for Diagnosis, Prognosis and Treatment of technological pathology affecting European cities, consists in building a functional and robust software platform able to address a problem that afflicts concrete buildings, in particular the corrosion of reinforcing steel, which can be linked to:

- carbonation (reaction of CO2 in the environment with calcium hydroxide in cement paste)
- presence of sulphates and aggressive agents present in the soil
- presence of chloride due to sea water and/or acid rain
- degradation due to freeze-thaw cycles

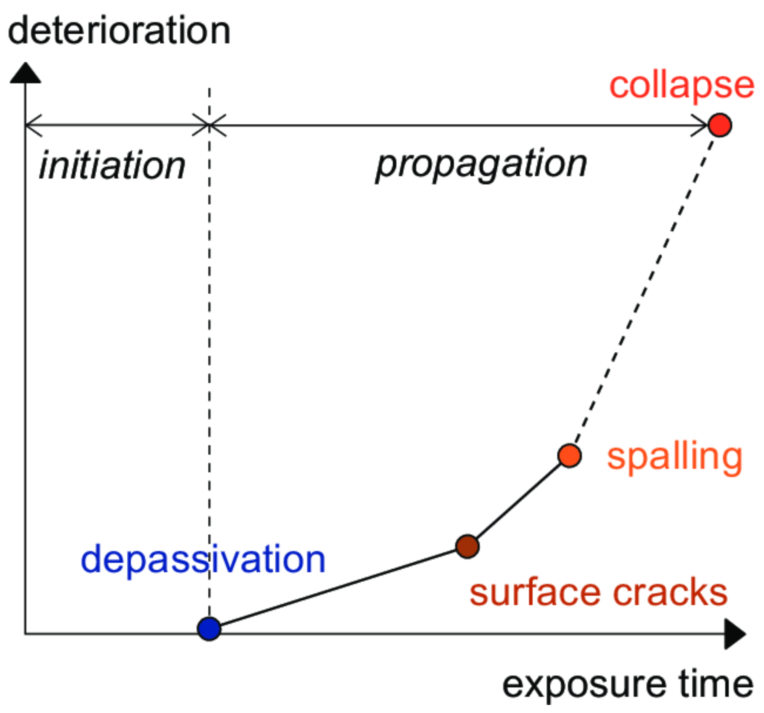


Figure 1: Deterioration of reinforcing steel with respect to exposure

As can be seen in the picture 1, it is a problem where the propagation at the beginning is steady but it increases with time.
For this reason the most timely are the corrective actions and maintenance, the lower will be the risk.
To inspect whether structures should be subject to maintenance manual inspections are conducted. But after the catastrophic collapse of the bridge of Genoa in August 2018 sparked research into the conditions of concrete structures highlighted the neglected state of many bridges.
AI-based algorithms have potential to help in the necessary support of preventive maintenance of such concrete bridges and other structures.
The focus in DiPreTreat's project solution is to develop a platform consisting of a network infrastructure which allow the view of long term measurements on the concrete deterioration parameters.
Thanks to a software application the aims are: to have the possibility of accessing the information gathered from the sensors installed in the building, to produce a data presentation able to give a timely insight on the problem and furthermore to have the possibility to process this data with AI-based algorithms with the aim of supporting corrosion issues predictive maintenance and forecast.
In order to fulfil this requirements, the most flexible and effective solution is to develop an internet application able to be used from any user.

# Purpose

## ICOSAF

The ICOSAF project is related to the integration between an HMI and a Digital Twin. In particular the idea is to use the potential of a Digital Twin to produce meaningful insight, to make predictions and to optimize the process described in the UC-A.
The aim is to create a system able to communicate with cobots, a Warehouse Management System (WMS) and a Management Execution System (MES) in order to fulfil the demands of CRF(Centro Ricerche Fiat) and of the partners involved in the project.
One of the first activity that was performed was the virtualization of the entire process with a specific simulation tool, for then being able to interact with it as it was the real one.
In order to face the requirements and the demands of CRF and of the other partners, the Digital Twin that was modeled, had specific parameters.
In particular some of them were:

- Speed of the cobot KUKA
- Speed of the cobot FRANKA
- Speed of the AGVs
- Speed of the operator (defined statistically from real workers)
- Picking time of FRANKA
- Picking time of KUKA

Also other relevant parameters more specifically related to the simulation of the process itself, were defined in order to be as compliant as possible to the demands:

the number of actors involved in the UC, their positions, their movements and actions, their behavior in case of error and so on.
Some of them can be modified by the user while launching the simulation in order to test and see, by having different simulation parameters, slightly different behaviours with respect to the standard one.
Via the HMI, the remote operator has the possibility to control the system.
Indeed, the HMI realized for the project was an Internet Application able to control the entire process flow of the system allowing a bidirectional communication between the virtualized process and the human machine interface itself and it was developed taking into consideration the demands of the partners involved in the project.

## DiPreTreat

The DiPreTreat project, as said, is aiming in monitoring one of the problems that afflicts buildings: the corrosion of reinforcing steel.
In order to collect relevant data useful for monitoring crucial measures related to this phenomenon an infrastructure of sensors was installed.
In such a way, it was possible to have an insight on the problem and to perform preventive maintenance of the building. To fulfil this need of gathering data, of visualizing them, and of using them to produce a reliable prevision, after a conscious cost benefit analysis, a web based software application was built. The reason for peaking this solution instead of others was because it is one of the most flexible one able to satisfy the needs with the minimum cost.

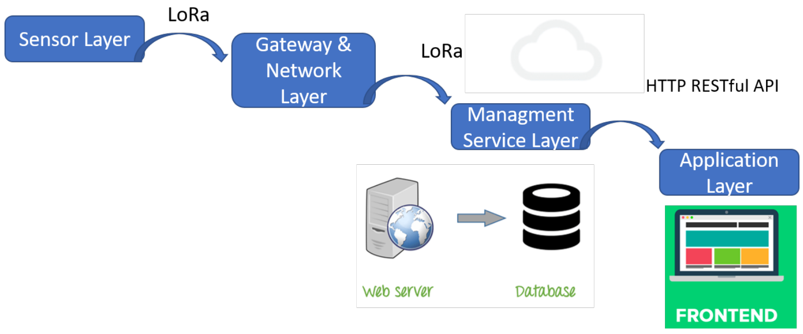


Figure 2: High level architecture of DiPreTreat platform

In particular, as can be seen in figure 2, the application was divided in two parts: Server and Client.
Fitting to the modern architecture precepts and to the demands of the modern web

applications (e.g. SPA), more responsibilities are given to the client, which, thanks to the higher computational power can reduce the work of the server by performing some of its important historical tasks.
Nonetheless the server maintains some important duties like:

- Authentication
- Authorization
- Validation

It is responsible of gathering data from the sensors and since it was decided to use a Representational State Transfer (REST) style of software architecture, one other important duty was also to expose APIs in order to map the entities identified in the project.
For this reason the client has not anymore only the duty of rendering the HTML page received from the server and of performing some decoration (e.g. JS), but it has some other important tasks to take care, like:

- Business Logic
- HTML generation

In particular, for the aim of the project, the responsibility of the client was to produce an interface able to facilitate the user in visualizing data coming from the server, producing forecast requests then elaborated from external servers and to manage all the related business logic.

# Document Structure

The following work of thesis is organized in three different parts:

1. Part I:
   In the first part of the document the technologies and tools related to the Digital Twin and the web applications have been analysed from a theoretical point of view

2. Part II:
   In the second part of this work of thesis it is shown in details the implementation of the first stages of a Digital Twin with the Anylogic simulation tool and the development of both the HMI used to control the manufacturing process and the web service used to monitor the problem of the corrosion of the reinforcement steel

3. Part III:
   In the third part the results and the conclusions about the work produced are presented

# First Part

# Digital Twin

## Introduction

Since the last decade, the manufacturing industries have experienced an important swing with the introduction of digital technologies within their factories. This trend known as Industry 4.0 relies on the employment of IT with the aim of creating the so called smart factories.
The mainstays of this trend were the Internet of Things (IoT), Big Data, Cloud Computing and the Artificial Intelligence (AI), but in the recent years, the technological innovation has led, with a new enthusiastic perception around the Industry 4.0 concept, to a reshape of its pillars.
In particular within this innovative tendency, attention has been driven towards to the creation of a more linked and tied synergy between the physical world and the virtual world.

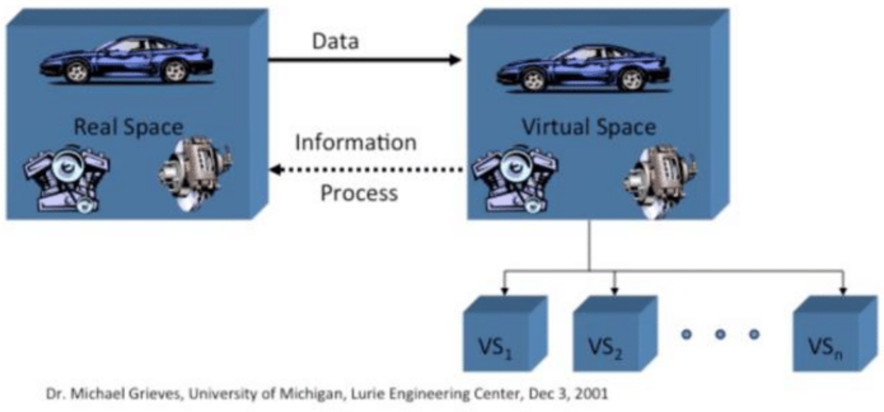

Data
Real Space
Information
Process
Virtual Space
VS$_1$
VS$_2$
VS$_n$
Dr. Michael Grieves, University of Michigan, Lurie Engineering Center, Dec 3, 2001

Figure 3: Model of "Conceptual Ideal for PLM" [15]

This interest comes from the idea that by having a strict connection between those two worlds, we may have eventually a tool able to support the decision making at different levels, from the early and abstract strategy to the more concrete tactical or operational decisions.

As shown in the presentation of industry in 2002 for the formation of a Product Lifecycle Management (PLM) center, this new paradigm has been decomposed in its fundamentals:

- Virtual Space
- Physical Space
- Bidirectional data flow among the two spaces

As can be seen in the picture 3 the idea behind the DT is that each system consists of two sub-systems: the always existed physical system and a new one, the virtual system that contains all of the information about the former.
This leads to an important aspect: the mirroring or twinning of the two systems between what exists in the real world and what exists in the virtual space and vice versa.

## Objectives

The Digital Twin strategy can be applied with different objectives and in different phases of a project. Well known companies like GE, Siemens, Shell and others had begun to make use of it with maintenance purposes, but this is not the only area where DT can be applied. Indeed there are other possible areas where the usage of DT strategy can produce a positive impact to the business of a company:

- Experimental: the Digital Twin technology is exploited in order to make tests and to validate possible alternative solutions.
  Since a DT perfectly reproduces the properties of a real system, and the difference in costs in developing and testing is significantly lower with respect to implement a real solution that potentially is inefficient, implementing a DT can be useful in order to minimize costs and optimize business performances and moreover the results obtained by the digital prototype can be used to improve real systems.
  Industries find it very useful to have cost saving and to reduce the Time-To-Market.

- Prediction: the processes need to be performing, efficient, controlled and fast. Since a DT fundamentally is an evolving digital profile based on a large amount of data collected from sensors, it can be used for a prediction purpose.
  Indeed predictive analysis while the system is running helps in trying to predict the future state and behaviour of the system and this may eventually lead to better performances and an higher profit margin

- Security: constant control and monitoring of the system that guarantees the security of the system itself and of the surrounding environment

- Diagnosis: a continuous analysis of the system to detect potential perturbation of behaviours and state of the processes

## Examples of usage

By providing a real-time insight of what is happening with the surrounding environment, in terms of equipment and/or other physical assets, DTs are a key strategy in supporting production, reducing maintenance problems and ensuring product and process optimization. But it is not the only area where DT can bring added value:

- Manufacturing: Fameccanica, one of the main world players in the industry of machines for the production of diapers and sanitary napkins, has equipped itself with a virtual room in which it can reproduce, in real size and in 3D, its machinery for better product development but also to train maintenance workers [23].

- Smart cities: in this sector the most famous example of a digital twin on a large scale can be identified in a project developed in Singapore. Virtual Singapore offers 3D semantic modeling of the city, where data related to insights of public transport, building components and infrastructure, electricity consumption as well as other information like demography, climate and traffic are associated to the model with the aim of creating a Digital Twin useful both in the phases of sustainable urban planning and development and moreover in emergency simulation and structuring of evacuation protocols.[8]

- Healthcare: digital twins can potentially revolutionize both healthcare operations and patient care. By using data about patients, digital twins can reproduce with a three-dimensional representation of organs, bones, venous, lymphatic, and nervous systems, that can allow surgeons and healthcare professionals to experience the procedures in a simulated environment rather than on a real patient.
  Bandage size sensors can monitor patients and produce digital models that

can be monitored by Artificial Intelligence and used to improve health care assistance.[23]

- Sport: in sports where every second counts, using simulation can help teams to know which adjustments are necessary to improve performance. This is the case of Formula 1 where digital twins are used to refine motor racing.[23]
- Energy: the EDF group, main player in France in the energy sector also present in Italy, announced in March 2019 that it plans to use digital twin of nuclear power plant turbos. The aim is to optimize plant performance, allow predictive maintenance and reduce costs of repair.[8]
- Farming: The Brazilian tractor manufacturer Stara, thanks to Iot sensors installed on the equipment and the performance monitoring, provides farmers of a real time insight of the conditions of plants, and identifies the best conditions to plant crops and improve yield, with an optimization in use of seeds and fertilizers.[8]

## History

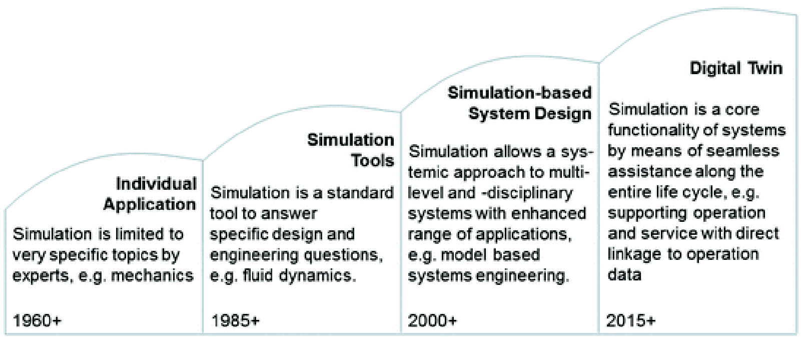

Individual Application
Simulation is limited to very specific topics by experts, e.g. mechanics
1960+
Simulation Tools
Simulation is a standard tool to answer specific design and engineering questions, e.g. fluid dynamics.
1985+
Simulation-based System Design
Simulation allows a systemic approach to multi-level and -disciplinary systems with enhanced range of applications, e.g. model based systems engineering.
2000+
Digital Twin
Simulation is a core functionality of systems by means of seamless assistance along the entire life cycle, e.g. supporting operation and service with direct linkage to operation data
2015+

Figure 4: Digital Twin Evolution [6]

By taking a look at the picture above, the Digital twin, as it is known today, is the last and most recent link in the evolutionary chain.

Throughout the years this concept has grown and its origin can be tracked all the way back in the sixties.
At that time one of the main American agencies, the NASA, was using simulation for reproducing significant situations that could have happened on the astronauts while in orbit. In particular are relevant the circumstances around the Apollo 13's journey where a group of astronauts were saved thanks to the power of various simulators connected through a digital computer network.

By itself, a simulator is not a digital twin, but what makes it different and make it, to be classified as one of the first usage of DT is the ability by the NASA to have complete control over the simulation to rapidly switch and adapt the context to match conditions on the real-life spacecraft.
They did this by creating physically duplicated systems matching the systems in orbit. In such a way, it was possible to reject many possible but not efficient solutions and to identify the perfect strategy required to bring the astronauts home.[10]

In the eighties simulation tools, thanks to a greater awareness of the power of the simulations, specific tools started to be used in order to answer specific engineering problems.

In the 2000s thanks to the strong advancement of technologies, simulation was used as a systemic approach, not to solve only specific engineering problems but also multi-level, multi-disciplinary and wide open problems.

Around 2015 simulation becomes the core of many industries. It basically started to provide assistance, at different levels, to each stage of the industrial process life-cycle, by means of a more tight and direct linkage to data.

Nowadays, a digital twin is an appealing solution to many companies.
It provides, with a relative lower cost with respect to its dawn, a way to build a virtual replica of physical, potential and actual resources equivalent to objects, processes, people, places, infrastructures, systems and devices. [23]

## Data

For what concerns the Digital Twin technology, it can be assured with absolutely certainty that the part related to the data collection and analysis is one of the most important pillars in its construction.

Data is so important in the development of a Digital Twin that without this key component no digital twin can be built.
Indeed a digital twin can be seen as a composition of statistics, information, insights or in general data of an already existing process, resource or entity in the physical world.
In order to have a meaningful representation, it is mandatory to have representative data of the process or resource to be analyzed and they should be as unbiased as possible.
If the collected data are not meaningful, it would lead to wrong insights and models

not able to produce a trustworthy prevision.

Collecting data is a without ending task that needs to be always performed in order to have an up-to-date representation model.
Since the last fifty years, thanks to the spread of electronic devices and interconnection among them, data proliferation is continuously increasing.

A lot of data has been collected for the more disparate contexts of usage and each day data are even bigger than ever before.
Thanks to the growing volumes of data, the dimension of statistical and simulated models has acquired more relevance and solidity.

Data collected gathering information by the multitude installed sensors are then used in building a Digital Twin that can be useful in many contexts.
The reason why digital twins are so closely related to the Internet of Things (IoT) and Industry 4.0 is that they make use of data coming from the same infrastructure, connected devices and sensors, for purposes like:

- Report operational data representing the state of the process
- Processing and analytics
- Process optimization
- Forecast analysis

The extraction process of data has become more affordable by the companies, because the key enabling technologies like cloud computing, reliable sensors, data storage and robust networks have reached an higher level of maturity.

Through the big data analysis based on cloud computing, companies could find the bottlenecks in processes, realize the causes and impacts of the problems, and find effective solutions.
A large volume of structured, semi-structured and unstructured data generated from the product life cycle are collected in real-time automatically by the IoT infrastructure.
The aim of gathering all this information is to exploit them in improving the efficiency of the processes in order to make companies leaner and more competitive.

## Architecture

Digital Twin is a digital representation model of assets, such as a physical or virtual resource, a process or service.
A DT can collect data from the sensors and their environment and it reflects its status in its own virtual domain. This status could be visualized just to have a deeper insight, or it could be stored for further analysis (analytics, artificial intelligence). Furthermore, it can be also used for providing a sort of alerting system able to warn operators when something is going out of the typical controlled scenario.
The functions supported by a typical Digital Twin architecture are:

- Data collection
- Data abstraction
- Data enrichment
- Modelling
- Analysis
- Visualization / Actuation

Digital Twin should include all the functionalities and should model what the particular asset can perform.
An important aspect is that the virtual representation created is not a static model with fixed and unchangeable characteristics. But it represents an asset in continuous evolution where services can be continuously expanded, as the real asset changes.
One of the aspects of a digital twin can be identified by the data collection.
As said in the previous section it is one of the most important pillars and this activity is usually performed by a sensing layer, characterized by a multitude of sensors. These could be any real physical sensors such as UWB, which are used for precise real time location of the goods inside the warehouse or inside the manufacturing plant or RFID tags usually used for tracing the throughput of the manufacturing line, a QR/Bar code reader or virtual data source.
Furthermore, there is the abstraction layer which can make transparent the origin of the data sources, providing a bridge or gateway to make all data sources indiscernible for the system itself.
The enrichment layer is used as the name suggest to enrich data, by creating new dimensions, aggregations of data, or elaboration of them in order to bring to light much more meaningful and useful aspects of the model considered.

The other import aspect is the modelling of the system. In order to create a virtual model, all the collected data first of all needs to be stored in a persistence layer.
By having this, any further elaboration, analysis or simply visualization of the model's characteristics will be possible and furthermore it will be created a representative model that can be used in many contexts: for simulation, forecasting, optimization and so on.
It's the business analytics layer that takes as input the built model and takes advantage of it by providing insights on the data simulation, provides predictions based on the received data and could be even used to try to optimize the processes analyzed.
Last but not least, there is the presentation layer.
In order to take a look on what is happening and to interact with the model that has been built, the construction of an HMI is mandatory. This can have several characteristics and can be built with many different technologies, but it has the role of a glue between the model and the physical world.

## Creation Process

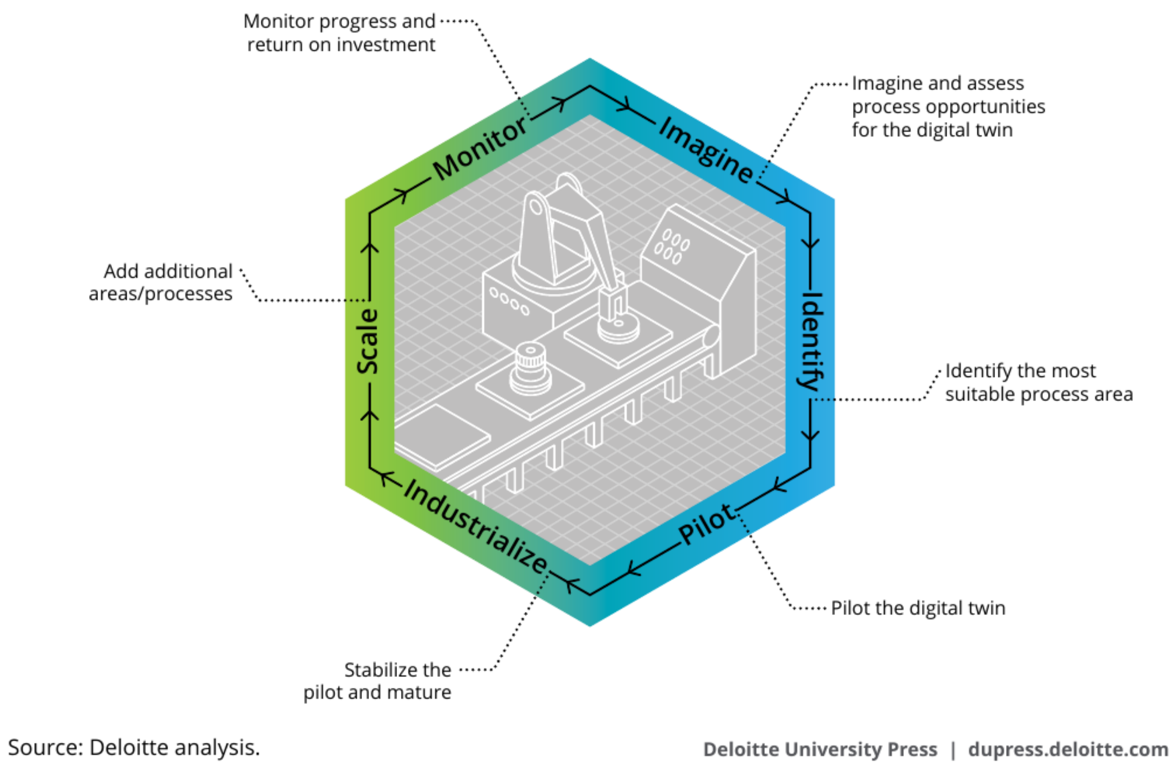

Monitor progress and return on investment
Imagine and assess process opportunities for the digital twin
Add additional areas/processes
Identify the most suitable process area
Pilot the digital twin
Stabilize the pilot and mature
Monitor
Imagine
Identify
Pilot
Industrialize
Scale
Source: Deloitte analysis.
Deloitte University Press | dupress.deloitte.com

Figure 5: An overview of getting started with the digital twin [1]

Developing a model of a real-world system involves multiple components and can quickly become complex.
A significant challenge in taking on a digital twin process can be identified in determining which is the ideal level of complexity and details in creating a digital twin model.
Models can span multiple domains and the number of statistics coming from millions of sensors and signals can become rapidly unmanageable, but at the same time a too simplistic representation of the problem could yield to false promises.
For this reason, an equilibrium between the two approaches must be found and it's not always an easy task.
A possible approach that tries to identify the right level of complexity is the one described in the picture 5, where are identified substantially six phases:

- Imagine
- Identify

- Pilot
- Industrialize
- Scale
- Monitor

Diving into the details, the first step is to try to identify which scenarios, processes, resource could benefit from the usage of the Digital Twin technology.
In order to be valuable, it should be a relevant scenario or process for the business of the company where there is still high margin of improvement.
The following step is to identify the process: in order to have a successful and trustworthy digital twin, the identification of which process to consider is essential. There can be the possibility of choosing of going into deep, complex configuration of a digital twin for a specific part of a process or it can be identified in a more general and flexible configuration where a wider look to what is happening is considered.
The next step is the creation of a pilot program. The pilot can be the representation of a particular use case, an entire business division and by using it in iterative and agile cycles, there is the possibility to learn more rapidly about the process itself and to maximize the return on the investment with the minimum risk.
Its aim is basically to try to show if further analysis or optimization can be worthy or not. Once the process has been defined and the pilot program has shown it value, the succeeding step is to industrialize the digital twin with established tools and techniques. Once it is deployed, if successful, it can be taken into consideration the possibility of scaling this solution, targeting similar processes or interconnected ones. The last step in the creation process of a Digital Twin is to monitor and take measures about the implemented solution.
This part is very important because the construction of a DT is a continuous process where always new analysis can be performed, and new configuration can be tried out.

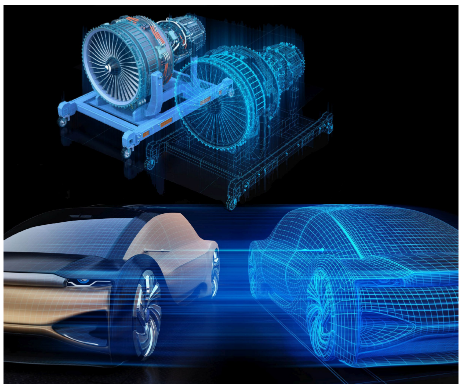


Figure 6: Digital Twin [13]

## Hardware and Software components

The *"Thing"* in the Internet of Things is the starting point for an IoT solution. It is typically the originator of the data and interacts with the physical world. "Things" are often constrained in terms of size or power supply, therefore they are often programmed using micro-controllers (MCU) that have very limited capabilities. The micro-controllers powering IoT devices are specialized for a specific task and are designed for mass production.
Hardware used in most IoT systems includes devices for a remote access and control, servers and sensors.
These devices manage key tasks which enable the systems in performing and supporting specific actions and goals.
Possible functionalities are system activation, action specifications, security, data communication and detection.

Regarding the ICOSAF project, some of these devices have been or are planned to be used.
Since in the use case there are two main actors the remote operator and the field operator two different categories of devices have been considered according to their tasks and role regarding remote access and control.
In particular for the remote operator, since devices must be user friendly and with a low cognitive load, the ones that have been considered were devices belonging to the category of PCs and Tablets.

The choice has fallen on those devices because of their familiarity, the type of visualization provided and the kind of application that the operator needs to use in the work context. Indeed, since the remote operator needs to use devices for using an internet application nothing is more preferable than the solutions proposed.
Regarding the field operator among the category of devices that have been considered the selection has fallen on the smartwatches.
This is due to the fact that the role of the field operator is more pragmatic and he needs a non-cumbersome device in order to fulfil his work.

Whether the remote operator needs to use an internet application, the field operator will need to use a hybrid application able to work both on Android and IOS. The reason of the choice stands in the fact that an internet application on such small devices are uncomfortable to use and are not as efficient as an integrated application already installed in those devices.

Taking into consideration another "thing", the server is for sure one of the most important element in the ICOSAF architecture.

It has been deployed in the Reply private cloud and it exposes a RESTful API that can be contacted both in the VPN and outside of it.
Its role can be synthesized in providing authentication, authorization and validation, and its implementation was written in nodeJS.
But these are not the only tasks, indeed the server in order to provide data persistence, it needs to interact with a proper DB.
In particular since in the use case, the data type that needs to be used are structured, the selection has fallen on an SQL database.
It's by exposing some APIs that the server gives the possibility to the client applications to collect, present and manage data in the proper way.
Data are exchanged based on the standard HTTP/IP communication protocol, and through the provided API it is possible to ask the server for specific resources and for specific actions to be performed.
The client application is instead responsible of managing the visualization, the rendering of the pages and the business logic.
The interaction with the client is depicted in the following picture:

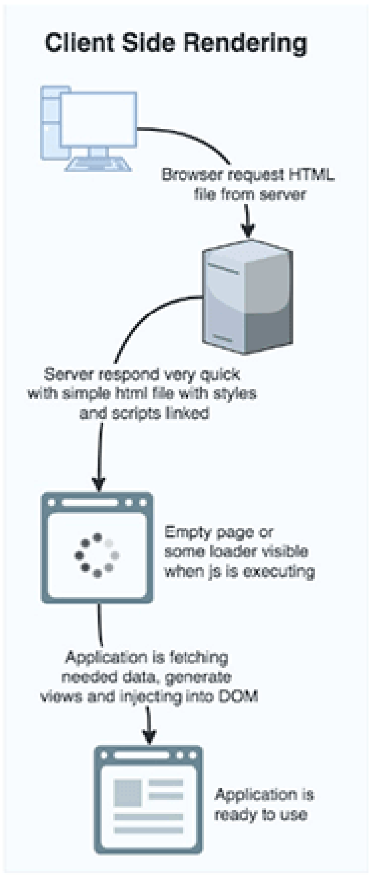

Client Side Rendering
Browser request HTML file from server
Server respond very quick with simple html file with styles and scripts linked
Empty page or some loader visible when js is executing
Application is fetching needed data, generate views and injecting into DOM
Application is ready to use

Figure 7: Overview of possible interaction for a client web application [9]

As can be seen from the picture 7 since a fat client solution has been chosen the server has been stripped of important roles.
Indeed the server via the APIs provide the client of the necessary data but leaves to it the management of the rendering of the page and of the business logic.
The client side rendering was implemented by creating an HMI through the Angular framework that provides all the necessary tools.
Regarding the sensors instead, they have been completely simulated via a software simulation tool.
In particular after a careful analysis, the tool that has been chosen was Anylogic.
Via this software platform, it was possible to simulate the entire use case and the sensors were implicitly defined by the behaviour of the actors involved in the use case itself. In fact, actors are able to correctly position to stop points, perform checks, avoid obstacles and perform all those tasks that would require a physical sensor or a machine vision camera to identify impediments.

## Use Case A

The use case that has been considered in this work of thesis is the Use Case A.

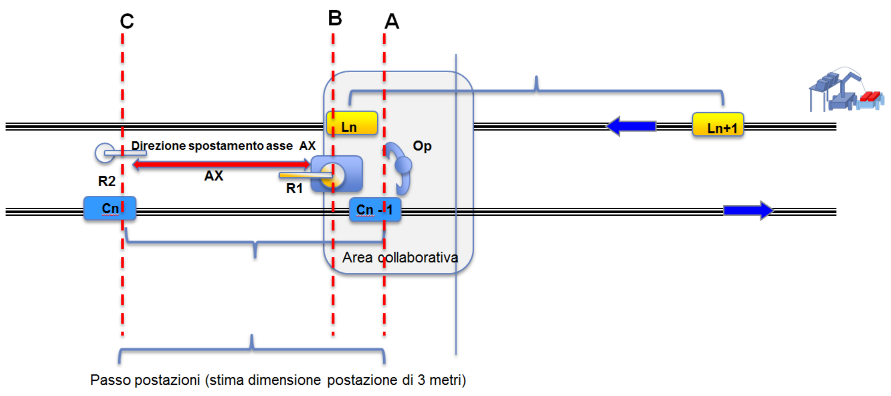

C
B
A
Ln
Ln+1
Direzione spostamento asse AX
AX
R2
R1
Op
Cn
Cn -1
Area collaborativa
Passo postazioni (stima dimensione postazione di 3 metri)

Figure 8: Use Case A architecture

This use case involves the setting up of a workstation with a collaborative robot (R1) on a linear slide for the assembly of two counter-rotating shafts (exhaust and intake) on the engine base (one of which in continuous moving) and a workstation with an operator for assembling the oil pump; the R1 robot operates between two AGV lines: an in-bound logistics line (Kit Line) for the transport of the components to be assembled on the engine (the counter-rotating shafts, the oil pump and the necessary small parts) and a main line (Line Motor) for transporting the motor base in question.
The operator performs the dual task of inspecting the counter-rotating shafts supplied through the kit line and then the picking up and mounting of the oil pump on the engine line.
A collaborative space sharing area is thus created between the operator and the R1 robot.A second collaborative robot (R2) has the task of making the two shafts available to R1 in order to create a decoupling between the logistic line (the kit transport AGV) and the assembly robot (production line) and at the same time to guarantee the two collaborative robots reachable for picking and placing operations towards the AGVs.
In particular:

- Ln and Ln + 1 represent the kit transport AGVs and their movement on the Kit Line;
- Cn and Cn-1 represent the AGVs transporting the engine block and they move on the Engine Line with the opposite direction to the logistics AGVs;

- Position A: the AGV performs a first stop allowing the operator to examine the contents of the transported kit (Ln) and to take the oil pump and the screw for subsequent assembly on the Cn-1 AGV;
- Position B: robot R1 finishes the assembly operation of the second shaft and returns to position C;
- Position C: the AGV Ln makes a second stop to allow the R2 robot to pick the two transported shafts. The two shafts passed directly from Robot R1 to Robot R2 through an optimal gripping position; The first shaft is assembled on the motor base with AGV stopped, the second shaft is assembled in continuous moving (AX direction, Robot R1 slides and AGV motor in motion).
- The R1 robot moves on the AX slide (in synchronous with the AGV Cn) from position C to position B (during the assembly phase of the 2nd shaft) and vice versa (for the picking preparation phase).

In order to verify the feasibility of the assembly process thus defined at the concept design level, a simulation in a 3D environment was carried out using the Anylogic simulation tool. The simulation made it possible to observe specifications the feasibility of the process, considering proximity and collisions of the inserted elements; it also represented a first tool for analyzing the times and sequences of the analyzed process.

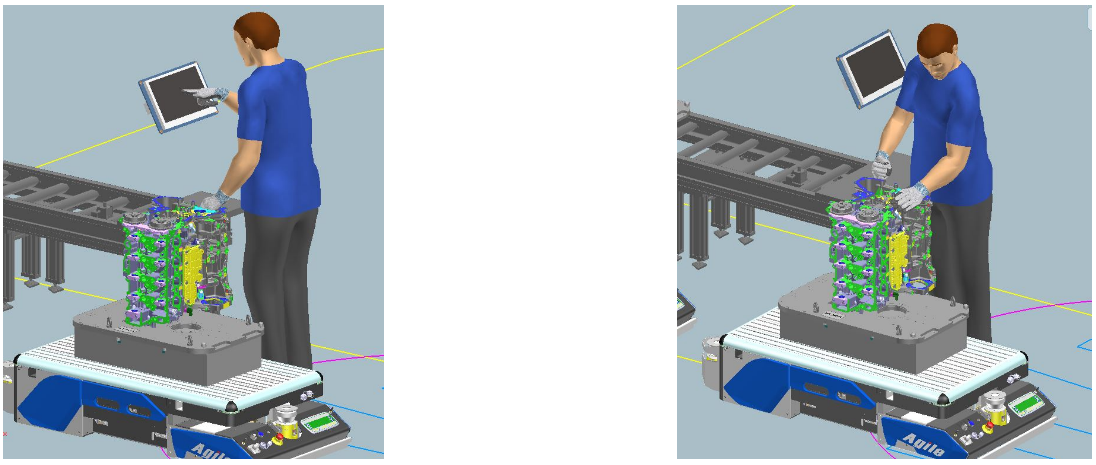

(a) Interaction with the HMI (b) Components assembly

Figure 9: Examples of operator activities

For the use case A, the overall system architecture has been defined thanks to the collaboration and the exchange of information with the other companies and

this collaboration characterized also the realization of other goals, like the ones concerning the human-robot and robot-robot cooperation, and the ones related to the definition of the human machine interface for the interaction with the collaborative systems.
After several analysis that includes functional, method and safety on the use case an overview of the UC-A architecture has been defined, making explicit the involved actors and the transport sled.
The overall architecture is depicted in the following picture:

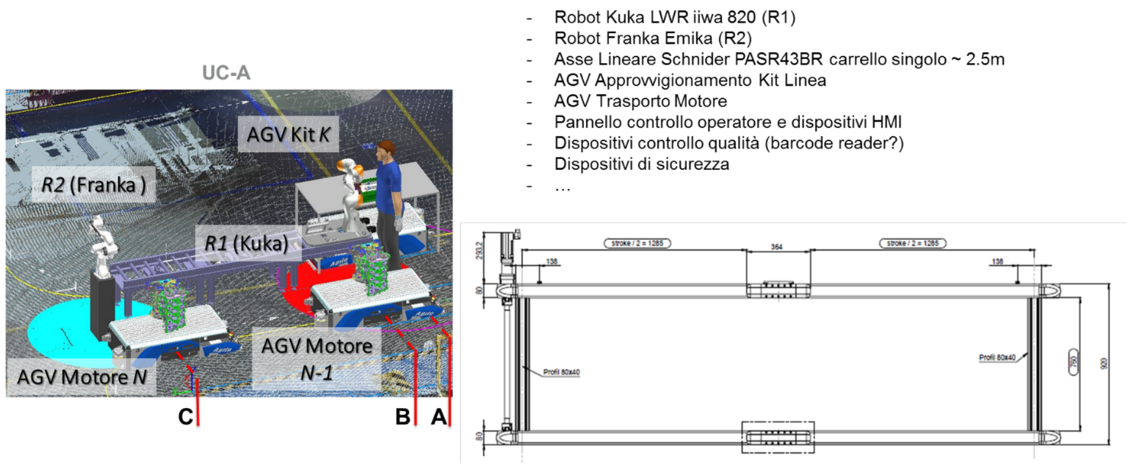

- Robot Kuka LWR iiwa 820 (R1)
- Robot Franka Emika (R2)
- Asse Lineare Schnider PASR43BR carrello singolo ~ 2.5m
- AGV Approvvigionamento Kit Linea
- AGV Trasporto Motore
- Pannello controllo operatore e dispositivi HMI
- Dispositivi controllo qualità (barcode reader?)
- Dispositivi di sicurezza
- …

Figure 10: Use Case A Architecture - Actors involved, transport sled R1

As it can be seen in the picture 10 the actors involved in the use case are:

- Cobot Kuka LWR iiwa 820
- Cobot Franka Emika Panda
- AGV transporting the Kit with relative support trolley for the kitting box
- AGV responsible of transporting the engine block

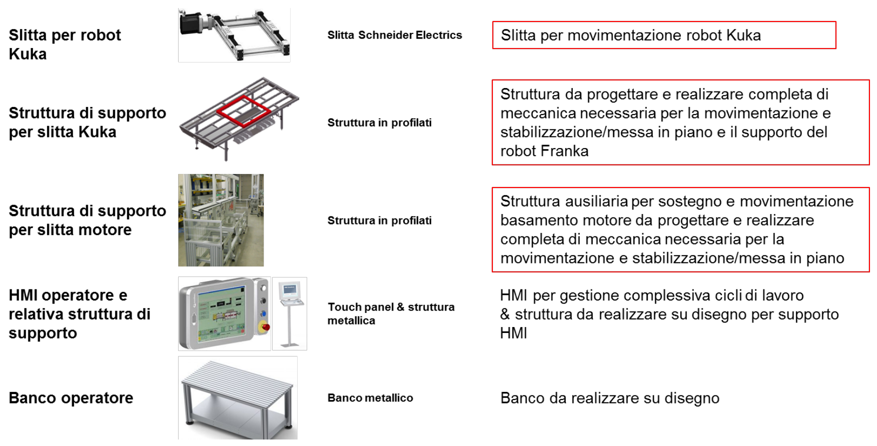
(a) Cobot Franka Emika Panda [12]

(b) Cobot Kuka LWR iiwa 820 [17]

Figure 11: Actors involved in the use case A

(a) Example of AGV [2]

The operator also uses other devices for executing his tasks. In particular it was considered the presence of a control panel and an HMI in order to represent what is happening on the assembly line.
The following figure 13 shows some specific components and devices of the UC-A, currently not available in the CRF laboratories, which will need to be procured for the continuation of the Industrial Research and Experimental Development activities of the ICOSAF project. For devices and components not shown in the figure (e.g. Robot R1 and Robot R2) it is assumed to use those that are easy to acquire already available, because they are part of the laboratory and equipment.



| | | | |
|---|---|---|---|
| Slitta per robot Kuka | | Slitta Schneider Electrics | Slitta per movimentazione robot Kuka |
| Struttura di supporto per slitta Kuka | | Struttura in profilati | Struttura da progettare e realizzare completa di meccanica necessaria per la movimentazione e stabilizzazione/messa in piano e il supporto del robot Franka |
| Struttura di supporto per slitta motore | | Struttura in profilati | Struttura ausiliaria per sostegno e movimentazione basamento motore da progettare e realizzare completa di meccanica necessaria per la movimentazione e stabilizzazione/messa in piano |
| HMI operatore e relativa struttura di supporto | | Touch panel & struttura metallica | HMI per gestione complessiva cicli di lavoro & struttura da realizzare su disegno per supporto HMI |
| Banco operatore | | Banco metallico | Banco da realizzare su disegno |

Figure 13: High level BOM - Use Case A

The support structures and the movement sled of the R1 robot are highlighted in the figure, because they are more critical, given that their unavailability, if prolonged over time, would preclude the initial development and testing activities of the UC-A.
In the thesis work all of those elements have been simulated with the tool Anylogic, indeed by defining structures and actors it was possible to realize all the depicted architecture.
If those elements belonging to the architecture have been simulated, the communication among those was not.
The proposal of the overall architecture for the control of the UC-A systems is based on the Gigabit Ethernet network where data are exchanged with the standard client server protocols (via TCP / IP and UDP socket).
A Restful application was built where the server and the clients communicate via the specified protocols.
In particular, the server exposes some APIs and those are reached by the HMI that was decided to be created by considering one of the most recent developing technologies: the Angular framework.
By implementing the HMI with this framework it was possible to visualize and control the status of the line.
To create a communication between the remote and the field operator, it was decided to use a notification system that had to be raised whenever a particular error in the assembly of the shafts and the motor was raised.
Every change of the system status is reflected onto the DB by communicating it to the server via those APIs exposed by the server itself.
In this way, even more than two clients can be synchronized on the system and can communicate with each other.
The UC-A has a demonstrative purpose in the context of an industrial research in

the ICOSAF project, which is extremely relevant for the final customer.
The proposed solution represents a usable use case, even if not intended as immediately transferable to the plant, for the automation of an essential work operation: the assembly of counter-rotating shafts.
The architecture conceived for realizing all the required tasks in all the use cases are:

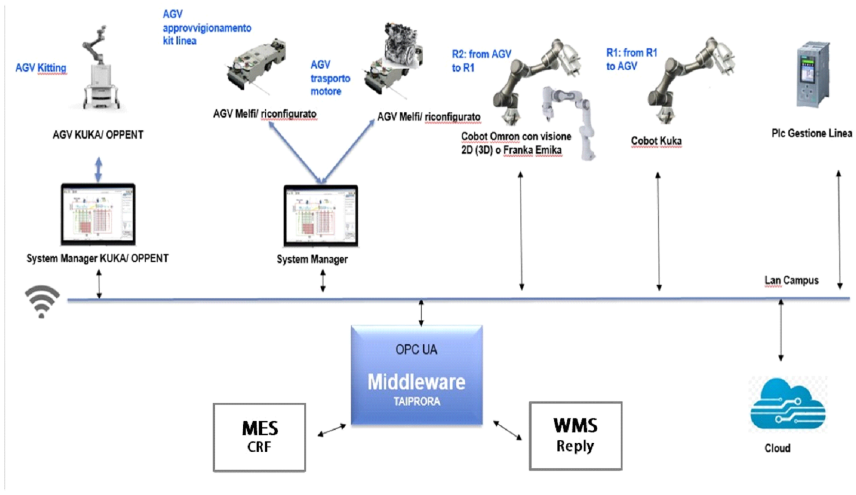


Figure 14: Use Case A-C Architecture

In the figure presented, it can be seen that there are many cobots involved in the system and they communicate on the LAN of the Campus with a middleware layer.
The middleware layer is a sort of bidirectional channel that translates and standardizes the communications between the smart factory and the devices and it communicates with two important proprietary softwares:

- MES
- WMS

These two system, the Manufacturing Executing System and the Warehouse Management System have two the key responsibilities. The first has to take care about the production orders that needs to be processed, so it has to define based on internal statistics which is the best production plan.
On the other hand the WMS has another important key role: as the name suggests it is responsible of caring about the state of the warehouse and its management. In particular, if a new order needs to be processed, it's task of the WMS to define the availability of the required components, their positions in the plant and which

AGV has to fulfil the particular order. These systems as said communicate with many cobots and devices inside the establishment, and many of those are reported in the upper side of the figure 14:

- Cobot Kuka
- Cobot Franka Emika Panda
- Kitting Agv
- Transport Engine Agv
- Agv with KUKA
- Programming Logic Controller
- System Managers

Cobots are different from the robots in particular for their scope of usage; indeed these are used in cooperative space where humans and cobots work together to perform a job.

Cobots take care of the low added value and repetitive tasks, while the operators can use their work time for more valuable duties.

The Automated Guided Vehicles are portable robots that, as the name suggests, can autonomously guide themselves in the plant based on the directives provided by the WMS. They are provided with sensors that collect data about the surrounding environment in order to avoid obstacles.
The PLC is used to send commands to devices which do not have a System Manager and its purpose is to translate high level commands into binary ones in order to be understood by robots.

Last but not least the system managers have an important task: through the HMI, it has to provide to the operator all the necessary tools in order to control and visualize the status of the system.
In particular it has to provide the possibility of controlling the AGVs based on their status and on the information received for instance through the OPC UA communication protocol, which can guarantee independent and secure communications. Every information and state is reflected in a persistence layer and the communication of this data is done towards the Reply private cloud where data will be further processed for analysis and diagnosis.

## Working Phases - Main scenario

The main scenario of this use case depicts a collaborative work area where an operator carries out a task in collaboration with several cobots. In particular the low added value tasks considered are carried out by the cobots involved in this scenario.
The actors involved are:

- Operator
- Kuka
- Franka
- Agv transporting shafts
- Agv engine transport

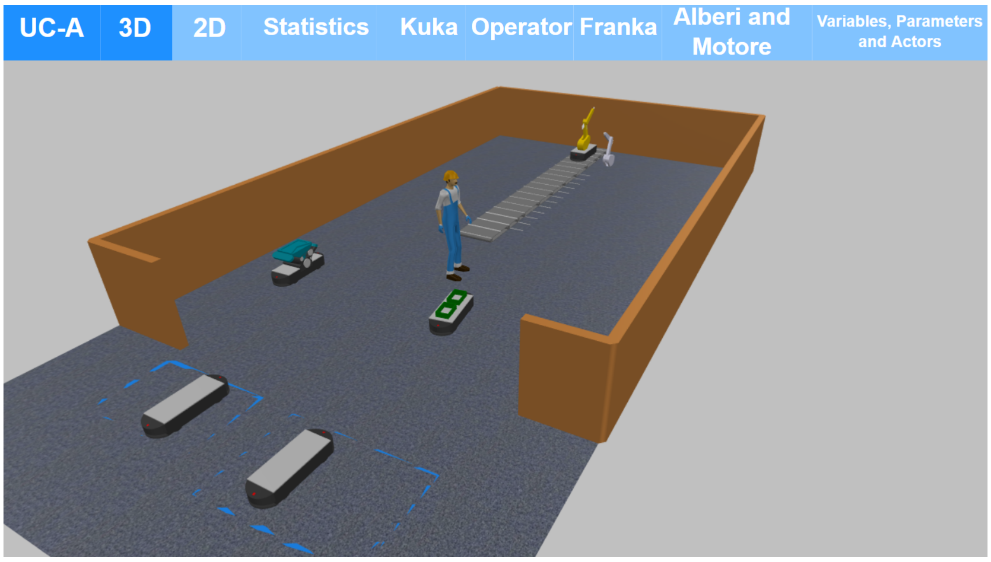


Figure 15: UC-A Overview

Whenever a new order needs to be processed, the agv transporting the shafts brings the shafts to the operator whom performs a check.
If the operator confirms that all has gone well then the agv brings the shafts to the cobot Franka which has the role of passing the shafts to Kuka. Once performed this task, the agv has to return back to the initial position in order to process the next incoming order.
In the meanwhile the agv transporting the engine goes to its stop point where the

cobot Kuka has to assemble the received shafts into the motor engine the first on place, instead the other in movement while returning to the operator.
After this process, Kuka returns to its start position while the agv transporting the motor proceeds its run towards the operator. Once it has arrived to the stop position for the check, the operator checks whether the assembly has been performed well or not. If the check performed tells that the engine has been correctly assembled then the agv goes back home releasing the installed motor and returning to process the next order.

## Working Phases - Error scenario

In this use case three main error scenarios have been pointed out.
Two are related to the checks performed by the operator: the first is related to the the check of the shafts, the second instead is related to the check performed on the motor engine when the two shafts are assembled.
In this two cases, a signal about the error is sent by the operator to notify about the new status of the process and the AGVs have to return back to the initial position, release the material and turn back to fulfil new incoming orders.
It's the field operator that communicates if some error was in place, by means of a user interface, who is in charge of controlling all the work areas. The third scenario of error instead is related to the errors signalled by the cobots of Franka and Kuka in performing their defined tasks.
A notification is sent to the remote operator who has the task of defining the resolutive actions to be performed in order to resume their activity.

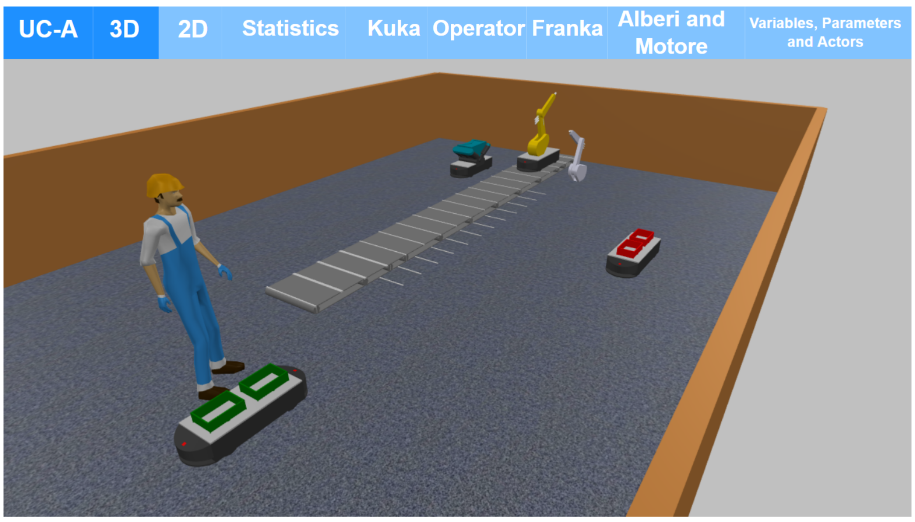


Figure 16: Example of error in the Shafts

## Simulation - State of the art

During the development phase two main simulation tools have been used:

- ROS
- Anylogic

Why was the simulation modeling used?
Because what a simulation does, is to try to solve real world problems in a safe and effective way.
It is able to provide a significant insight about the simulated context, where solutions can be easily tried without any risks, in order to collect as much information as possible.
A powerful, flexible and trustworthy simulation environment can positively impact in defining a successful solution.
Differently from physical modeling, the software simulation provides a dynamic environment where it is possible to have meaningful insights about the computer models while they are running [4].
But its usage is drastically changed during the years. With the advent of the digital twin these tools have acquired an important key role in their development.
Digital Twins are significantly more accessible than before and their usage it's valuable for companies.
Thanks to them indeed, it is possible to test and understand how systems and products will behave in a wide variety of environments, using virtual space and simulation.
Among industries, these tools have often been used with the purpose of conducting experiments on real systems, aiming to have an idea about the effectiveness and the impact about possible solutions or decisions.

### Tools description

In the first part of the work the Robotic Operating System was used.
ROS is a flexible open source framework for writing robot software.
It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot across a wide variety of robotic platforms [19].
That's because developing a robust robot is truly an hard task.
What seems to be trivial from the human perspective can be though from the robot one.

At the lowest level, ROS provides services designed for computer cluster, low level device control implementation of a message passing interface that provide inter-process communication [19].
These messages have different formats, can happen among different nodes in the system and are exchanged via a publish/subscribe mechanism.
In addition to this, at the very low level ROS by providing robot specific libraries and tools, aims to facilitate the development of the robot itself by implementing the low level functionalities, in order to allow the developer to focus on more specific and advanced features.

During the work of thesis this tool was mainly used to represent the two main cobots involved in the use case A: Franka and Kuka

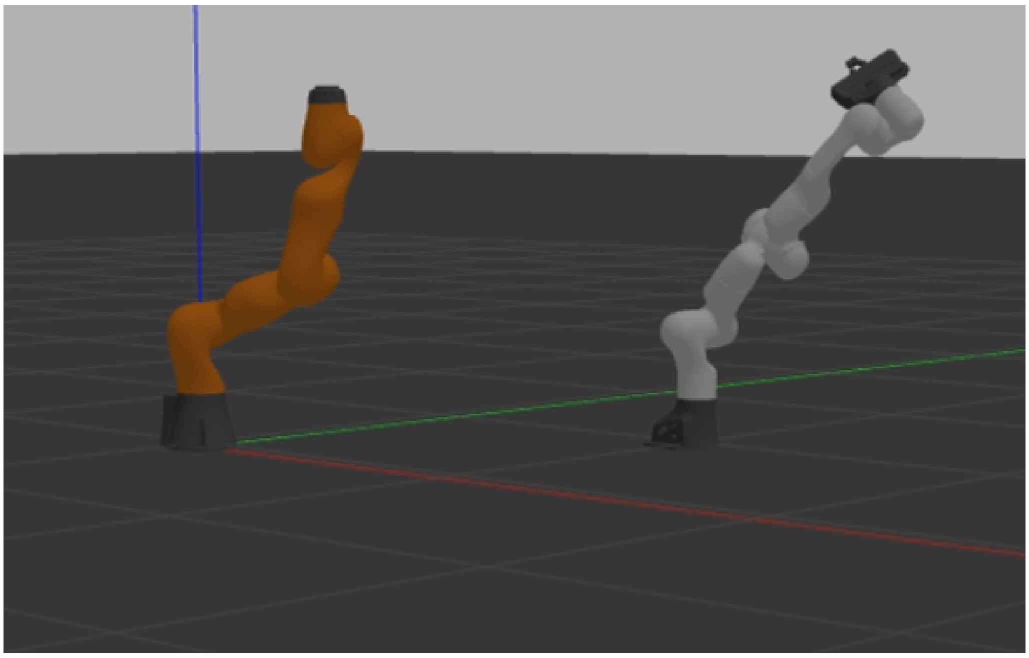


Figure 17: Franka and Kuka simulated in the same virtual environment

On the other hand Anylogic was used in a more exhaustive way by representing the process depicted in the use case A.
The actors, the logic and environment has been defined with this tool which communicate its status to the server by means of the corresponding APIs.

Anylogic is simulation modeling tool that supports agent-based, discrete event and system dynamics simulation.
It is a cross platform, object-oriented software which provides several functionalities. In particular, the supported operating systems, the compatible softwares to

perform specialized functions, the capability of controlling the application or running it with an external program and the multiprocessor CPU support are the main technical aspects of Anylogic.
As for ROS, Anylogic is a framework that takes care of most of the basic functionalities of a robot.
This allows the developers to focus more on the implementation of the logic which is written in Java code.
Each actor can be represented with a class and his behaviour can be defined with methods of the class itself.
Furthermore, Anylogic is a multi method simulation modeling tool that enables the possibility of collecting statistics with an already integrated database and it also provides the possibility of visualizing the simulation in 2D or in 3D.
The choice of selecting Anylogic instead of ROS in simulating the use case A, is related to some important aspects that have been considered:

- Level of details
- Learning Curve
- Required previous knowledge

It can be said that the level of details which ROS is able to reach during the simulation is deeper than the one that can be reached using Anylogic. That's because it takes care not only about the "generic" logic, the one necessary to model the process, but also about every movement and aspect of the robots.
This absolutely from many points of view is an advantage, but for the scope of the use case, such level of details was not a mandatory aspect.
What it was necessary was a tool able to simulate the behaviour of the actors involved in the process depicted in the use case A with trustworthy performances.
Another aspect to be considered is that, this advantage in the granularity of the simulation comes not for free.
It is paid with an higher level of difficulty in defining the logic, because the developer has to take care not only about the process but also about every movement and aspect of the actors involved.
Last but not least, another reason that led the choice towards Anylogic was related to the fact that in order to use it, very few previous knowledge were required.
The most important required knowledge was to know how to program with the Java programming language.
On the other hand in order to develop an application with ROS several skills are required, not necessarily belonging to the programming world:

- Programming languages (C++, Python)

- Command line skills
- Communication concepts
- Robotics concepts
- Mathematics
- Artificial Intelligence
- Mechanical constraints

This combination of factors led the choice towards a solution that not overkill the problematic and at the same time provide a trustworthy representation of the process.

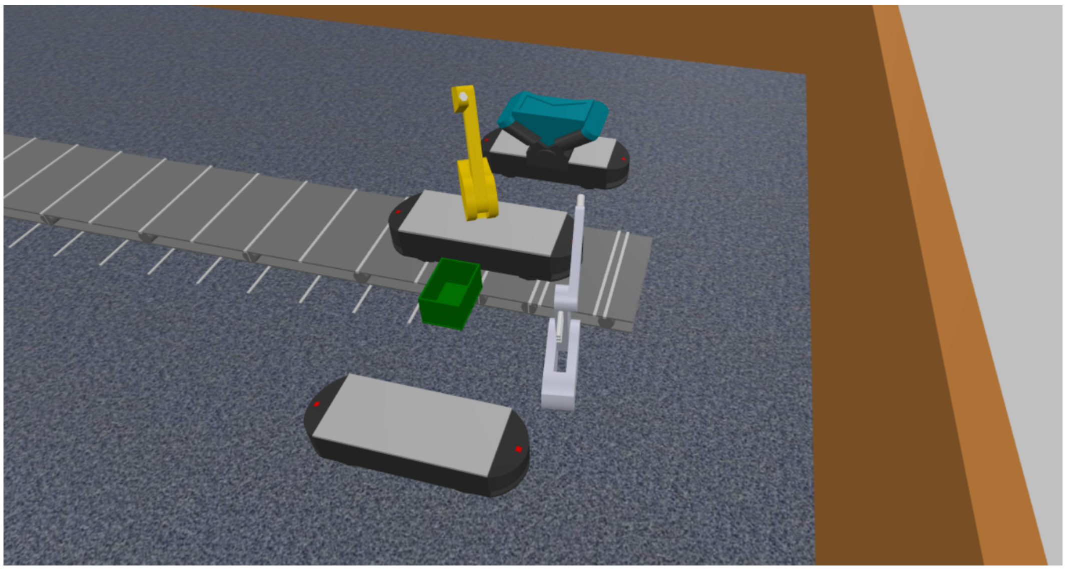


Figure 18: UC-A: picking shafts

# HMI and web services

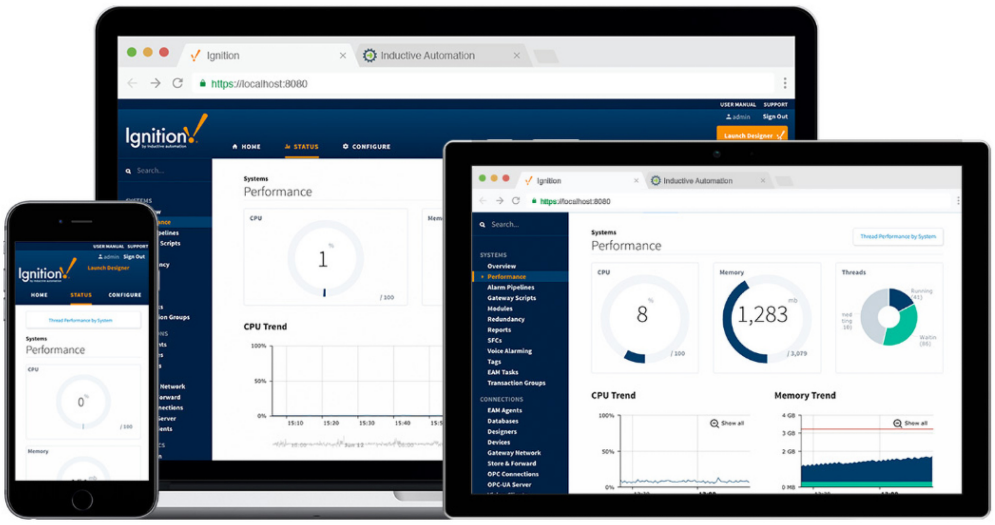


Figure 19: HMI Example [5]

The definition of HMI in cooperative systems is fundamental both to obtain advanced functionalities and to evaluate the cognitive overload. In fact, the operator's environment is full of stimuli, ranging from:

- Functional interfaces
- Information on the movement of the robot
- Safety rules
- Exchange of information

During each work activity, the operator must face the stimuli coming from the robot that can represent a risk.

In this situation, the understanding of the interfaces both in a active and a passive way is essential.
Human Machine Interface (HMI) is a system that provides the controls that enable a user to manage a machine, system, tool or device.
The HMI systems include all the elements that a person will touch, see, hear or use to perform control functions and receive feedback on such actions.
These systems, nowadays, can include supervisory control and data acquisition (e.g. SCADA), alarm systems, as well as the provisioning and the receiving of information from other systems in the network, such as material management or enterprise resource planning systems (e.g. ERP).
The HMI is the main point of contact, the layer between the user and the technology which makes noticeable the aim of a particular technology to the user.

A well designed HMI fits the task the user needs to perform. Its effectiveness is so relevant that in several fields it determines the acceptance of the whole system and can affect the overall success or failure of the product itself.

Traditional industrial HMI can be used for the following operations:

- View data in real time
- Track the time of production, movement of tagged objects
- Monitor the KPIs
- View the input and / or output of machines or robots

The main feature of an HMI is its usability, which includes how easy it is to learn and how productive can be it's usage.
These systems provide the operator of the functions he needs to carry out assigned tasks with the minimal required effort, improving in this way the productivity.
It is the task of all those involved in the HMI design, engineers, managers and industrial designers to meet the usability requirements defined for a specific HMI system.
A well designed HMI system doesn't just present control functions and information; it provides to the operators the needed functionalities to perform their tasks, the feedback on the results of those actions, and information on system performance.

The main aspects to be considered are:

- Generic functionality: how many features will be controlled by the interface; what type of feedback will be more useful for the operator in performing the defined functions (visual, auditory or tactile)

- Real-time indicators: how many data streams are presented, security consideration about data, respect of standards

- Complexity of the output: on / off switch or touchscreen; deciding the input requirements of the information will also influence the output requirements

- Inter-connectivity: the system must be able to interact with other systems if necessary

- Environmental considerations: device conditions of usage such as humidity, temperature, the presence of dust or other substances, magnetic fields etc. Therefore the interface must be designed for an even higher level of reliability, as it is the critical link between the operator and the system

- Regulatory Standards: a deep understanding of ergonomics, engineering and manufacturing standards is critical for the HMI system design

By optimizing an industrial process by digitizing and centralizing data for visualization, leveraging on the HMI, operators can see important information displayed in graphics, diagrams or digital dashboards, view and manage alarms, and can connect with WMS and MES systems, all through a single device.

For what concerns the ICOSAF project, regarding the human machine interface, its goals is to define an effective interaction with collaborative robotics and to support operators in performing their tasks.
It is necessary to develop new HMI systems able to interact effectively with AGVs for the logistic and to display logistic related data in real time.

## State of the art - Front End Frameworks

Figure 20: Front-end development and service [7]

In order to fulfill the goal of creating a web application several possible development tools and environment can be chosen. Among those, the ones that are linked to the JS environment are the most known and used all over the world.
The reason is related to the fact that, originally before the advent of frameworks, front end web pages had to be written with HTML to define the appearance, CSS to define the style and JavaScript to animate the page and create some rendering on the fly.
The popularity of JavaScript, it can be said that has contributed to its wide propagation in the web development world.
The most known and used JS frameworks for the front end developing and considered in the ICOSAF project are:

- React
- Angular

What are the main features that make a framework actually a framework?
Starting from its definition a software framework is a pre-written app structure

useful for a developer to build on top of it.
Practically it is a collection of files and directories already present in the project that takes care about common problems like boilerplate code, directory structure (usually following a design philosophy) and the definition of a set of design principles.
Not every framework actually defines design principles, but the one that actually do it, are usually referred to as *opinionated* [18]

Actually not every JavaScript based "framework" is an actual framework. For instance React and Vue can be classified as JS libraries for the presentation layer, but thanks to the introduction of several libraries, they can be turned to behave as actual frameworks.
A front end is the presentation layer of the web application.
Generally it is referred as the page that the user sees, but this definition doesn't provide a full understanding about what are the functionalities and the work requested.
The presentation layer indeed takes care, as the name suggest, about the presentation; but generally speaking it is responsible for the definitions of efficient solutions for displaying, updating, storing all the data received from the back end through some provided API.[18].
A simple front-end can be developed with just three files: HTML,CSS and JavaScript. However, it is not a scalable solution, because while growing the developed application will be unreadable and unmantainable.
In order to build a robust, efficient and maintainable web application, one of the main components that needs to be used is a front-end framework.
The main advantages in using frameworks are: [18]

- Maintainable solutions: *"divide et impera"*, it is easier to make little changes without impacting on the rest of the application
- Reusable solutions: through a modular architecture it is possible to deploy very easily the same solution in different contexts
- Avoidance of boilerplate code: thanks to creation of reusable code, it is not necessary every time to reinvent the wheel
- Consistent and intuitive UI: standardization of solutions themes and style makes the application more recognizable to the final user
- The usage of well known pattern paradigm (e.g. MVC) facilitates the task of developing in the long run
- Separation of concerns: modern frameworks encourage the modular architecture. Each component will have to take care about specific problems

- Speed: already implemented solutions in different projects can be reused for different scopes

On the other hand, like every solutions, it comes also with some drawbacks that have to be considered, before using it for a project. Here some factors that have to be considered before using a framework: [18]

- Abstract code: until a thoroughly and deep familiarity with the framework, the code can be difficult to understand. It can be hard to distinguish how much of the code is helpful to your application and frustrating to fix bugs due to the unfamiliarity with the code.

- Learning curve: in order to master the usage of a framework, it takes time. To be productive, you need to understand the syntax, tools, and philosophy behind how a framework works. For projects where speed is essential, learning a brand new technology is definitively not the best solution.

- Overkill for smaller projects: for simple projects where is required for instance only a static site, you might not need the power and overhead of an entire framework.

- Setup: setting up and customizing a framework for the specific use case takes time. If speed is essential this might not be the best solution.

- Strong opinions: an opinionated framework may feel constricting, and its design principles may clash with the philosophy of the project.

- Ecosystem evolution: The JavaScript framework ecosystem is famously volatile. The most popular framework today may not be popular next year, and with this changes, developers and support may be hard to find.

## Comparison: React - Angular

Among the plethora of different tools and frameworks, in the ICOSAF project mainly two solutions were considered for the implementation of the HMI:

1. React
2. Angular

React was introduced in the world of the JS frameworks several years ago and became definitely one of its leader. As the name suggest, React encourages to use a reactive approach and the functional programming paradigm.
The applications created with such a technology are divided into multiple components.
A single component file contains both business logic and HTML markup.
Flexibility is one of its main features and advantages, and this was reached with the development of a large spectrum of additional tools. Nevertheless React has some issues related to this flexibility, because for this feature, one of the problems that arise is to identify which solution for a specific problem is best in a specific context. A lot of time can be wasted in order to figure out which is the best option.

On the other hand a competitive alternative that raised is Angular.
A very peculiar history characterizes this framework and a lot of changes have been made during its evolution, but a main turning point was the introduction of Angular 2+.
It started as a competitor of Backbone, but with the advent of React it nearly became obsolete.
In order to come to terms with the new competitor the structure of Angular has totally changed.
One of the key features that Angular began to share with React is the the separation of concerns that produces highly maintainable code.
This concept in Angular has been developed to an higher level, indeed not only several components are possible, but also nested or child ones.
Each component can be seen as a unit that focuses on a portion of the project, on a specific task; it is characterized by a composition of three different files:

1. DOM description (HTML)
2. Style description (CSS,SCSS)
3. Business logic (Typescript, CoffeeScript)

Furthermore Angular has become modular, indeed it is possible to install several external modules very easily; a commendable feature that allows developer to install only what is deemed necessary in the application.
Apart from the Angular components, other important elements in the angular architecture are:

- Services: used as the model of the application, it is the part that is responsible of the communication with the backend and the management of data
- Directive: special HTML attributes that are used to extend the power of the HTML by giving it new syntax
- Dependency injection: DI is a coding pattern in which a class asks for dependencies from external sources rather than creating them itself

But most importantly it is an opinionated framework, unlike React indeed it defines a philosophy of design that, if embraced, can facilitate the work of implementation.

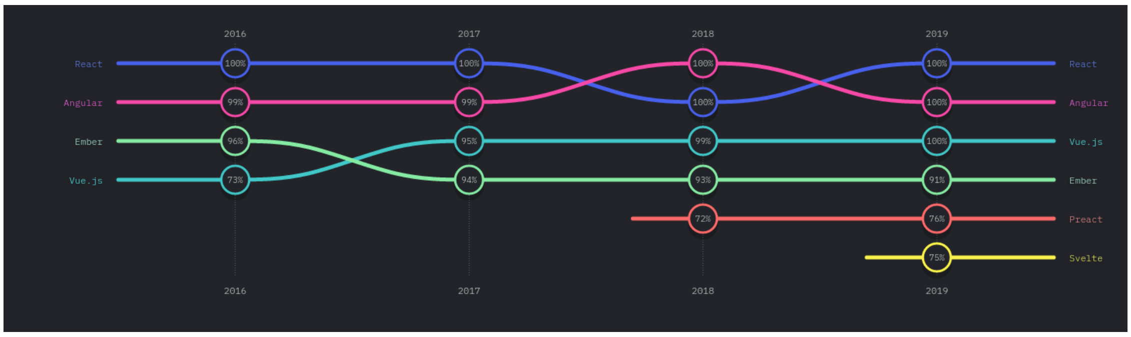


Figure 21: Front-end framework awareness during the years [22]

As can be seen in the picture above 21, many other frameworks were possible and valid alternatives but the ratio for selecting at the beginning this two alternatives was related to the awareness of these tools in the companies.
In conclusion both the alternatives are extremely valid from the technical point of view and both obviously have their pros and cons, for this reason the choice of implementing the solution with a framework with respect to the other was mainly dictated by the familiarity and awareness with it, because as said in projects where speed is important the best solution is to use something it's already known.

## Angular Framework - Architectural Overview

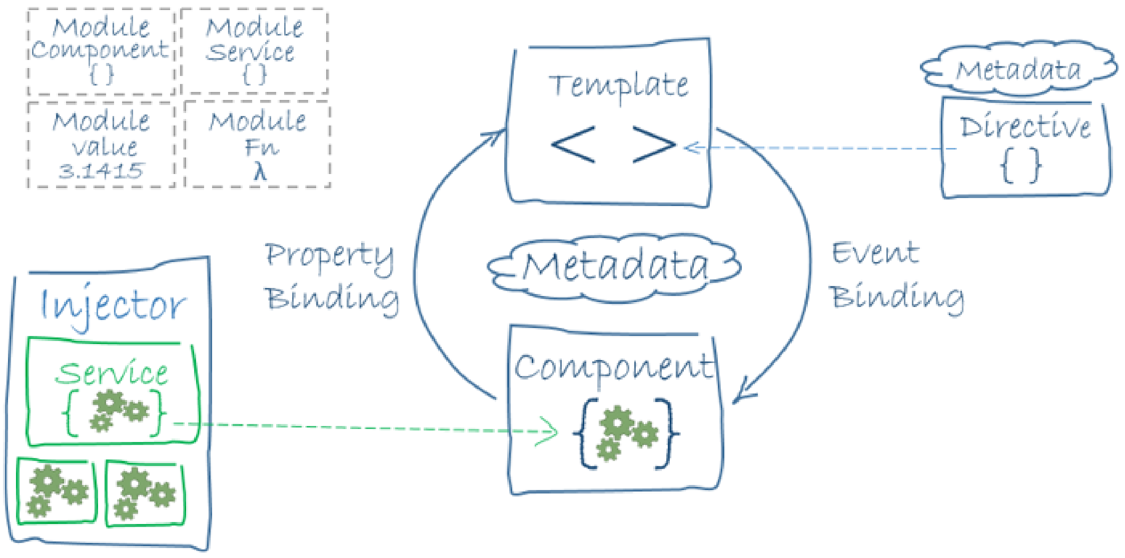


Figure 22: Angular architecture Overview [14]

The HMI in the ICOSAF project was implemented with the Angular framework. Angular is a framework that defines an environment where web applications can be developed. In particular it belongs to the JS front-end frameworks and it provides all the necessary tools for building Single Page Applications (SPA) using mainly three languages:

- Typescript
- HTML
- CSS

The Angular framework is one of the most popular and many companies have developed applications with this framework. Its core and functionalities are defined in Typescript with several libraries that must be imported in order to exploit them. The architecture of an Angular application is based on some key concepts that have to be introduced in order to fully understand what happens behind the scenes.
In particular one of the basic blocks of Angular are the components which are organized into NgModules.
These elements can be seen as containers of functional related code, that are used by the Angular framework to organize related functionalities together.

In a single app there is at least one root module used for bootstrapping the application, and typically for scaling purposes and for a better organization, others modules are added.
As said the components are the nourishment of the angular framework, indeed they define at the same time the presentation of a particular part of the application and the corresponding business logic.
Components make use of another important element: the service.
Services are elements which are responsible of providing, as the name suggest, all the services needed to obtain all the necessary data, taking care about possible communications with the back-end.
In order to have a good organization and a cleaner code, Angular has introduced the concept of the Dependency Injection. Indeed, a component in order to make use of services, it injects dependencies from external sources rather than creating them itself, making code modular, reusable and efficient.[3]

Another important element in the Angular architecture is the metadata that are provided in the various elements through the usage of decorators.
These metadata are used by the compiler in order to describe what are the purposes of the various elements and to define how they should be treated.
Metadata in a component class for example is used to associate it with the template that define the corresponding view, linking in this way the presentation layer, basically the page document object model (DOM) with the application data and logic defined in the component.

A template is the description of the presentation layer, usually written in HTML combined with specific Angular markups used to modify HTML elements before displayed.
The directives in the template helps in defining specific behaviours and through the concept of binding is possible to create a link between the application data and the DOM.[3] There are two types of data binding:

- Event binding
- Property binding

Event binding allows the application to react to specific actions of the user (e.g. click, double-click) with the corresponding logic defined in the component.
On the other hand, property binding allows to bind values of the application data layer with the HTML DOM.
In other words, similarly to an object oriented language, they provide a similar functionality to the getters and the setters in class. The difference is that here it is actually defined a logic in a reactive way, based on the user interaction.

Before rendering a view, Angular evaluates the directives and resolves the bindings in the template, to modify the DOM according to the component data and the business logic defined.
Last but not least another key element is the router.
Since this framework was introduced to develop SPA, this element allows to define navigation paths which describe the different application states and view hierarchies in the application.
It has been modeled based on the familiar browser navigation conventions: [3]

- Based on the URL in the address bar, the browser has to load the corresponding page

- By clicking on links the browser navigates to the landing page

- By clicking on the back and forward browser's buttons, the browser navigates backward and forward through the history of pages visited.

In order to recreate this behaviour without actually navigating to a new page, the router maps URL-like paths to views instead of pages. For this reason, based on the user action the router intercepts the requests and shows the corresponding view in the hierarchies instead of loading a new page as a browser would do, all in a lazy-load fashion.
This binding of the view with the URL-like paths is done according to the navigation rules and states defined in the application itself, associated with particular components.

## Comparison: Traditional Web App - Single Page Application

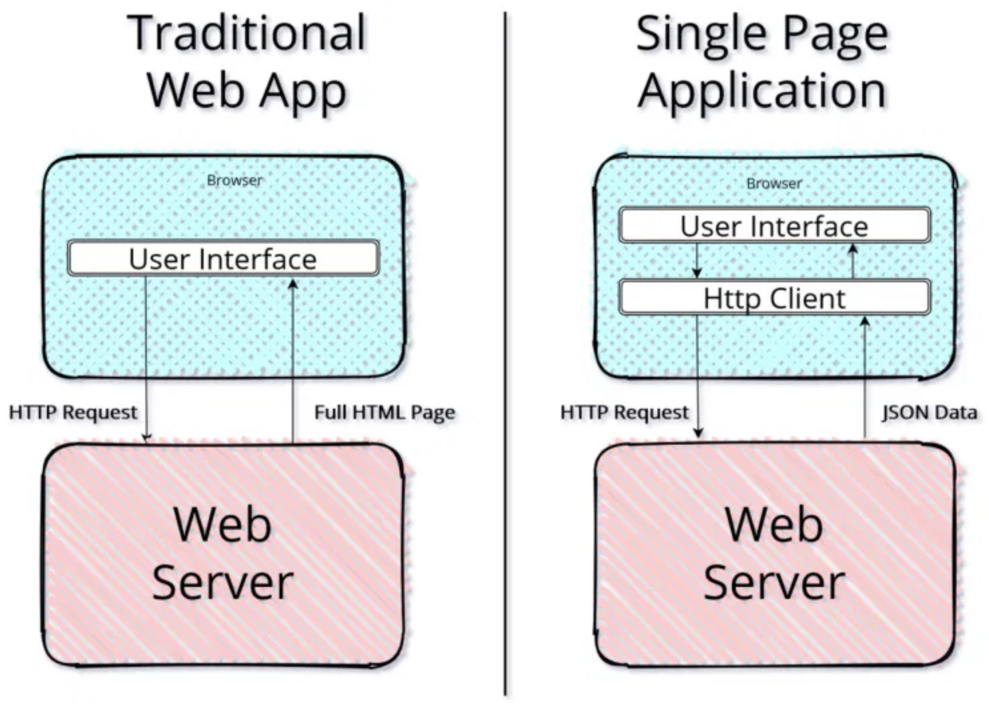


Figure 23: Traditional vs Single Page Application [11]

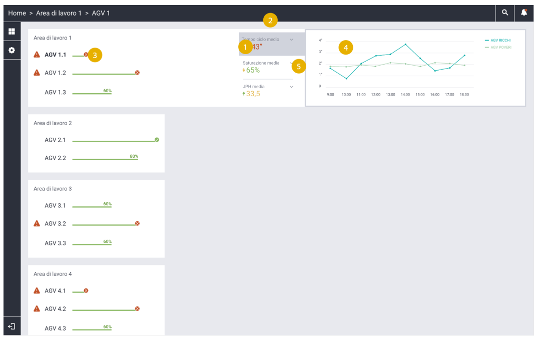
There are two main design patterns for web apps: the traditional web application and the single-page application (SPA). Obviously, both models have their pros and cons. A single-page application is defined in such a way that it is not required page reloading during its usage.
For this reason one of the key difference with respect to the traditional web apps is its responsiveness.
They are faster, because the structure of the DOM and its style are defined and sent only once, whereas during the life cycle of the application only data are exchanged back and forth.
Moreover it defines a pattern where the idle time, where the page is basically unusable, is reduced to its minimal terms.
Many of the applications that every day are used, are defined using this kind of design pattern: Gmail, Google Maps, Facebook or GitHub. Obviously this advantage comes with a price: its implementation is less straight forward with respect to a traditional web application because different aspects have to be managed in order to make the SPA working in the proper way.

Moreover in order to implement this kind of solution usually it is suggested the usage of a framework, which if it is not known a-priori it requires some time to get familiar with it. On the other hand traditional web apps works in a different way. Every URL, indeed, corresponds to a page to be displayed but each time there is a navigation request, the browser actually reload all the required resources (new page generated server-side).
For this reason the traditional web applications are slower with respect to the Single Page ones, and moreover they overload the server with tasks that nowadays clients can easily manage by themselves.
The main advantage of traditional web application is that there is not the necessity of managing anything about routes mapping and components, but this is paid in terms of performances.

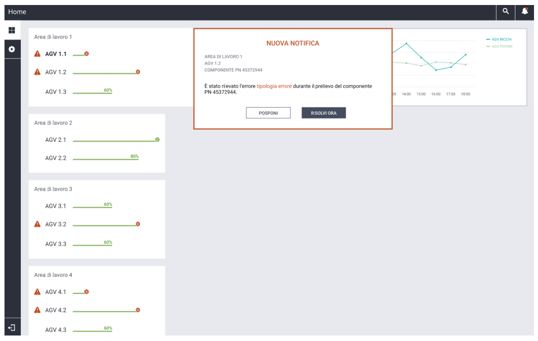
## Dashboard - ICOSAF

In the ICOSAF project, it was required to develop a web application able to provide all the necessary tools to the operator in order to fulfill his main tasks.
In particular the dashboard had to allow the possibility of visualizing the status of the processes (defined in the use case documentation) and to manage the execution of the process itself by controlling the activity of the cobots involved in the process.
The technology used to implement the Dashboard was a Single Page Application developed with the Angular Framework.
The application interacts with the back-end in the Reply private cloud to obtain all the necessary data and to communicate the status of the process in order to be reflected in the persistence layer.
The interaction was possible thanks to the definition of a RESTful web API accessed by opportune HTTP requests performed by the client devices such as smartwatches, PCs or tablets. Both the remote and the field operators visualize all the needed information to perform their tasks. But it's the remote operator who uses the dashboard to have deeper insights about what happens in the different working areas and what are the status of the different AGVs. It is possible indeed to see the situation of each AGV in each working area and also relevant statistics related to the tasks which help the remote user understand how the process is proceeding and if a corrective action needs to be performed.
Moreover thanks to a push mechanism, notifications about errors are produced in order to warn the operators that something out of the ordinary scenario happened, and this facilitate the responsiveness of the overall system.
The style of the application has been defined thanks to the work of a designer involved in the ICOSAF project, who was responsible of providing sketches and wireframes both for the dashboard and for the mobile application that needs to be

developed in order to be used on smartwatches by the field operator.
She took care of defining an application design able to provide the most important information in a clear and intuitive way.



(a) Web Application prototype



AREA DI LAVORO 1
AGV 1.2
COMPONENTE PN 45372944

È stato rilevato l'errore tipologia errore durante il prelievo del componente PN 45372944.

(b) Notification example

Figure 24: Wireframes - ICOSAF project

## Web application - DiPreTreat

Concerning the DiPreTreat project, it was required to developed an application able to provide and show all the necessary information about the specific problem that this project aims to monitor and solve: the deterioration of the reinforcing steel for concrete buildings.

The analysis about the technologies to be used are valid for both the ICOSAF and the DiPreTreat project.
Since it was required to develop a web application, among the frameworks representing the state of the art, it was decided to use the Angular framework in order to develop a Single Page Application in the shortest time thanks to the familiarity with the technology.
Differently from the other project the style and theme of the application were not defined by the work of a designer involved in the project but decisions about the presentation and organization of the view were made during the development phase.
The application interacts with a Reply server to obtain the necessary data and this interaction is based on the definition of a RESTful API provided by the server.
The client application contacts the server to the particular endpoints that have been defined and requests all the necessary data by using the HTTP communication protocol.
From the logical point of view this application represents in a way a glue among the different functionalities that the project would like to provide.

The aims of monitoring the corrosion of the premises with a network of sensors and the analysis with AI algorithms performed on the data collected are used as input for creating a web application able to show and represent in a meaningful way the problem to be faced.
It's indeed here that all the data collected are organized, processed, represented in dedicated charts, provided to external servers in order to perform forecasts and last but not least provided to the final user in order to support the decision process.
One of the most important aspects of the application is the definition of an intuitive and explicative web service able to be used by most of the people and not only technicians.

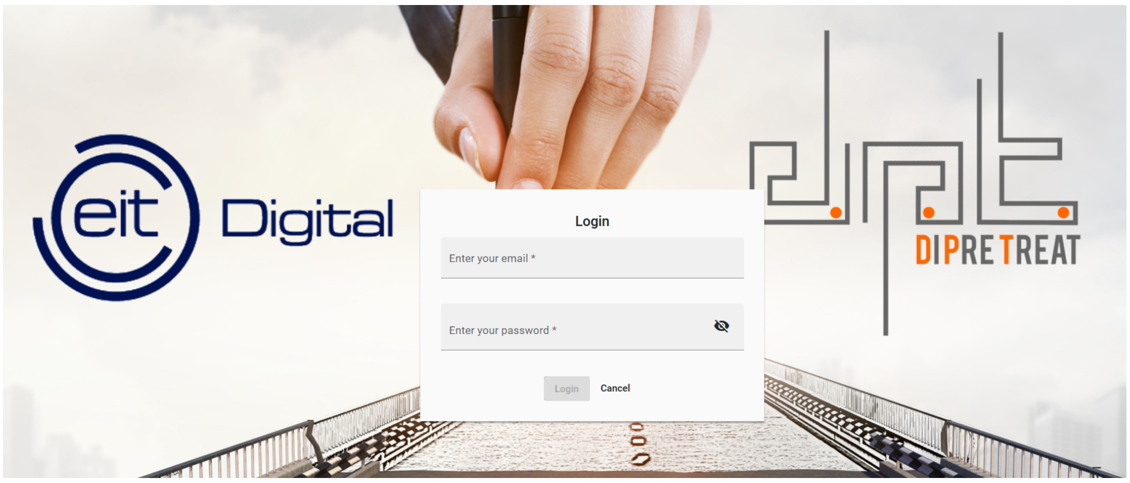


Figure 25: Example of web application

# Second Part

# ICOSAF - Simulation

In the ICOSAF project, the process described in the use case A was defined and optimized with the creation of a simulation model. The tool used for creating such a model was, as said, Anylogic, and the purpose was to try to follow the description and directives provided by CRF.
The simulation is used to solve real problems and to answer to specific questions raised when the solution is still a draft. By using the simulation indeed it can be created a virtual representation of what is needed to recreate the process, resource or phenomenon in question.
By doing this it is possible to produce something more tangible than simply an idea and it can be collected more information and data to have a deeper insight about the solution proposed. The aim of the simulation is to define a model that reflects with the highest fidelity possible the characteristics of the real one.
For this reason all the elements involved in the simulation, starting from the actors, needed to be defined in order to perfectly fit the demands and requisites.
By using simulation models moreover it is possible to understand if a process, a system or a particular solution to a specific problem is feasible and effective or not.

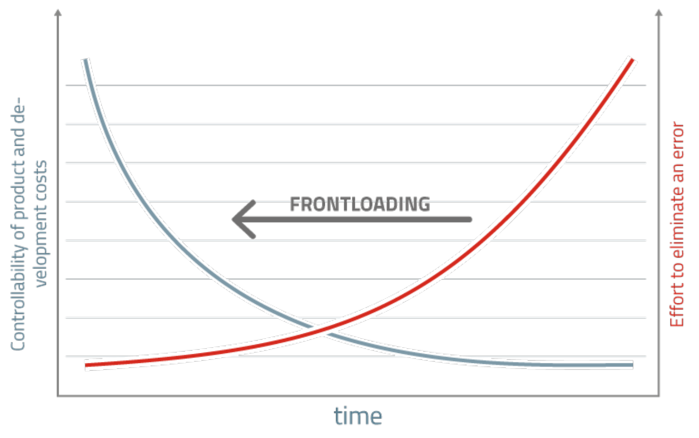


Figure 26: System-Simulation Development process [20]

It must be considered that corrective actions in projects that are still in their draft phases, have the biggest impact on development and production costs as well as on the products characteristics, as shown in the picture 26.
Making corrective decisions and changes in the early phases of the project can positively impact in the definition of a successful product or process.
Based on the principle of the *"front loading"*, simulation models allows to have detailed information about the process itself, and they can be used to evaluate and possibly modify the overall product in order to comply with the required properties. [21]
The concept of front loading is a strategy that tries to improve and optimize a process by shifting the identification and the solving of the problems themselves, to the early phases of the process lifecycle.
Simulating a system, hence, provides certainly a concrete advantage in terms of cost savings, optimization of the process and can give meaningful information about the performance and effectiveness of the solution proposed.
In order to virtually simulate the physical process described in figure 8, different tasks have been carried out.
It must be considered that a partial solution of the process was already defined for the use case A, for this reason the activity here performed was, by taking advantage of what already was defined, to optimize the process and to complete it in order to comply with the specifications.
The simulation is characterized by 7 main parts:

- Aims of the activity
- Variables, Parameters and Actors definition
- 2D Representation
- 3D Representation
- Communication
- Logic
- Optimization

Each of them will be discussed and explained in details.

## Aims of the activity

As said, a partial representation of the process depicted in the use case A was already defined with the Anylogic simulation tool.
Hence, the aim of the activity was not to completely recreate the process from scratch, but to complete, enhance and optimize it.

In particular given the solution already implemented it was asked to rethink about the logic, to modify it in order to optimize it and to fit the demands of CRF, and furthermore it was asked to enable the communication between the simulation environment and an external server in order to have the possibility of gathering data from an external source and represent it on a dedicated HMI as it was a real process.

The already implemented solution, had some discrepancies with respect to the requests defined in the documents of the project.
One of the discrepancies, was related on the way it was implemented the synchronization system, realized to make the actors cooperate in the environment created for the process.

As depicted in the architecture figure 8, there are 4 cobots involved in the process:

- Cobot Kuka LWR iiwa 820
- Cobot Franka Emika Panda
- AGV transport Kit with relative support trolley for the kitting box
- AGV responsible of transporting the engine block

All of them have to perform their own tasks and must cooperate to reach the goal of the simulation. These actors are synchronized based on the activity of the main actor: the shop floor operator.
It's indeed the field operator that by using the HMI is in charge of the advancement of the process.

By performing the checks on the shafts and on the engines, the operator communicates their status, if a problem has raised or not, and by doing so, basically makes the AGVs going forward in the process.
In the provided solution, the synchronization was based on a non standard solution: timers.

It was indeed a very rigid and unmanageable way of synchronizing the process where any possible change in the structure or in the actors activity could have led to the modification of the entire logic of the implemented solution.
For this reason, the aim was to recreate a process where the synchronization was not based anymore on timers but on a more solid and maintainable strategy: asynchronous events managed with hold blocks.

An hold in Anylogic is a particular block that can stop the execution of an *agent* activity along a particular flow.
By using holds with parameters that provides a way of defining a callback whenever the parameter changes, it was possible to create a synchronization system.
This solution is more solid with respect to the previous one, that's because if a task is taking a longer time to be fulfilled, with timers the system goes out of sync and the solution proposed breaks, on the other hand the new solution is time independent and for this reason any change in the time execution does not affect at all the synchronization of the system.

A second aspect that was a bit different with the demands of the CRF was the number of actors involved.
It is possible that in the early stages of the project more actors were involved, but with the advancement of the project itself, some of those have been reconsidered and removed.

A third aspect that have been considered in the activity related to UC-A is the optimization of the process.
In particular in the solution previously defined there were many more parameters and variables than the ones effectively needed, and moreover the logic defined was a bit cumbersome.
For this reason, another task was to make the code a bit cleaner in order to be understandable, maintainable and more efficient.

A fourth aspect was the definition of a statistics section in order to gather data and insights about the process itself thanks to the definition of different graphics and charts.

## Variables, Parameters and Actors definition

In order to correctly represent a process, in any simulation tool, one of the key aspects that should be considered is the definition of the actors involved.

In the proposed solution for the use case A, the involved actors were already defined in their 3D representation but since the logic of the process has been changed, also some aspects in their definition changed too.

In particular given the fact that each actor represented was defined in a Object-Oriented fashion with its own *class* and *properties*, slight modifications in their definition were performed.
Useless or redundant actors were removed and only the necessary ones, the one described by the prescriptions of CRF were maintained.

Moreover in order to correctly define the behaviours of the actors in the simulation, the usage of variables and parameters are mandatory.
In Anylogic the definition of those are slightly different and their usages are suggested for different scopes, because they have different characteristics.

In particular, parameters are considered and used to define something more related to the modeled object, a characteristic that during the execution of the process might change but if it does, a callback can be defined in order to apply a specific logic.

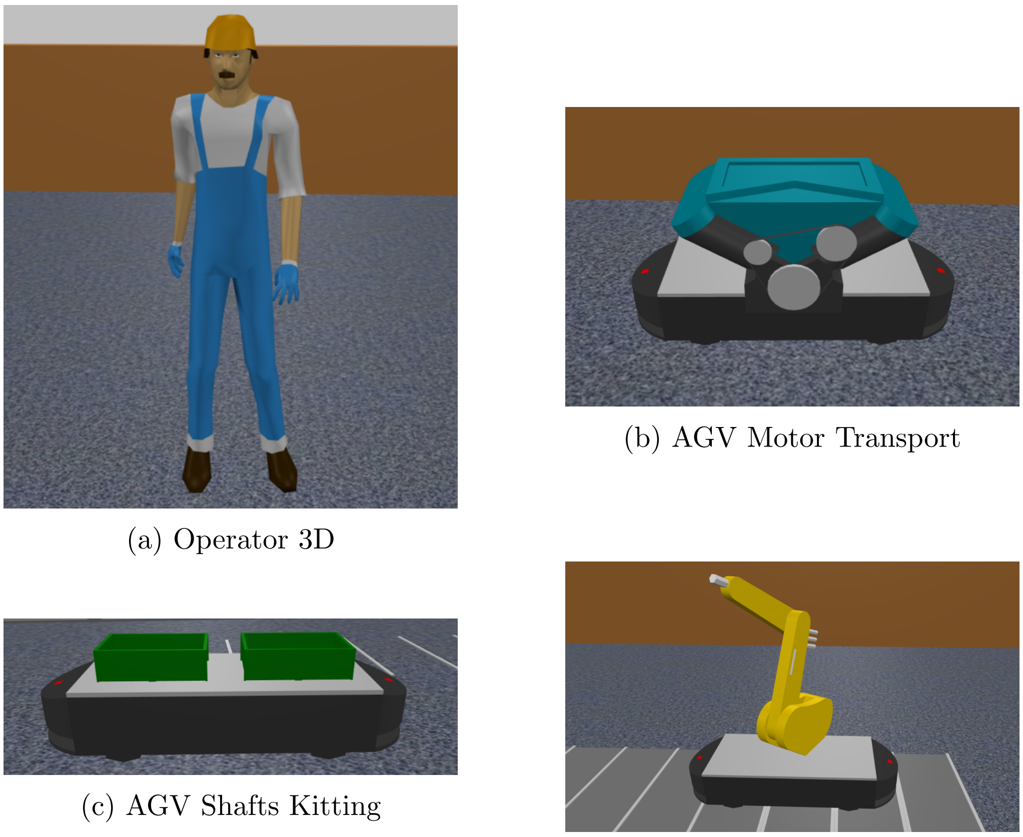
On the other hand a variable in Anylogic is a standard variable as defined in any programming language.

## Actors

In Anylogic every entities involved in some actions need to be defined.
For this reason in the list of actors that have been modeled it must be considered the shafts and the motor.

Both are created as *agents* in Anylogic and their characteristics defined in their own modules. As said the previous implementation already provided a 3D representation of every actors involved in the process, for this reason characteristics like the initial position, the color and the scale etc. were not redefined.

Nevertheless in order to match the requests, some properties have been added to some of the actors involved.
In details, the tasks, that have to be performed by the actors, belonging to particular orders (defined at the beginning of the simulation) and other features were added.
Since the actors must know which order and in particular which task is executing, those orders have been assigned to the agents in order to than allow the communication with the HMI.


(a) Operator 3D

(b) AGV Motor Transport

(c) AGV Shafts Kitting

(d) Kuka 3D

Figure 27: Main Actors

## Variables, Parameters and Functions

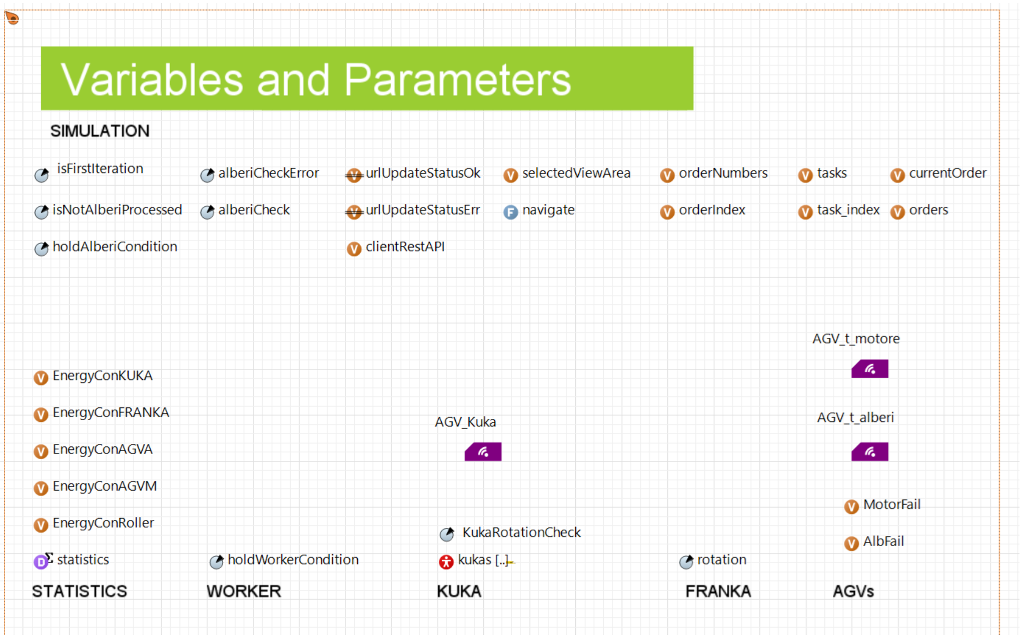


Figure 28: Variables, Parameters and Actors

Throughout the redefinition of the logic new variables and parameters have been added in order to be used both for configuring the simulation and for applying the logic explained in the use case A.

In order to redefine the synchronization mechanism a consistent use of parameters was made, because they allow to define callbacks whenever the considered parameter changes, in order to define in this way a reactive logic.
In particular the ones that were useful for synchronizing actors are the ones used in the hold blocks as condition for blocking the activity of an *agent* or in the conditions where checks were performed (e.g. holdAlberiCondition, alberiCheck ).

The variables introduced in the simulation instead were mainly used for describing the logic of the process.
The ones that are relevant for the logic described in the use case, like the orders that should be carried out during the simulation, the tasks related to a particular order and so on, are either collected at the beginning of simulation or after each time an order is executed.

Moreover, as shown in figure 28, the navigate function has been defined to navigate among 3D visualization of the simulation, 2D, logic of the different actors and so on.
One of the most important variable that has been introduced is the clientRestAPI. Since Anylogic is based on the Java programming language, it allows the definition of custom Java classes.
The ClientRestAPI class is one custom class used for a very specific purpose: communication.

This is not the only one that was defined, indeed other two specific classes were defined:

- Order
- Task

Both are representation objects of the data structures that are received from the back-end when communicating to understand the orders and tasks to be executed. Both are defined as POJO classes, but besides the classic *getters* and *setters* also an important method is defined:

- getTaskForEngine();

This method is really useful for understanding the details of task related to the AGV motor transport when reaching KUKA for the assembly operation.
This method was defined because of a main aspect in the definition of the logic for the process represented in the Anylogic simulation tool: *agent centric.*
It means that the advancement in the process and the several tasks are viewed from the point of view of the shafts and the engine.
Those two actors move forward in the different stages of the process and are involved in all the crucial activities.
For this reason it was decided to perform the communication to the backend by following these actors that are the ones interested by all the actions to be performed.
Since the two-flow work in parallel, it was necessary to distinguish the only task that could have been out of sequence and that was the movement of the AGV motor transport that goes towards the cobot Kuka for the assembly operation.
After that indeed the two flows (of the shafts and of the engine) begins to work together and the calls become sequential.
In order to discern that particular operation this method was crucial, because once the order was assigned at the beginning of the operations, all the operations related to the motor and to the shafts were defined correctly.
In this way the decoupling of the tasks related to the two agents was completed and the two flows were able to work in parallel.

## 2D Representation

The 2D representation was defined already in the previous solution accordingly to the prescriptions. In this use case it is an area where two circuits of operations are defined.
The one below related to the AGVs transporting the motor and the upper one related to the AGVs transporting the shafts.
In the center, in green are defined the positions of the operator which moves among them by performing the different tasks.
On the right there are two spawning positions, the one for the AGVs transporting the motor and the one related to the AGVs transporting the shafts.
On the utmost left instead there are the nodes where Kuka and Franka perform the exchange and the assembly of the two shafts on the motor.
As can be seen there are several blue circles, and those represent the several nodes (stop nodes) where the AGVs needs to stop to perform some task before continuing their own activities.
Another very important issue that has been managed at this level is the camera.
Its position in fact needs to be set in order to see the 3D simulation from a good starting point of view, without the need of navigating in it to find the perfect spot.

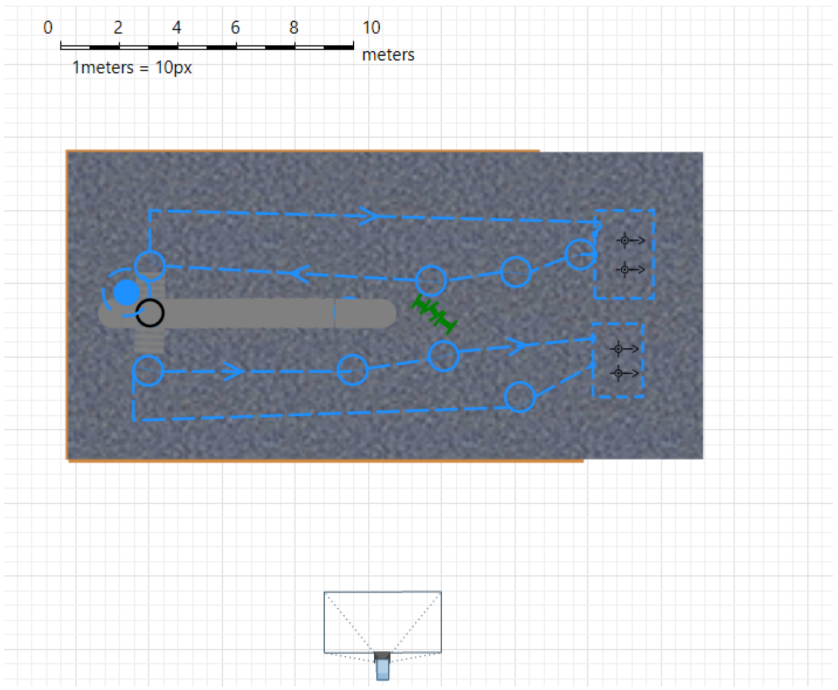


Figure 29: 2D

## 3D Representation

The 3D representation is automatically defined by AnyLogic starting from the 2D's one.
As said a representation of the actors in 3D was already provided, but what has been introduced here is the possibility to easily switch between different sections by means of simple clicks on buttons.
The sections that have been highlighted are the ones where meaningful insights could be gathered.
Indeed the sections where there are the names of the different actors it is possible to visualize their logic structure, whereas by clicking on the utmost right button it is possible to visualize the figure 28 with all the variables parameters and actors that have been defined for the simulation.
By clicking on one of the first three buttons instead, it is possible to visualize respectively the 3D, the 2D representation and the statistics section about the process where information about the number of errors found and statistics about the consumption of the cobots and other insights have been provided.

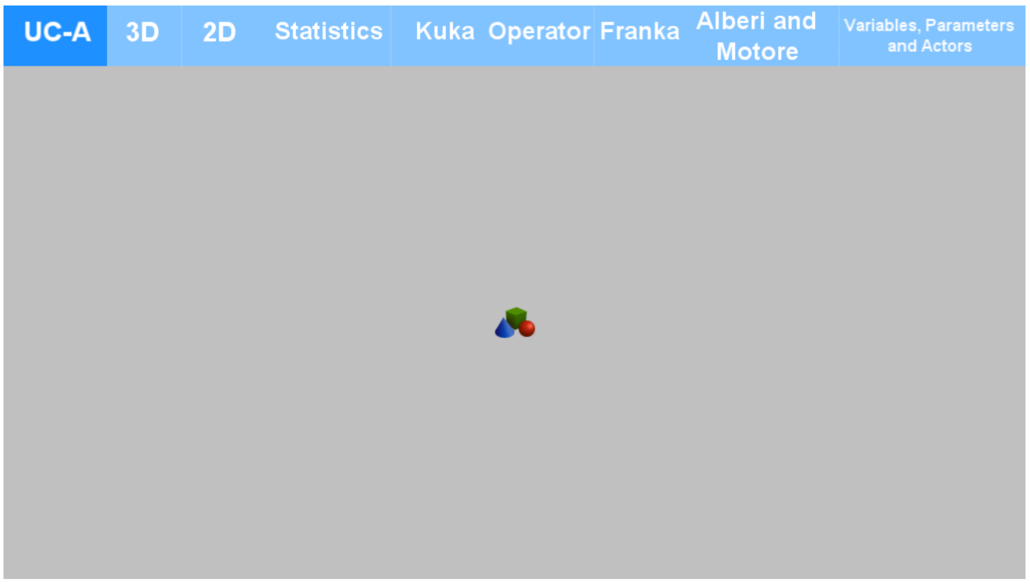


Figure 30: 3D

## Communication

The communication is a feature of the application that was totally not implemented in the previous solution.
In order to allow the communication in a REST fashion a specific class was introduced: ClientRestAPI.
This class was defined to take care about the communication with the beckend.
That is allowed due to the fact that the server provided a variety of APIs useful to communicate what is happening and to make the server react to specific requests.
An instance of this class is in charge of producing specific requests according to the needs: for instance to obtain the list of orders to be executed during the day.
By getting the response, this class is in charge of processing it, elaborating it, if necessary to convert the formats and return specific objects to the entities making the requests. Some methods of this class, as said, are used to ask for the order numbers to be fulfilled during the day, but also for requesting the list of tasks to be accomplished for each order or simply to perform an HTTP call to signal whether a pick has been correctly performed or not.

The name of the methods implemented are:

- activate(String urlPassed):
  used to contact a specific URL

- askForTasks(int orderId):
  used to ask for the tasks of a specific order

- askForOrderNumbers():
  used for requesting the order numbers of the day

These three methods are called obviously in different moments during the execution of the use case. In particular the first method is used a wide variety of times for contacting either *updateStatusOK* or *updateStatusErr* APIs of the server in order to communicate if a specific task has faced an error or not.
On the other hand the second one, is called whenever there is the necessity of processing an order, while the third for the moment is called only once at the beginning of the simulation to obtain the orders of the day.
A possible enhancement that can be done is to request for new order numbers whenever all the scheduled tasks of the day are processed, but for the moment this functionality was not implemented because is not strictly necessary for the aim of representing the process.

## Logic

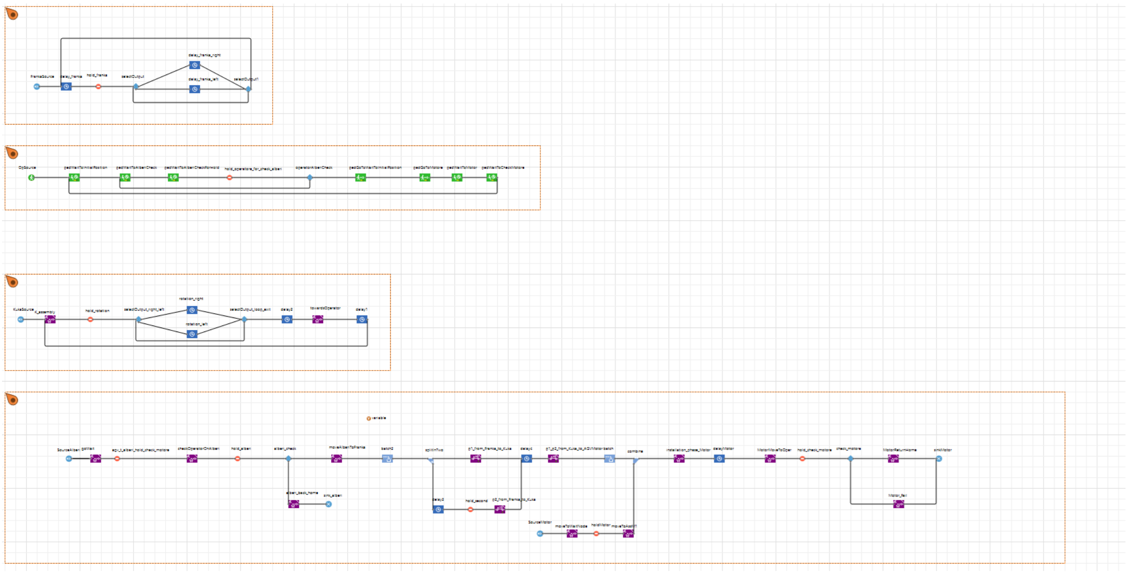


Figure 31: Logic

The logic of the process is without any doubt the most crucial part of the simulation. This is the most important aspect of the simulation in Anylogic, because it is in this stage that are defined the behaviours of the different actors involved, the communications with the backend, the usage of the parameters and variables previously described and the synchronization mechanism.
In order to develop the logic of a process Anylogic decompose it in blocks. Each block takes care of a specific action and the connections among the different blocks give the possibility to create a flow of actions that the specific actor need to perform.
Inside each block that represent a specific action, it is possible to add Java code that most of the times is integrated for implementing specific functionalities.
As it can be seen four different sections are defined and those represent the logic that have been defined for the different actors:

- Operator
- Kuka Cobot
- Franka Cobot
- AGVs kitting shafts and motor transport

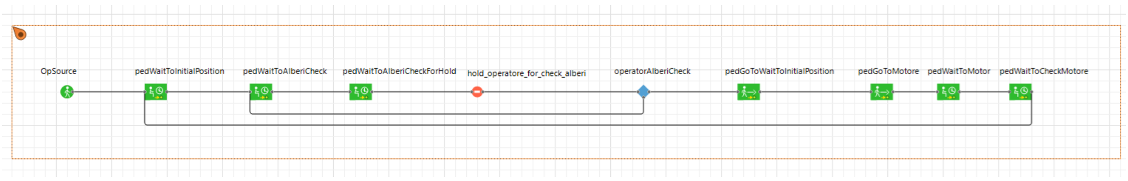


Figure 32: Operator Logic

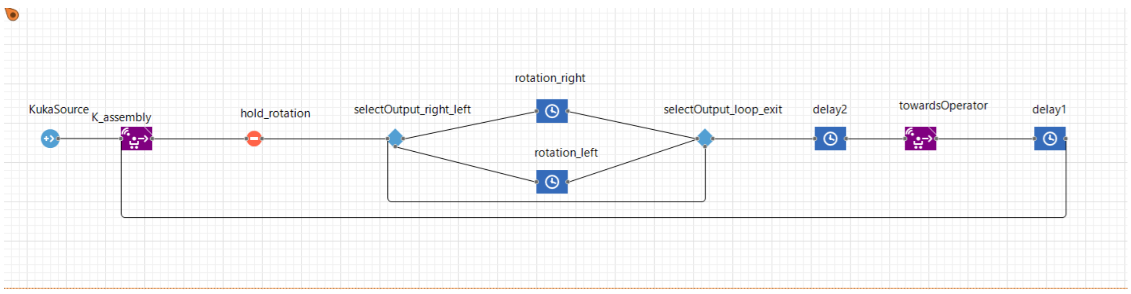


Figure 33: Kuka Logic

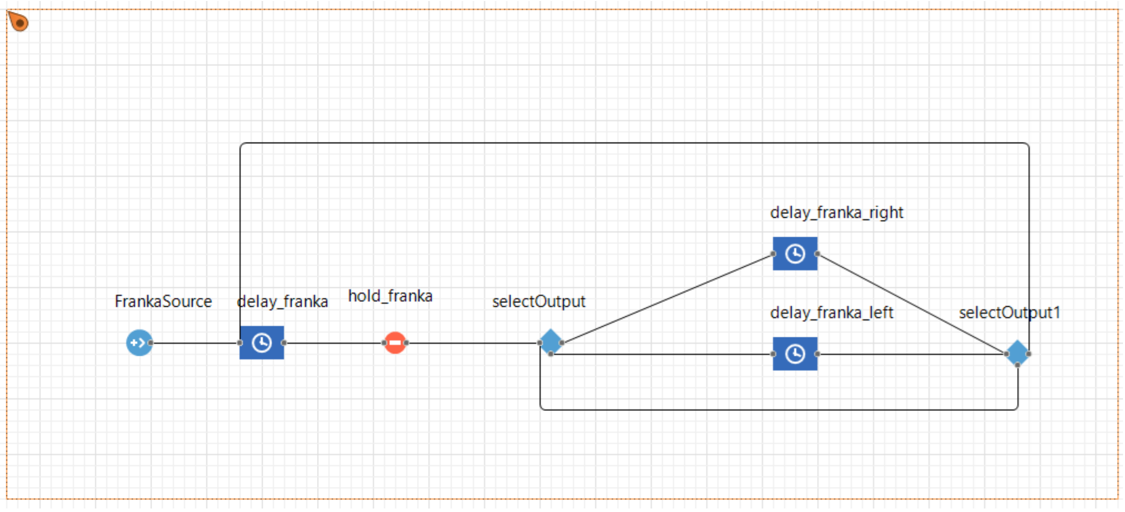


Figure 34: Franka Logic

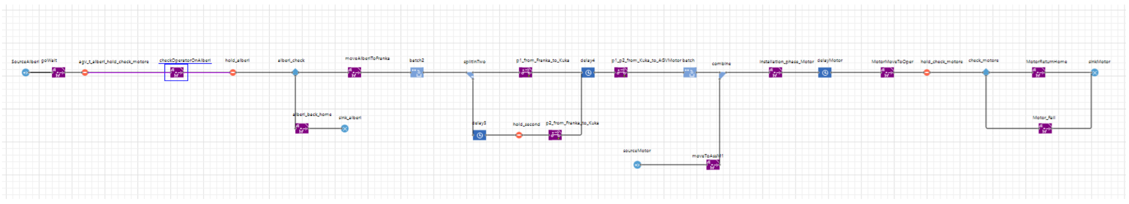


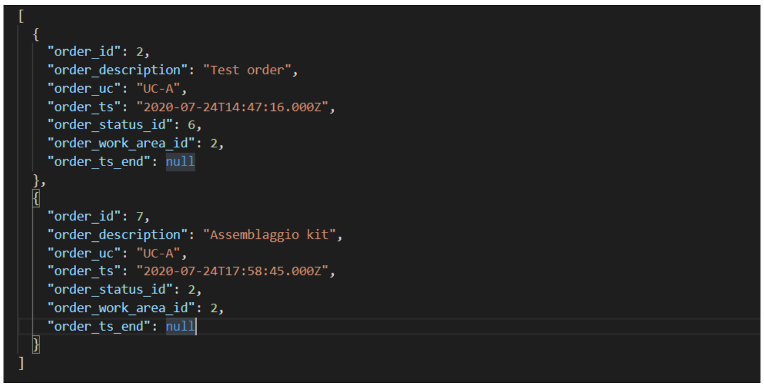
Figure 35: AGV Motor Transport and Shafts Kitting Logic

As it can be seen from the previous pictures for each actor a corresponding flow of activities has been defined.
The operator has to move from the positions where he can perform the checks and move back according to specific conditions.
In particular to synchronize the activity of the operator with the other cobots different holds were used.
Regarding the logic defined for the Kuka cobot instead, another flow of activities have been defined.
In particular Kuka has to move to the assembly node, has to perform the picking of the shafts and the assembly of the shafts, the first one on site, the other while moving towards the operator.
In order to synchronize the activities of Kuka with the one of the other actor, also in this case an hold with the corresponding callback was used.
For what concern the activities of Franka defined in figure 34 instead, it can be seen that the activity to be performed here is to simulate the rotation to the right and to the left in order to emulate the picking process. Also in this case the synchronization was made thanks to holds and parameters.
All the activities that have been presented so far are repeated for each order that needs to be processed.
Last but not least the main logic, the one related to the motor transport and to the kitting of the shafts are defined in a single branch in a *agent centric* fashion.
This logic definition is more complex with respect to the previous one because here all the actions related to the two types of AGVs are here unified in a single graph.
The reason of this choice is related to the fact that the logic of the two actors are strictly interconnected and furthermore a draft of such a structure was already implemented in the previous implementation.
For this reason without deleting all the process, a reorganization of the code, a modification of the existing logic and an optimization of it was performed.

The process starts with the simulation environment asking for the orders and the tasks that are scheduled for the day.
This activity performed in the main startup is carried out via the usage of an instance of the ClientRestAPI class.
It's indeed via this class and specifically by calling the methods askForOrderNumber and then askForTasks that a series of requests are performed to the backend by contacting endpoints like the one reported below:

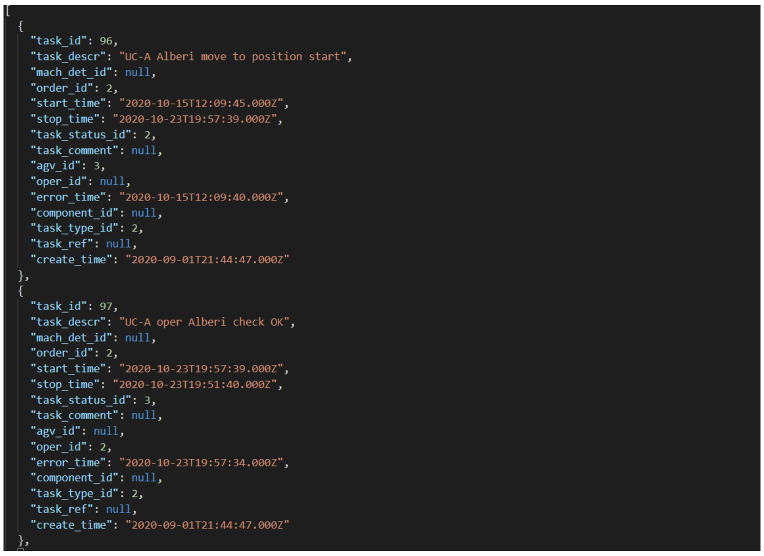
- `http://icowms.cloud.reply.eu/Details/getOrdListbyDate?ts=2020-07-24&uc=UC-A`
- `http://icowms.cloud.reply.eu/Details/getTaskListOrder?order_id=1`

By performing HTTP requests with GET method to those endpoints first the orderNumbers are collected and than the list of tasks for the first order is filled. As it can be seen in the following pictures the response with the useful data are provided in a JSON format.
These are the structure of the data received regarding the orders and the tasks.



```
[
  {
    "order_id": 2,
    "order_description": "Test order",
    "order_uc": "UC-A",
    "order_ts": "2020-07-24T14:47:16.000Z",
    "order_status_id": 6,
    "order_work_area_id": 2,
    "order_ts_end": null
  },
  {
    "order_id": 7,
    "order_description": "Assemblaggio kit",
    "order_uc": "UC-A",
    "order_ts": "2020-07-24T17:58:45.000Z",
    "order_status_id": 2,
    "order_work_area_id": 2,
    "order_ts_end": null
  }
]
```

Figure 36: Orders



```
[
  {
    "task_id": 96,
    "task_descr": "UC-A Alberi move to position start",
    "mach_det_id": null,
    "order_id": 2,
    "start_time": "2020-10-15T12:09:45.000Z",
    "stop_time": "2020-10-23T19:57:39.000Z",
    "task_status_id": 2,
    "task_comment": null,
    "agv_id": 3,
    "oper_id": null,
    "error_time": "2020-10-15T12:09:40.000Z",
    "component_id": null,
    "task_type_id": 2,
    "task_ref": null,
    "create_time": "2020-09-01T21:44:47.000Z"
  },
  {
    "task_id": 97,
    "task_descr": "UC-A oper Alberi check OK",
    "mach_det_id": null,
    "order_id": 2,
    "start_time": "2020-10-23T19:57:39.000Z",
    "stop_time": "2020-10-23T19:51:40.000Z",
    "task_status_id": 3,
    "task_comment": null,
    "agv_id": null,
    "oper_id": 2,
    "error_time": "2020-10-23T19:57:34.000Z",
    "component_id": null,
    "task_type_id": 2,
    "task_ref": null,
    "create_time": "2020-09-01T21:44:47.000Z"
  },
```

Figure 37: Tasks

Each order and task is characterized by the above structures and they are then processed to be transformed in the corresponding objects.
Some important attributes for the order are beside the description that explain in a human understandable language which type of order is considered, the property

*order_work_area_id* tells the id of the working area to which the order is associated. This information is especially meaningful for the HMI which needs to represent the status of the processes for each working area.
On the other hand a peculiar attribute for the Task object is the *task_status_id*. This attribute defines which is the status of the task that can fall in one of the following cases:

- Created
- Stopped
- Pending
- Executed

The relation between a Task and an Order is many to one.
Once this data are obtained the process starts with a logic as similar as possible to the one already presented in the chapter related to the use case A in part I.
For each activity performed either by the AGVs or by the operator a call to a specific API is performed.
The method of the ClientRestAPI instance that is called to performed this task is *activate*. This method is defined in order to be as generic as possible. By simply looking to the prototype of the function it is indeed possible to understand that this method can be used to contact whichever API is necessary.
For the use case, and in particular to communicate that a task either has been performed or a problem has been encountered the API that are contacted are listed below:

- `http://icowms.cloud.reply.eu/Details/updateStatusOk?task_id=96`
- `http://icowms.cloud.reply.eu/Details/updateStatusEr?task_id=96`

If no error is raised then the process as in the main scenario of the use case A, on the other hand if an error is encountered from the simulation point of view after calling the corresponding API to communicate such state, the color respectively of the shafts or of the motor is changed in red to immediately visualize it.

## Statistics

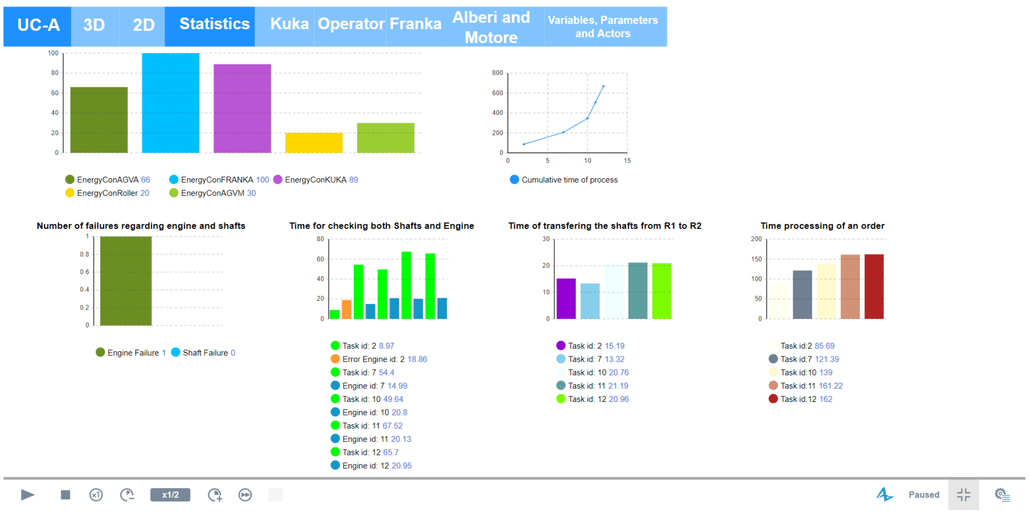


Figure 38: Statistics

The section related to the collection of statistics is of a crucial importance because according to the data collected it is possible to gather insights about the process itself, and have an idea about the effectiveness and feasibility of the process.
Moreover they are useful for the stakeholders in order to take decisions regarding the possible investment in such a solution or not.
The reason is because, through the collection of data, it is possible to understand the performances of the solution proposed taking into consideration that all is done with the minimal economical risk, since the cobots and the overall process have been virtualized.
The data collected take care of reporting different aspects of the process itself.
As showed in the picture five different charts have been reported for the simulation process.
In particular, the first one tries to represent the energy consumption of the different AGVs and cobots plus the energy consumption of the roller.
This insight it was decided to be reported even though data have not been yet collected in the moment in which this work was written (it could be considered as a further improvement in the collection of statistics or simply it was implemented later), because it is a relevant factor in defining how much AGVs need to be present in the collaborative space in order to have a long lasting service.
The second chart, is a cumulative time plot of the time spent for performing all the tasks that constitute an order.

This is a relevant measure to understand the amount of time that is partially saved thanks to the fact that it is spent from the cobots whereas the operator perform more relevant tasks: the checks performed on the shafts and the assembled engine and the mounting of the oil pump on the engine.

The third chart on the other hand gives a representation of the number of failures that could be be encountered during the process.
The number of failures here depends upon a probability value according to which the failures on the shafts and on the engine happen.
Obviously by changing this value different statistics and insights can be collected pointing out relevant aspects of the solution with different configurations.
In particular, among the plethora of configuration tried out in this thesis work are presented two opposite configuration that are able to show peculiar aspects of the process: one where the probability for the shafts to be correct is 0.3, and another where the probability is 0.8.
The insights obtained on both the configurations will be reported in the results section.

The fourth chart, is a bar chart where it is reported the time spent in checking both the shafts and the engine. In order to be more complete, it was decided to consider also the time wasted in the error case and not only the one belonging to the working scenario, in order to better understand the magnitude of a failure.

The fifth graphic instead report the time spent to transfer the shafts from Franka to Kuka creating a decoupling between the assembly and the logistic lines.
This measure was collected in order to compare it to the other ones.
The idea was to identify if present a bottleneck considering all the activities in the process, but the results of this analysis will be better explained in the results section.

Last but not least, a bar chart reporting the amount of time required for executing all the tasks in each order.
The idea behind the collection of this measure was to understand which was the amount of time spent for the processing of each task in a single order to see possible fluctuation of this value in different orders.

# ICOSAF-HMI

Regarding the ICOSAF project, apart from the simulation another important aspect has been taken care about: the development of an HMI.
It's indeed to answer the operators' necessities of visualizing all the necessary data for taking decisions and for reporting the status of the process that a web application has been developed.
In order to facilitate the visualization of what are the possible alternative processes and to easily visualize a dashboard with data related to the corresponding process, when the client application is used for the fist time, the first decision to be made is among four possible alternatives that corresponds to the four use cases in the project. Use case B and D dashboards are not yet implemented because the focus of Concept Reply was mainly about the use case A and C.

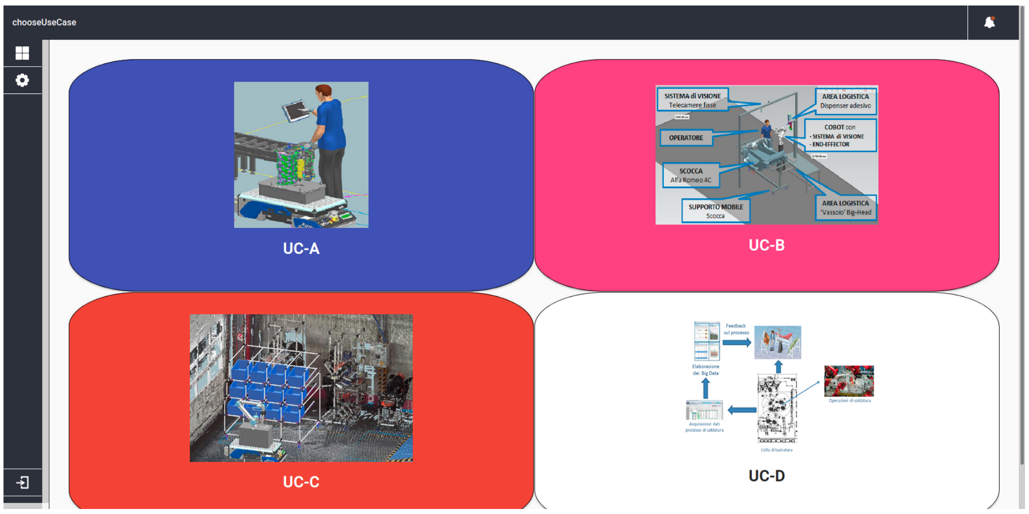

chooseUseCase
UC-A
SISTEMA di VISIONE Telecamere fisse
AREA LOGISTICA Dispenser adesivo
OPERATORE
COBOT con - SISTEMA di VISIONE - END-EFFECTOR
SCOCCA Alfa Romeo 4C
SUPPORTO MOBILE Scocca
AREA LOGISTICA "Vassoio" Big-Head
UC-B
UC-C
UC-D

Figure 39: Use Case Choice

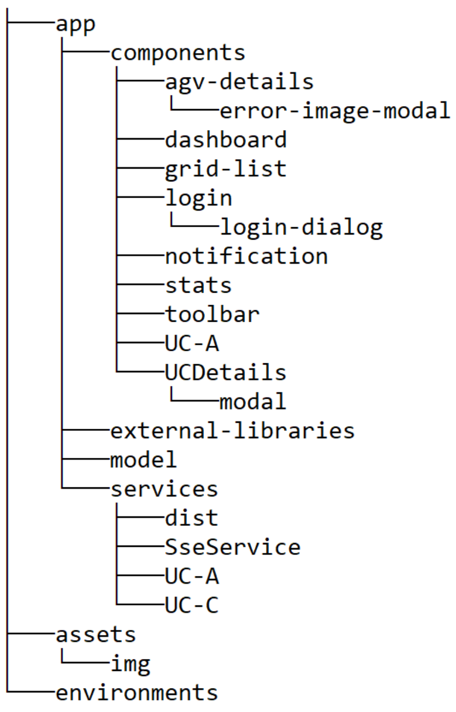
As said in the part II, the theme and the organization of the visualization were defined thanks to the work of a designer involved in the project whereas the implementation of the SPA on the other hand was assigned to Reply Santer.

The architecture of the application is organized in folders as follow:

1. components
2. external-libraries
3. services
4. assets
5. environments

The *components* folder contains the different angular components that were inserted in order to implement all the requested functionalities.
Each component take care of a specific portion and functionality of the application and those make use of *services* defined in the same named folder, to collect the needed data by contacting the server.
*External libraries* were introduced in order to enable specific functionalities like the charts for showing statistics.
In the *assets* can be found all the needed resources as the images and the icons provided by the designer involved in the project.
In the *environments* folder, environment related features were defined like the name of the server, in order to easily change it in the future.

A tree structure of the project is here reported:



```
├───app
│   ├───components
│   │   ├───agv-details
│   │   │   └───error-image-modal
│   │   ├───dashboard
│   │   ├───grid-list
│   │   ├───login
│   │   │   └───login-dialog
│   │   ├───notification
│   │   ├───stats
│   │   ├───toolbar
│   │   ├───UC-A
│   │   └───UCDetails
│   │       └───modal
│   ├───external-libraries
│   ├───model
│   └───services
│       ├───dist
│       ├───SseService
│       ├───UC-A
│       └───UC-C
├───assets
│   └───img
└───environments
```

Figure 40: Web App Architecture

The app Component and the routing module are defined at this level, the first is the bootstrap component and it is defined in such a way to contain all the shared features and elements like the sidenav and the toolbar, while the latter is a particular module used inside Angular to couple a particular state of the application represented by a specific URL and a specific component loaded when necessary.

In the implementation of the sidenav and toolbar general functionalities are provided by clicking on specific buttons like the the possibility to perform the login, to see the list of pending notifications received, the possibility of navigating towards a setting page (not yet defined by the partners) or towards the dashboard, and a sort of status reminder that provide information about what is your position inside the application.

Moreover inside the structure of the app component a router outlet was defined. This particular element acts as a placeholder that Angular dynamically fills based on the current router state.

At the beginning by contacting the server, the app component is loaded and the router-outlet is replaced with the component *ChooseUseCase* which provide the possibility to the user to select among one of the possible use cases.
Once the navigation is performed by clicking on one of the buttons, the corresponding component paired in the routing module is loaded and replaced to the router-outlet.

When the remote user choose the use case A, the Dashboard component is loaded and all the needed information will be loaded by contacting the backend.
In the implementation of the dashboard (one of the main components inside the application), the aim was to represent the list of the working areas involved in the process depicted in the use case A, their composition in terms of AGVs belonging to the working area and for each of those their status.

In order to implement it, the list of working areas were realized by means of cards on the left side of the screen, where their composition were reported and the status of the different AGVs was represented by means of bars and icons.

The status of the AGVs is a crucial information to provide to the remote operator, and this is highlighted also by the choices in the representation (colors and icons).
With this information, the remote operator can have at a first glance an idea of what is happening in the different working areas and what are the current status of the different AGVs involved in the process.

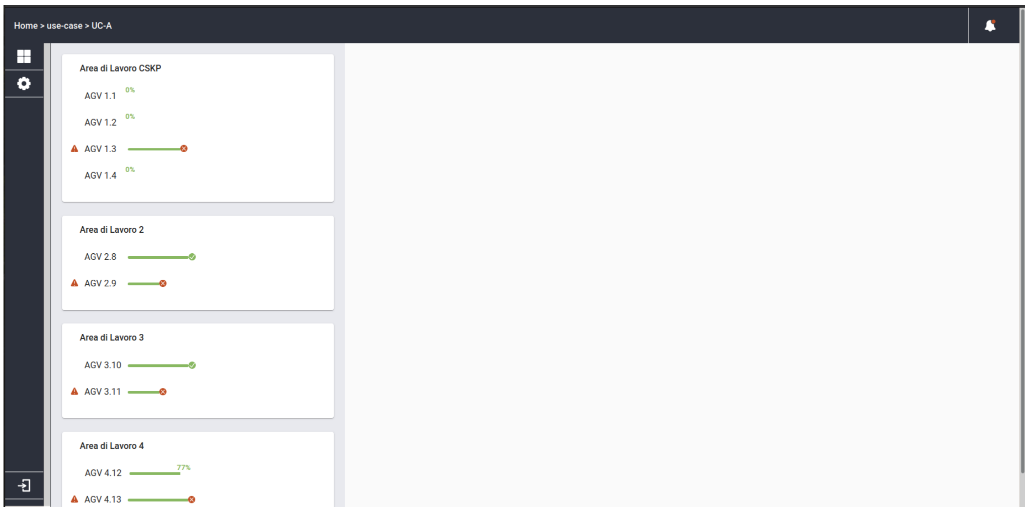


Figure 41: Dashboard

The entire application has been structured in such a way that it is intuitive and explicative. On the right in the HTML structure a named router-outlet (placeholder) has been inserted in order to provide a peculiar style of navigation where the dashboard remains where it is and the content on the right dynamically change according to the application status.
A named router outlet instead of a normal one was defined in order to make a distinction between the one defined in the app component and the one defined in the Dashboard.
If the remote user would like to have an insight about statistics on the process of a particular working area it has only to click on a card.
The state of the application will be changed, the related data needed for the statistics will be loaded and the corresponding component will be loaded replacing the named router outlet.
A picture about the page with the statistics visualized is here reported:

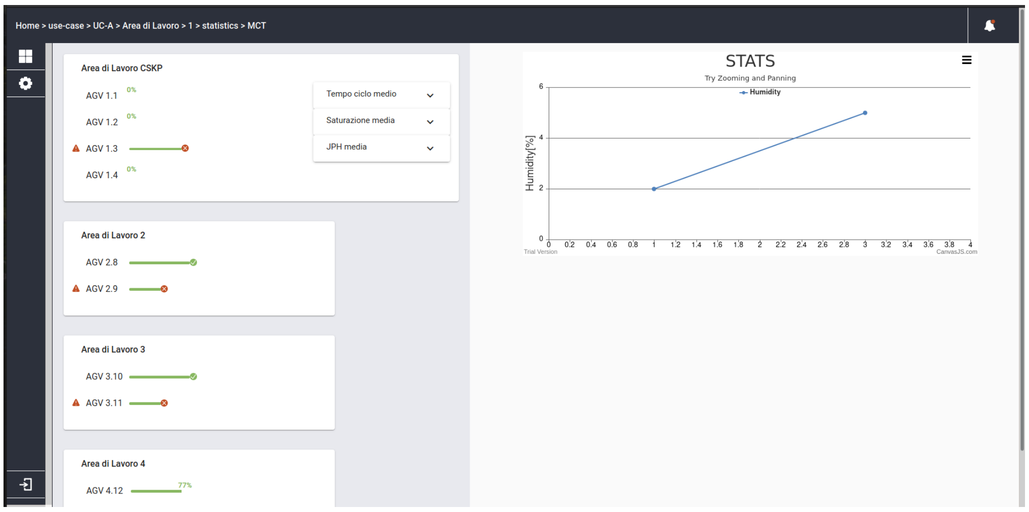


Figure 42: Web Application Statistics

As it can be seen also in the picture, in the current state of the project since it is not clear what actually are the information to be here reported a meaningless chart is reported, only to show that something similar will be shown but with real and meaningful data. One of the most important parts of the application comes when the remote operator clicks on a particular AGV.
When performing such an action the *AGV-Details* component is loaded and two expansion panels appear on the right side of the screen in a specific order.
First there is the Errors panel (”Problemi Rilevati”) which contains all possible errors happened in the simulated environment during the execution of the different tasks of the AGV which will be populated as soon as an updateStaturErr is received by operator in the simulation.

In the second panel instead the list of the tasks that have been executed by the selected AGV is reported.

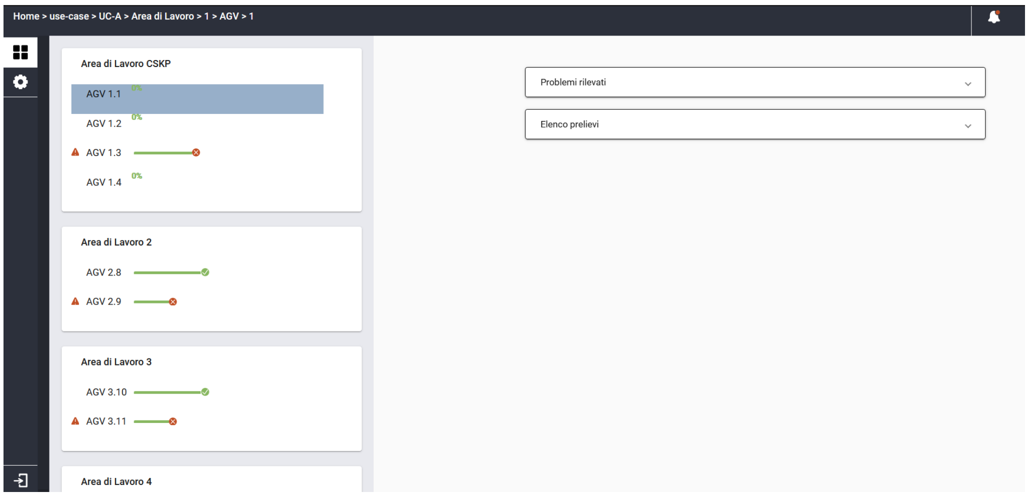
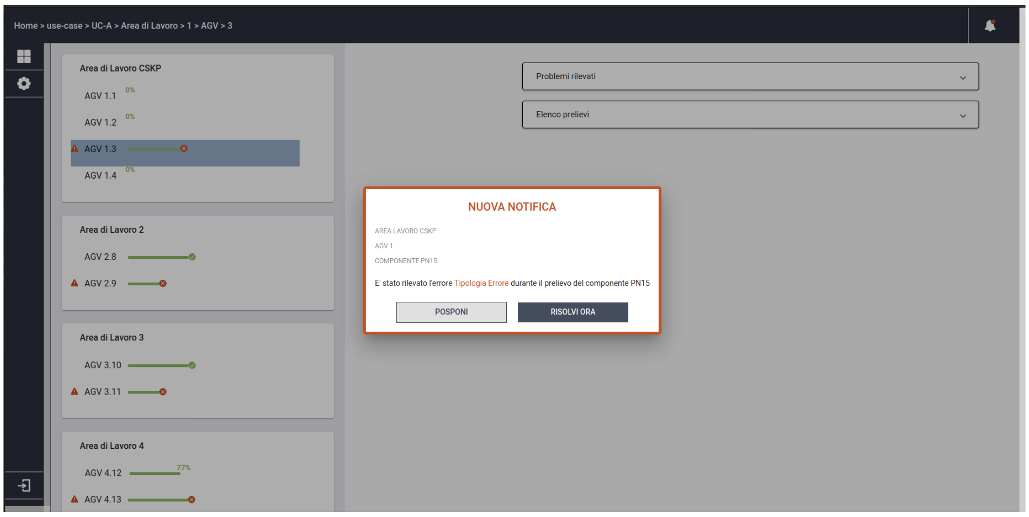


Figure 43: Panels

The work area interested by the use case A is CSKP, whose name stands for *Counter-rotating Shafts Kit Preparation.*
As it can be seen in the first picture it is possible to notice that the first AGV has not performed any task yet and the percentage reported in the card is 0%.
While the process proceeds and some tasks are fulfilled the percentage reported in the work area changes and the list of tasks fulfilled will be reported in the corresponding panels.
If during the execution of a task an error is raised a dedicated notification will popup as can be seen in picture 44.
In the popup information about the type error, the location, the AGV involved and the type of component which caused the error are provided and two possible alternative action can be performed: to postpone the resolve activity or to immediately take care about the error.
If the first alternative is chosen, then the notification is appended to the list of notification accessible on the top right corner through the bell icon (functionality not yet implemented because no drafts were provided), the list of errors of the AGV is refreshed and the status in the card is correspondingly changed in order to make the remote operator notice the error.
If on the other hand the *Risolvi ora* option is selected, a navigation inside the application to the details of the error is performed in order to give the possibility to the remote operator to actually resolve it.



NUOVA NOTIFICA

AREA LAVORO CSKP

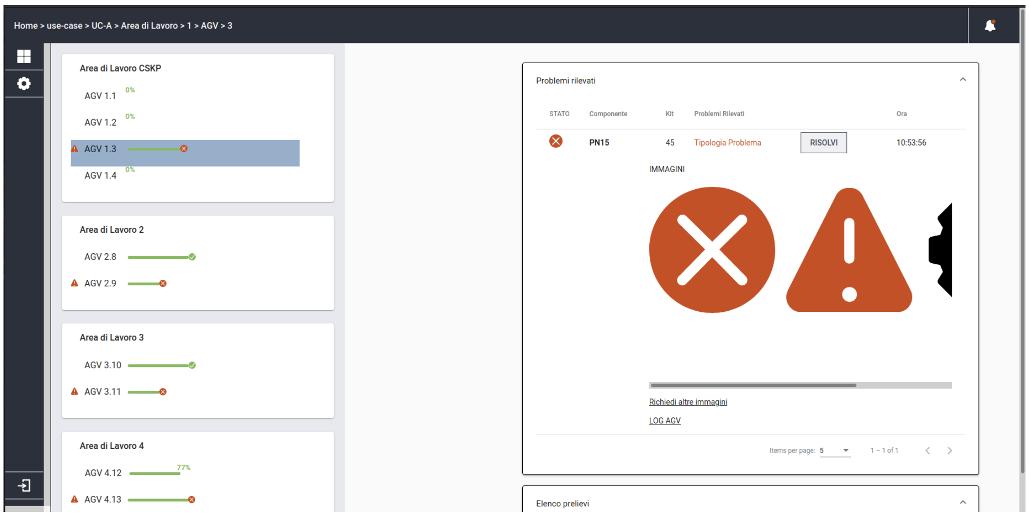
AGV 1

COMPONENTE PN15

E' stato rilevato l'errore Tipologia Errore durante il prelievo del componente PN15

Figure 44: Notification

This popup functionality is provided by the *Notification* component which is loaded whenever an error is generated and communicated by the server.
The mechanism that has been chosen for this kind of communication is based on the SSE whose acronym stands for Sever Sent Event.
Through this mechanism a push notification system was developed and the status of the different areas and AGVs are refreshed whenever a new event is defined.
Whenever an error is generated if available also the corresponding images that can be helpful to have a clearer idea of what happened are provided to the remote operator.



| STATO | Componente | Kit | Problemi Rilevati | | Ora |
|---|---|---|---|---|---|
| ✖ | PN15 | 45 | Tipologia Problema | RISOLVI | 10:53:56 |

Figure 45: Problem details

As can be seen in figure 45, also here a mock representation of error images are provided, but in a real context they will be useful to the remote user to have a deeper insight.
Furthermore it is possible to see more in details the error image and in order to see that a dedicated component is loaded: the carousel.

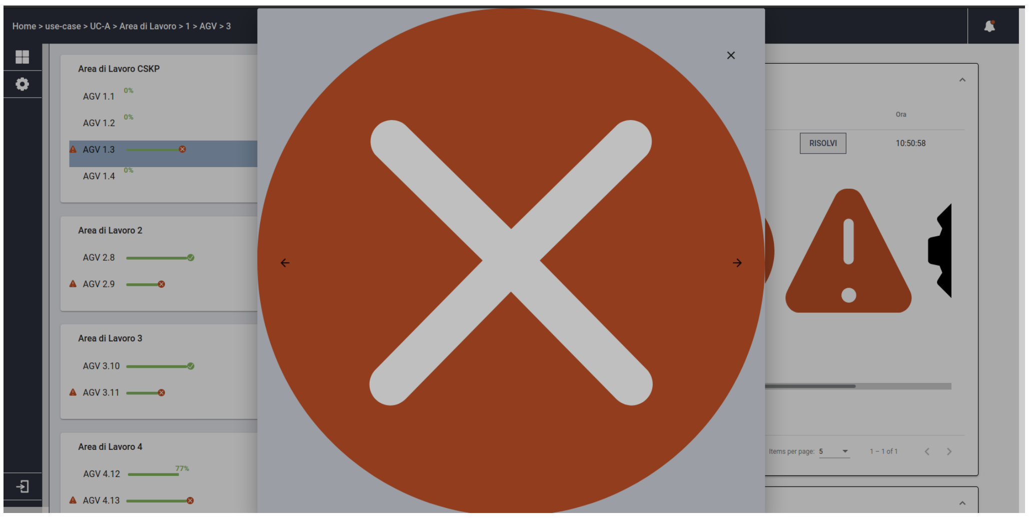
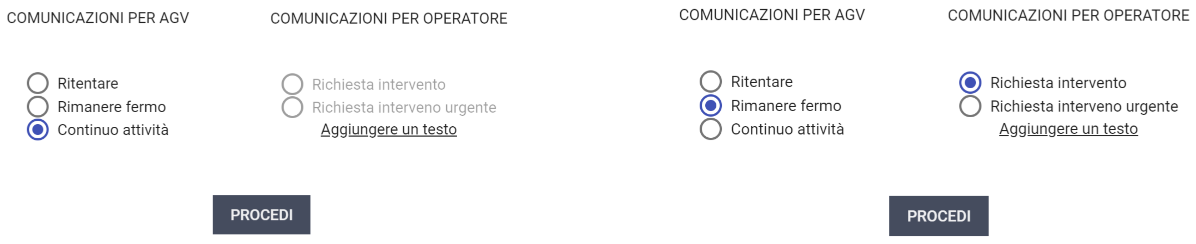


Figure 46: Mock image of a problem raised in the process

Apart from the functionalities useful for understanding the problems, it is given to the remote user also the possibility to resolve them.
The different options that have been defined in this section were defined according to the information provided by the company partners.


COMUNICAZIONI PER AGV

- ○ Ritentare
- ○ Rimanere fermo
- ● Continuo attività

COMUNICAZIONI PER OPERATORE

- ○ Richiesta intervento
- ○ Richiesta interveno urgente

Aggiungere un testo

PROCEDI

COMUNICAZIONI PER AGV

- ○ Ritentare
- ● Rimanere fermo
- ○ Continuo attività

COMUNICAZIONI PER OPERATORE

- ● Richiesta intervento
- ○ Richiesta interveno urgente

Aggiungere un testo

PROCEDI

Figure 47: Possible solutions to a problem

As it can be seen two different communications can be sent, one to the AGV that have default actions to perform like:

- Try again
- Stand still
- Continue activity

The option "try again" is used whenever the remote operator understand that the problem that was raised can be solved with the AGV that try again its task, and obviously this kind of solution is adopted when the entity of the error is not so important and the issue can be solved autonomously.
On the other hand the "continue activity" option is used whenever the problem in some way have been already considered and the activity of the AGV can be resumed.
Last but not least the "Stand Still" option is used when the entity of the problem is more severe.
Here the AGV cannot continue its own activity or cannot try again to solve the issue, but the only thing that it can do is to wait for the intervention of an operator to fix the problem.
By selecting this option indeed it must be selected also a communication to the operator and a further description of the problem can be added in a dedicated section.

Whatever action is performed the server is contacted and an update with the decision taken is made in order to take track about the status of the process.

The status of a task that is processed by an AGV can fall under three possible alternatives:

- Failed
- Success
- Error Solved

This solution can be affected by some changes during the evolution of the process but for the moment those are the only possible alternatives.
For each status a corresponding icon has been assigned:

(a) Status failed

(b) Status success

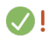

(c) Status error solved

Figure 48: Icons representing the status of a task

# DiPreTreat - Web Application

Concerning the DiPreTreat project it was asked to create and develop a web service able to monitor the problem of corrosion.
The development of such application was made as said by using the Angular Typescript based open-source platform for building mobile and desktop web application.
The landing page when accessing to the application is:

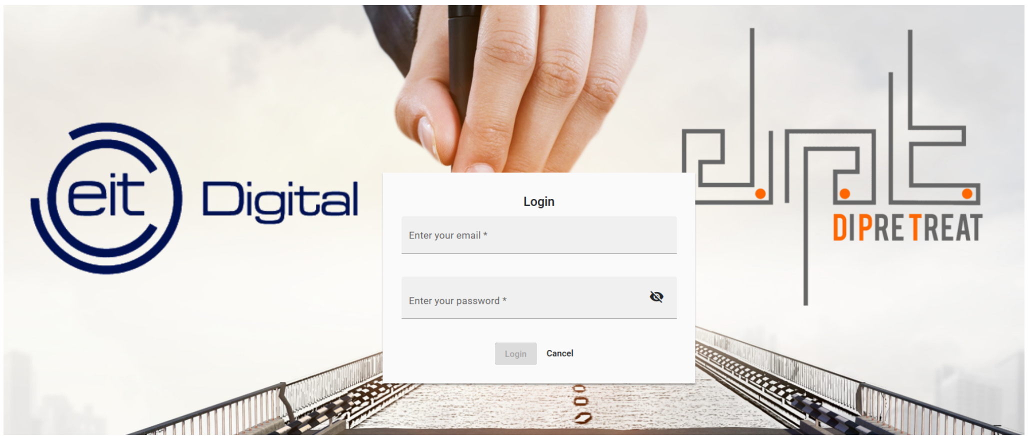


Figure 49: Landing page

As it can be noticed from figure 49 the landing page is characterized by the logo of the project and a form where the user can authenticate himself in order to access the service provided.

After the authentication a very simple and intuitive page is shown where the user has to take a first decision.
The user can either, by selecting a particular country, decide to visualize all the premises in the chosen region or on the other hand can select a particular premise by name and visualize its details.
The choice is made possible by the definition of two drop down list, one containing the list of countries and the other containing the list with the names of the premises.

Choose a Premise
Choose a Country

Figure 50: Choose premise

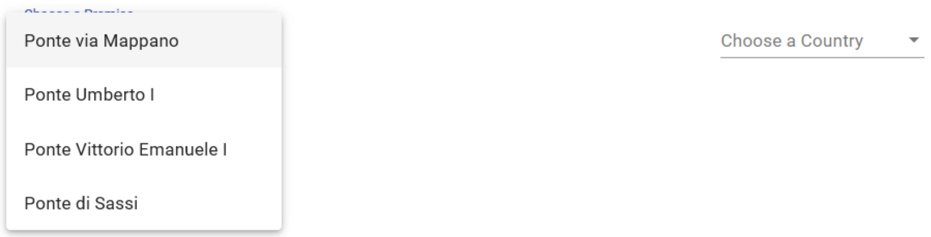
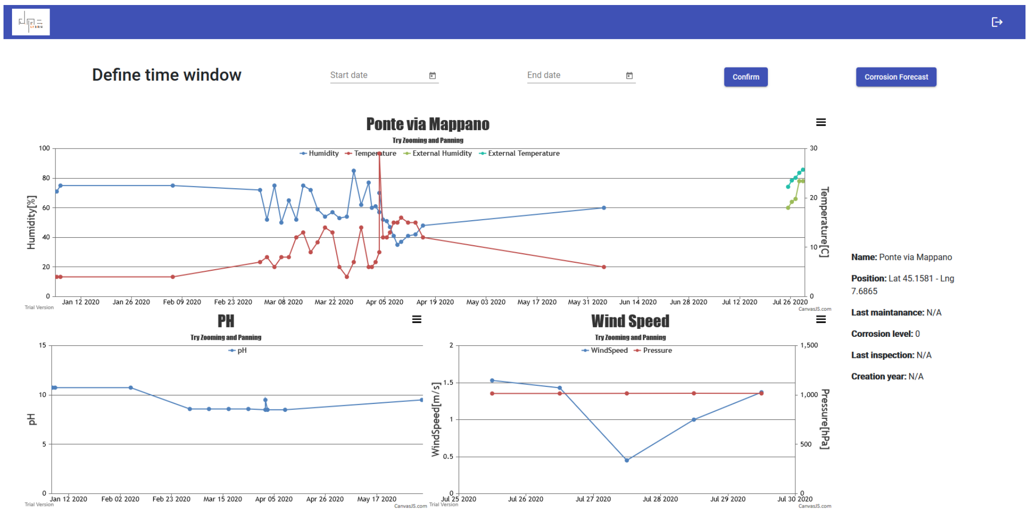


Figure 51: Choose premise

If a particular premise is selected than by default the data related to the last month are shown in dedicated graphics.
In particular four main factors are monitored:

1. Humidity
2. Temperature
3. pH
4. Wind Speed
5. Pressure

Those environmental factors are measured both from the internal and the external of the buildings and are reported with different colors in the charts.



**Name:** Ponte via Mappano

**Position:** Lat 45.1581 - Lng 7.6865

**Last maintanance:** N/A

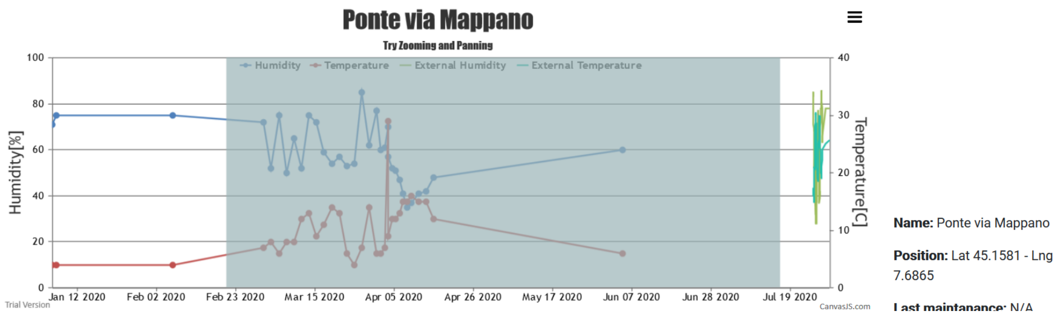
**Corrosion level:** 0

**Last inspection:** N/A

**Creation year:** N/A

Figure 52: Statistics measured about the environmental factors

As it can be notice the user by defining the time window by selecting a start date and the end date can have an insight about a specific period in time, and can understand if a particular factor was more influential in deteriorating the reinforcement steel of the monitored building.
On the right, it was decided to show all the specifications about the premise considered like the name, its position, its creation year and information related more specifically to the problem like the corrosion level which tells how badly this phenomenon is affecting the considered building and the last inspection which tells the last moment in time where a physical inspection was made.
Furthermore it was given to the user the possibility not only to see, but also to interact with graphics, by allowing the possibility of zooming in and out in order to have a more detailed or a shallower view about a specific period.
An example is reported in the following picture:



**Name:** Ponte via Mappano

**Position:** Lat 45.1581 - Lng 7.6865

Figure 53: Example of interaction with the graphics - Zoom in the Stats

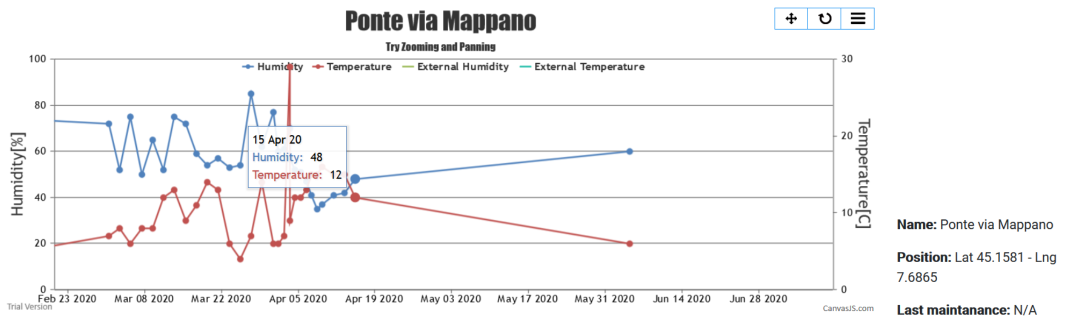


Figure 54: Example of interaction with the graphics - Details info

Furthermore the web service offers the possibility for requesting a forecast for a specific period in time.
Indeed by choosing the corrosion forecast option, a new module will be opened with the same structure and architecture of the previous page where the user can define the time window and receive the forecast for the specified period.
This functionality, shown in the figure 55, is provided in collaboration with the company Pipple.
Specifically, there was a whole negotiation on formats, on the type of requests and on how to receive possible forecasts: whether to view them directly in already implemented graphics, or whether to receive only the data necessary to recreate it.

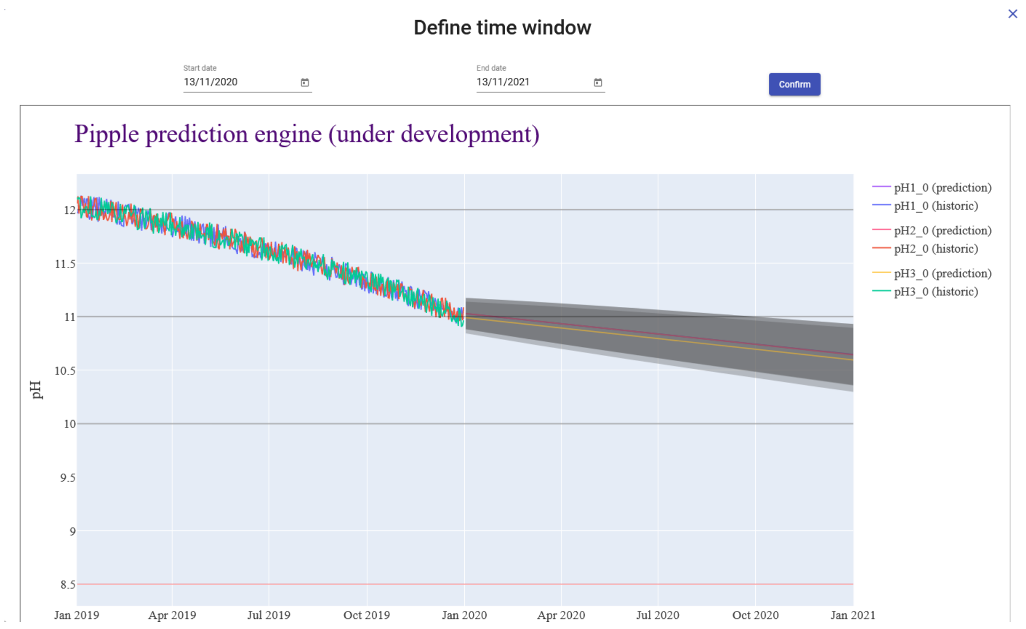


Figure 55: DiPreTreat - Corrosion Prevision

There were many changes during the development phase about all those aspects for the functionality to be provided, but at the end, an in the middle solution for both the companies (Reply-Pipple) was found.
The format chosen was JSON, the data are directly collected by requesting them to the Reply server and the forecast is directly sent as an image to the client which has the responsibility of visualizing it.
On the other side, if at the beginning the user selects a country, then a dedicated section is loaded where a geographical representation of the region of interest is depicted and where the user can actually have a clearer idea about the buildings that are monitored.
The priority of the visualization web service is to provide the user with as much information as quickly as possible from which stakeholders can make decision about allocating their resources.

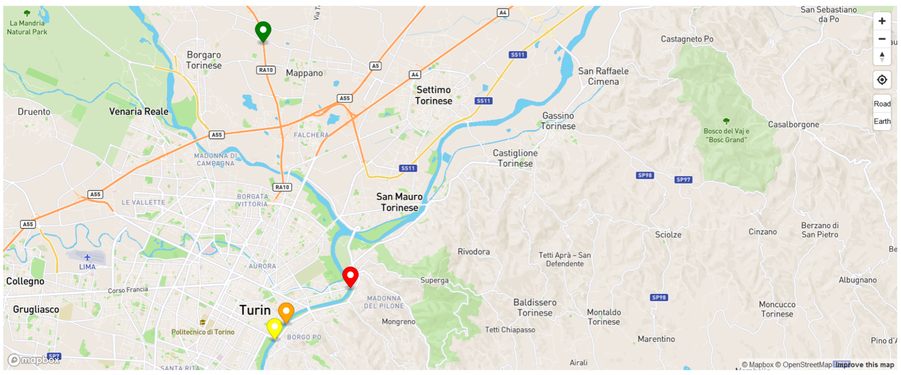


Figure 56: Geographical representation of the region of interest-Street view

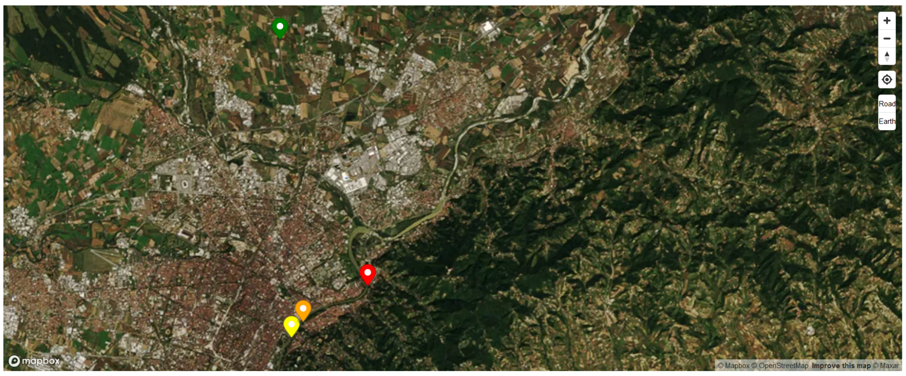


Figure 57: Geographical representation of the region of interest-Earth view

As it can be notice from the previous figures the user can select to visualize the region either in *Maps* style view, or in *earth* style.

The several facilities that are highlighted with different colors based on the gravity of the corrosion level and the possible values can be:

1. No deterioration (Green) - the construction shows no signs of deterioration either because it has just recently been inspected or because the sensor data indicates the structure is in good shape.

2. Starting deterioration (Yellow) - the structure is at the early stages of deterioration and inspection of the structure is advised.

3. Advanced deterioration (Orange) - the structure is at an advanced stage of deterioration and needs immediate inspection.

4. Critical (Red) - the structure is at a critical level of deterioration and needs immediate maintenance.

In the picture only few facilities are reported because it is a representation based on mock data, but obviously in the real context of use there will be more buildings monitored.
In order to have further details about the facilities the user has only to move the mouse on the corresponding building and a details window will popup showing the data corresponding to the selected premise.
Moreover by selecting one of the building the data corresponding to the premise will be shown in the previously described section with dedicated charts of the environmental factors.
The architecture of the application is organized in a similar way to the one seen in the ICOSAF project, indeed as it can be seen in the following picture where the organization in folders is reported, it can be notice that there are not so many differences.

```
.
├───app
│   ├───auth
│   ├───components
│   │   ├───di-pre-treat-map
│   │   ├───di-pre-treat-premise
│   │   │   └───forecast
│   │   ├───di-pre-treat-toolbar
│   │   ├───dipretreat-login
│   │   ├───dipretreat-menu
│   │   ├───home
│   │   └───not-found
│   ├───external-libraries
│   ├───model
│   └───services
│       └───premises
├───assets
└───environments
```

Figure 58: Tree Structure of the *DiPreTreat* project

As it can be seen the folder organization is very similar, each component in the *components* folder focuses in providing specific functionalities in the web service, the different *services* manage the exchange and the processing of the data, the different elements, in the *auth* folder, focuses in the management of the security for the authentication service used at the beginning from the user with a form, in the *model* folder, the classes representing the JSON entities received from the server are defined and in the *assets* and *environments* there are respectively the static resources and the shared variables (e.g. name of the server).
From an architectural point of view, the application is simpler with respect to the one implemented in *ICOSAF* because here the structure is flatter.
For each functionality most of the times, a dedicated component is loaded which take the place of the router outlet defined in the App Component (the component that Angular defines by default) according to the rules defined in the routing module.
Among the several difficulties that can be found in the implementation of a Single Page Application, the one that took more time to be overcome in the DiPreTreat project, is the implementation and definition of a component able to depict a geographical representation of a region.
It is something not so common, especially for Angular projects, indeed most of the solutions that were found and tried out during the development were not 100% functioning or deprecated.
It took some time to find a valid alternative still supported and that satisfied all the requirements asked in the DiPreTreat project.

# Third Part

# Results

According to the purpose that was defined in the dedicated section, the objective for the ICOSAF project was to implement a Digital Twin of a process controlled with a dedicated HMI able to communicate with the surrounding system and actors in order to process the incoming orders in the work area while for the DiPreTreat project was to develop a web application able to support the monitoring of a problem afflicting many facilities: the corrosion of the reinforcement steel.
Regarding the ICOSAF project, the objective of implementing a pilot Digital Twin for a process was to evaluate the feasibility and the effectiveness about the modeled process.
The modeled process has been simulated taking into counts all the prescriptions defined by the partners involved, and the corresponding HMI has been implemented in such a way that it allows to communicate in both directions, from the twin process towards the HMI and vice versa in a fashion that is as close as possible to the one that there will be between the real process and the HMI.
Since the simulated process has been defined in such a way to respect all the characteristics of the process itself, the results that came out from the collected data can be considered trustworthy.
Considering those premises, according to the objectives that have been defined, it can be affirmed that even though the solutions proposed have margin for improvements, anyway a solid starting point where the pillars and basis for the development of a digital twin have been already laid.
The solution implemented with Anylogic which emulates the real world process, gives to the stakeholders significant insights about several aspects of the process itself. It gives them an idea about the requirements needed for implementing such a collaborative space and gives them also an idea about the effectiveness of the solutions developed.
In particular, regarding the effectiveness of the process as said in the section regarding the statistics collected, two main configurations in the simulation have been considered where a relevant parameter has been changed: the probability of succeeding in the checks performed by the operator, with two opposite values, one where the probability of success is 0.8 and another where the probability is 0.3.

Through the two different configurations, it was possible to understand different aspects of the solution.

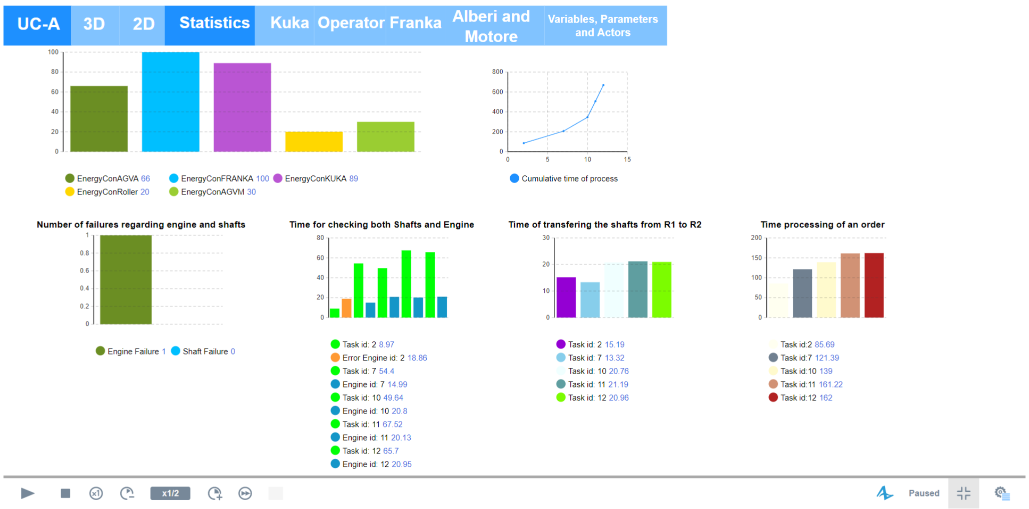


Figure 59: Statistics section - working process P = 0.8

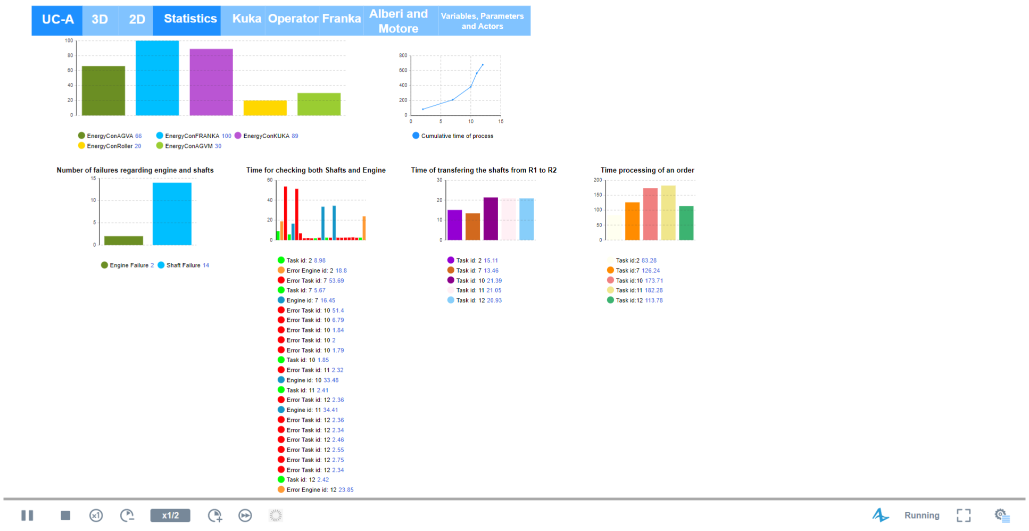


Figure 60: Statistics section - working process P = 0.3

In particular, for both given the data collected it was possible to understand that there is a bottleneck in the process and it regards the decoupling of the assembly and the logistics lines.

Indeed, from the data collected reported in the two pictures, it is possible to understand that the amount of time spent from the cobots to transfer the shafts is almost twice the time necessary for the operator to perform the check on the shafts. This observation, led to an important conclusion: it is not necessary to add more AGVs to gain a relevant advantage in terms of time, but a speed up in the transfer of the shafts can be instead considered.
This can be obtained either with an higher speed from the cobots of Franka and Kuka involved in the task, but it must also be consider that an higher speed could lead to an higher number of errors, or through the insertion of another cobot helping in the transferring of the components.
The advantage anyway could be derisory, and a lot of aspects in the collaboration among the cobots should be taken in consideration: the amount of space required, their speed, the synchronization among the different cobots etc.

Furthermore by taking a closer look to the results obtained with both the configurations, it can be observed that having two couples of AGVs is relevant to speed up the execution in the working case but more importantly in the error scenario, where thanks to the the second AGV the impact of the failure is significantly reduced.

The insertion of the cobots in the process can be considered fully justified considering the fact that they will take care of repetitive and low added value tasks, letting the operators focusing on more relevant tasks such the control of the validity and quality of the assembled engine or the correctness of the shafts.

Regarding the HMI is now possible to evaluate and validate it with real time data and with realistic communications since the Digital Twin developed is representative of the collaborative space it simulates.

As said there is large margin for improvements from both sides, the simulated environment and the implemented HMI, because for the first more aspects can be taken into consideration to have a more detailed representation of the emulated process, for instance more out of the ordinary cases can be analyzed and regarding the behaviours of actors, in particular the one of the operator that intervenes for solving problems, can be defined more in details. For the HMI instead, other functionalities can be introduced in order to provide other services to the operators such a communication between the two operators, and furthermore a mobile application could be developed for the smartwatches and small devices, in order to allow the usage of the HMI also from those devices.

It's anyway a starting point from which some insights have been already pulled out about the speed of the AGVs in the collaborative space while transporting the kits that needs to be not to high in order to allow the collaboration with the operator, the speed of the two robotic arms in particular for the decoupling of the two lines, the amount of cobots to be used with respect to the presence of a single operator in the shop floor, because a too high number of AGVs cannot be worth it in terms of advantages in the performances considering the investment, and other information about the process itself, like statistics about the number of errors that there can be, the amount of space required for allowing such collaboration among cobots and humans and so on.

Regarding the DiPreTreat project instead, a web service in the context of Industry 4.0 has been developed, which based on the data collected from a network of sensors installed in several facilities provides an application that helps in monitoring this kind of phenomenon providing all the functionalities defined by the partners involved in the project.
The aim of the application was to provide in an efficient and effective way all the information deemed necessary to understand the problem coming from the data collected and the analysis performed.
Standing to the actual specification, the application provide all the functionalities deemed necessary for this purpose, but obviously other aspects can be further analyzed and developed.
Other services can be integrated in the web application in order to enlarge the support for the monitoring of the premises, and the same service can be developed also for other devices with native mobile applications.
Also a section to signal damages to the facilities can be developed in order to take track and be more effective in the identification of the problem.
According to the premises, it can be said that the result for the DiPreTreat project in supporting the monitoring and the decision process for taking preventive actions for solving the problem of the corrosion has been successfully reached with the developing of the dedicated application.

# Conclusions

In conclusion it can be asserted that the work of thesis has reached its main goals: regarding the ICOSAF project to develop a Digital Twin of a real world process able to interact with an HMI producing an integrated working environment that can be used in manufacturing for exploiting all the advantages of Digital Twin and of collaborative robots whereas for the DiPreTreat project to implement a web service in the context of Industry 4.0 for providing a way to monitor the problem of corrosion for concrete facilities.
For ICOSAF the evolution of the project is notable, indeed at the beginning of the work only a rough solution providing an isolated representation of the use case A process was provided, but during this work of thesis, the solution has been re-implemented, optimized and it has been characterized of an important functionality that was totally added from scratch: the communication with an external HMI .
At the moment a realistic 3D representation of the process has been created which can be controlled in several aspects by means of an HMI and this interaction perfectly emulates what happens when there will be real cobots in a collaborative environment. This is definitely a significant result in the perspective of industrializing the process.
In this context, the potentialities of the Digital Twin technology was used to produce interesting insights about the process itself.

Concerning the HMI development it must be said that is a web application totally written from scratch that was designed and implemented taking into consideration always the final user thanks also to the work of the partners involved.
Nonetheless it must be considered that the work is still not finished and there will be further improvements while the project will go on, by answering to the demands and the requests made visible from the partners involved and of course from CRF.

Regarding the DiPreTreat project according to the necessities that have been pointed out from the partners involved, a web application, in particular a Single Page Application has been defined and developed from scratch in order to fulfill the requirements defined.

The solution proposed tries to fulfill the goal in an effective and efficient way by modeling a modular and easy to use web service.
The collaboration with the partners for the definition of the service, related to the forecast, was a moment of negotiation for the partners involved and it was a moment of growth since compromises have been made.
The solution provided can be affected by changes and improvements in the future, while the project will go on in order to satisfy all the requirements that will come out, but according to the premises and the demands that came out, the solution proposed perfectly address all the necessities of the stakeholders.
A margin of improvement is still possible, with the insertion of new components in the web service in order to provide new functionalities, and the application can be developed also for small devices in order to broaden the audience of possible users.

# Bibliography

[1] LANE WARSHAW AARON PARROTT. *Industry 4.0 and the digital twin.* URL: https://www2.deloitte.com/content/dam/Deloitte/kr/Documents/insights/deloitte-newsletter/2017/26_201706/kr_insights_deloitte-newsletter-26_report_02_en.pdf.

[2] AGV. *Agv.* URL: https://www.indiamart.com/proddetail/agv-automated-guided-vehicle-10989805612.html.

[3] Angular. *Introduction to Angular concepts.* URL: https://angular.io/guide/architecture.

[4] Anylogic. *Anylogic Simulation Software.* URL: https://www.anylogic.com/.

[5] Inductive Automation. *What Is HMI.* URL: https://www.inductiveautomation.com/blog/sites/default/files/inline-images/What_Is_HMI-6.jpg.

[6] Rosen R. Boschert S. "Mechatronic Futures". In: Springer, Cham, 2016. Chap. The Simulation Aspect.

[7] Cedex. *front-end-development.* URL: https://www.cedextech.com/front-end-development/.

[8] Josephine Condemi. *Digital twin: cos'è, come funziona, esempi e vantaggi del gemello digitale.* URL: https://www.internet4things.it/industry-4-0/digital-twin-cose-come-funziona-esempi-e-vantaggi-del-gemello-digitale/.

[9] *csr.* URL: https://i.ibb.co/GMQ5KFM/ssr-ssg-csr.png.

[10] Stephen Ferguson. *Apollo 13: The First Digital Twin.* URL: https://blogs.sw.siemens.com/simcenter/apollo-13-the-first-digital-twin/s.

[11] Christian Findlay. *Traditional-Vs-SPA.* URL: https://i1.wp.com/christianfindlay.com/wp-content/uploads/2020/07/Traditional-Vs-SPA.jpg?resize=768\%2C532&ssl=1.

[12] FRANKA. *Franka Panda Emika.* URL: https://www.franka.de/.

[13] Ginger Gardiner. *Digital twin, digital thread and composites.* URL: https://www.compositesworld.com/articles/digital-twin-digital-thread-and-composites.

[14] Bhavika Garg. *Angular Architecture Overview.* URL: https://medium.com/@bhavikagarg8/angular-architecture-overview-1e7cc7483a0.

[15] Michael Grieves. *Origins of the Digital Twin Concept.* Aug. 2016. DOI: 10.13140/RG.2.2.26367.61609.

[16] ICOSAF. *Document of Work.*

[17] Kuka. *Kuka cobot.* URL: https://www.kuka.com/en-ch/products/mobility/mobile-robots/kmr-iiwa.

[18] Max Pekarsky. *Does your web app need a front-end framework?* URL: https://stackoverflow.blog/2020/02/03/is-it-time-for-a-front-end-framework/.

[19] ROS. *About ROS.* URL: https://www.ros.org/about-ros/.

[20] simulationx. *system-simulation-developmentprocess.* URL: https://cdn.simulationx.de/en/image/system-simulation-developmentprocess.png.

[21] simulationx. *Why System Simulation.* URL: https://www.simulationx.com/system-simulation/benefits.html.

[22] stateofjs. *Front End Frameworks.* URL: https://2019.stateofjs.com/front-end-frameworks/.

[23] Laura Zanotti. *Digital twin: cos'è e come funziona il modello del gemello digitale.* URL: https://www.digital4.biz/executive/digital-twin-cose-e-come-funziona-il-modello-del-gemello-digitale/.

# Ringraziamenti

Ringrazio il mio relatore, il professor Paolo Garza persona estremamente preparata, il quale ha dimostrato tanta disponibilità sia durante il periodo di insegnamento sia in questo periodo di tesi.

Vorrei anche ringraziare la dottoressa Liudmila Dobriakova, persona che stimo con cui ho condiviso questo percorso di tesi che è sempre stata super disponibile con aiuti e chiarimenti in tutte le fasi del lavoro.

Vorrei rivolgere un ringraziamento alla persona che mi ha indirizzato a intraprendere questo cammino infondendomi la passione per lo studio: il professore Giuseppe De Giorgi, il quale mi ha insegnato ad essere costante e a portare fuori le mie capacità.

Vorrei rivolgere un ringraziamento al mio padrino Bruno, persona che ho sempre stimato sin da quando ero piccolo e che vorrei rendere partecipe di questo traguardo.

Vorrei cogliere l'occasione inoltre per ringraziare i miei colleghi del Politecnico, in particolar modo i membri di quella che poi è diventata la "SbadaGang": Claudio, Giacomo e Marco, importanti amici che hanno condiviso con me l'intero percorso della magistrale tra studio e risate. A loro vorrei fare un sentito in bocca al lupo per il futuro.

Vorrei rivolgere un sentito pensiero ai miei amici, in particolar modo un ringraziamento va ai miei amici più stretti, i membri del "Night": Gioele, Paolo, Claudia, Elia e Ramona, con cui ho condiviso gran parte delle mie felicità e con cui mi sento sempre a casa. Vi voglio un mondo di bene ragazzi.

Vorrei dedicare questo traguardo ai miei nonni: nonna Rosanna, nonno Tonino, nonna Fiorigia e, anche se non più tra noi per condividere questo momento, mio nonno Nino, i quali sono stati sempre molto fieri di me, e in questo momento di felicità lo saranno ancora di più.

Vorrei ringraziare la mia ragazza Ramona, persona molto importante che mi conosce nel profondo, con la quale ho condiviso le più significative esperienze della mia vita e renderla partecipe di un simile momento di gioia non fa che riempirmi il cuore di felicità.

Per ultimi ma assolutamente non per importanza, vorrei ringraziare e anche dedicare questo traguardo a mio padre Giuseppe, a mia madre Giulia e a mio fratello Mattia, le persone più importanti della mia vita.
Grazie a loro infatti ho potuto intraprendere e raggiungere un simile traguardo.
Senza il loro supporto in tutti questi 5 anni di università, non sarei arrivato dove sono ora e non sarei la persona che sono.
A loro rivolgo tutta la mia gratitudine.