# Towards "Industrie 4.0" in the context of investment casting industry

*Isabel Maria Lousada Soares Figueiredo*

**Dissertação de Mestrado**

Tutor in FEUP: Professor Doutor Engenheiro Germano Manuel Correia dos Santos Veiga
Tutor in INEGI: Engenheiro Ricardo Paiva
Tutor in ZOLLERN & Comandita: Engenheiro Rui Félix

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

**Mestrado Integrado em Engenharia Mecânica**

Janeiro 2019

“When wireless is perfectly applied the whole earth
will be converted into a huge brain…all things
being particles of a real and rhythmic whole.”

Nikola Tesla, 1856 - 1943
(About the Internet of Things)

## Abstract

In a fast changing and ever-growing world, where technology is the future and being "connected" is what brings advantage, the industrial world follows the same rules, and the need to grow into the Industry 4.0 concept is what motivated this thesis.

The thesis will describe the manufacturing process of Zollern & Comandita in Porto, identify where this process could be improved and how and propose solutions for the shortcomings.

The preliminary evaluation performed in this work lead to the conclusion that the data extracted from Zollern & Comandita's manufacturing process is still short comparing to what the PLCs of the factory can control or display, and that the data that was being extracted was not being treated.

In an attempt to create solutions for this problem, three different possible data gathering processes were developed. One develops a solution to parse the data files that are already extracted and send its information to a database. The other two solutions develop a way to process this information directly from the machine's PLC and in real-time.

The first solution is an application C# programming language that reads, previews and writes the data from the files to a SQL database.

The second is a CODESYS program that simulates a PLC from a machine and writes its data to the same SQL database.

The third solution develops a CODESYS program that also simulates a PLC from a machine, which is connected to a C# application that reads the data from the CODESYS program and writes it to a Cloud database. This application also writes input values to the CODESYS program, thus allowing the control of the machines' process from a distance. As an intermediate stage to this solution, a simple C# application was developed to provide data that would simulate a machine and write the values to a cloud database.

The solutions were developed to present different ways to improve or solve the issue faced in the analytical stage of this thesis as mentioned above. With this work it is expected that the solution for this problem becomes clear and, in the end, help the company to find the best solution regarding the company's needs and the future of industrial technology.

The solutions don't represent the development of new technology, but nonetheless represent "the last mile" that enables different systems to interconnect to become a data driven and more responsive whole process, a prime objective of the Industry 4.0 concept.

## **Desenvolvimento de soluções para implementação do conceito “Indústria 4.0”** em fundição por cera perdida

## Resumo

Num mundo em constante mudança e crescimento, onde ter a melhor tecnologia e estar “online” é o que mais vantagem traz, o mundo industrial segue as mesmas tendências e a sua necessidade de crescer aliado ao conceito “Indústria 4.0” é a principal motivação desta tese.

Neste documento encontra-se a descrição do processo produtivo da Zollern & Comandita (situada no Porto) e um estudo sobre onde e como este processo pode ser aperfeiçoado.

Neste estudo inicial, chegou-se à conclusão que os dados extraídos do processo produtivo ainda são poucos comparando com os parâmetros controlados pelos PLCs e exibidos nas HMIs da fábrica, e mesmo os que são de facto extraídos, não são tratados ou analisados.

No sentido de resolver este problema, foram desenvolvidos três diferentes possíveis caminhos para a extração destes dados. A primeira solução desenvolvida, resolve o processamento dos ficheiros que já são extraídos dos PLCs e envia estes dados para uma base de dados.

As outras duas soluções vão no sentido de processar a informação dos PLCs diretamente e em tempo real sem que esta tenha de ser armazenada em ficheiros intermédios.

A primeira solução desenvolvida foi uma aplicação em linguagem de programação C# que lê os ficheiros, os dados que estes contêm são visualizados e montados em tabela e por fim a informação é escrita numa base de dados SQL.

Para a segunda solução foi desenvolvido um program em CODESYS que simula um PLC de uma máquina e escreve os dados daqui extraídos para a mesma base de dados SQL.

Na terceira solução, foi desenvolvido um programa em CODESYS que simulasse de novo um PLC, o qual foi posteriormente ligado a uma aplicação C# que lê os valores do PLC e os escreve numa base de dados em *cloud*. Esta aplicação também permite escrever *inputs* para o PLC permitindo o seu controlo à distância. Como etapa prévia a esta foi desenvolvida uma aplicação simples em C# que simula uma máquina e escreve os dados provenientes da mesma numa base de dados em *cloud*.

As soluções desenvolvidas no âmbito desta tese vão no sentido de apresentar à empresa várias opções possíveis para melhorar e/ou facilitar a captação de dados, acima mencionada. Esperando-se, portanto, que a solução para este problema se torne mais clara e, por fim, conseguir ajudar a empresa a encontrar a melhor solução, tendo ainda em conta o futuro da tecnologia no mundo industrial.

As soluções não representam o desenvolvimento de novas tecnologias, mas representam a "last mile" que permite que diferentes sistemas se interliguem tornando-se num processo uno impulsionado por dados e mais responsivo, um objetivo primordial do conceito Indústria 4.0.

## Acknowledgments

I would like to start, thanking everyone at Zollern and Comandita for all the help in data gathering and availability to answer all my questions, specially to Marco and Nuno in the wax injection department, Zé in the ceramics department, Bernardo in the foundry department, Rui Ferreira the PLCs responsible in the factory and Eng. Rui Félix the tutor from Zollern & Comandita.

Next, I would like to acknowledge everyone in the Zollern4.0 team, at INEGI, for all the help during this work, always giving good inputs to this thesis and letting me share ideas for the project. Naming, Eng. Ricardo Cardoso, Eng.ª Ângela Cardoso, Eng. Vasco Pinto and Eng. Ricardo Paiva the tutor at INEGI.

I would like to thank my tutor at FEUP, Professor D. Germano Veiga, for the support throughout the process, for the fast response to timing issues that helped give a direction to this thesis and for his help with the development of the software solutions.

Finally, I would like to thank all my family and friends. Specially, my parents and grandparents for all the support, patience, good advice, blind confidence, help reviewing this essay and giving me this amazing opportunity. To my uncles and aunts, for all the help in the research and during the solution development. To my siblings and my friends for always finding a way to reassure me and take some pressure off.

# Contents

# List of Figures

## List of Tables

## Abbreviations and Acronyms

| | |
|---|---|
| AGV | Automated Guided Vehicle |
| ASCII | American Standard Code for Information Interchange |
| CPS | Cyber Physical Systems |
| CSV | Comma Separated Values |
| FBD | Function Block Diagram |
| HMI | Human Machine Interface |
| I4.0 | Industry 4.0 |
| IL | Instruction List |
| IP | Internet Protocol |
| IoS | Internet of Services |
| IoT | Internet of Things |
| IRT | Isochronous Real Time |
| LAN | Local Area Network |
| LD | Ladder Diagram |
| MAC | Machine Access Control |
| NRT | Non Real Time |
| OPC | Open Platform Communications |
| OPC UA | Open Platform Communications Unified Architecture |
| OSI | Open System Interconnection (model) |
| PC | Personal Computer |
| PLC | Programmable Logic Controller |
| RT | Real Time |
| RTU | Remote Terminal Unit |
| SCADA | Supervisory Control And Data Acquisition |
| SFC | Sequential Function Chart |
| SQL | Structured Query Language |
| ST | Structured Text |
| TCP | Transmission Control Protocol |
| TCP/IP | Internet Protocol Suite |
| XML | eXtensible Markup Language |
| ZC | Zollern & Comandita |

# 1 Introduction

In this chapter the reader will be introduced to the project which this thesis was a part of, the technology associated with it and the thesis development throughout the time.

## 1.1 Project’s framework and motivation

This thesis emerges from a project in INEGI called ZOLLERN 4.0, which has the purpose of renovating and developing the ZOLLERN & COMANDITA factory in Porto, Portugal towards a more Industry 4.0 (I4.0) oriented production process.

## 1.2 ZOLLERN

The ZOLLERN GmbH &Co. was founded by the Prince Meinrad II of Hohenzollern-Sigmaringen in 1708, in the state of Baden-Württemberg, Germany. Three hundred years later the company is still run by the same family. Nowadays, the company employs about 3000 people around the world [1] [2].
In 1991, ZOLLERN & COMANDITA (ZC) was established just outside Porto, Maia, and its production specialises in investment casting parts. Nowadays, this facility employs more than 158 people and manufactures for a number of applications, such as, aeronautics and automobile, medicine or energy applications (Figure 1) [3].

Figure 1 - Zollern’s products [3]

## 1.3 Investment Casting process

This high precision casting process is developed in seven steps (Figure 2), the first is the wax injection, where a cast, with negative form of the object being produced, is filled with wax to create a model. In the next step, a previously established number of models are assembled (glued) to a large log of wax, creating a tree of wax models. The third step consists in bathing the wax tree in a ceramic paste as many times as required for that type of part, and drying each layer separately - in this part of the process it is sometimes necessary to sprinkle or plunge the tree into a sand bath while the ceramic layer is still wet to provide more stability to the ceramic shell. In the fourth step, the wax inside the ceramic shell is removed in a specific type of autoclave (Boilerclave® [4]) that reaches temperatures around 180ºC-200ºC and 10 bars of steam - dewaxing. After this, the liquid metal or the required material is poured into the ceramic cast and is let to set and cool for some time. When the material is completely hardened and cooled off the ceramic cast is removed, and all the residues washed off. Finally, the pieces are cut from the tree and final touches are taken care-of, for example deburring the connection to the tree, and the part is finished.

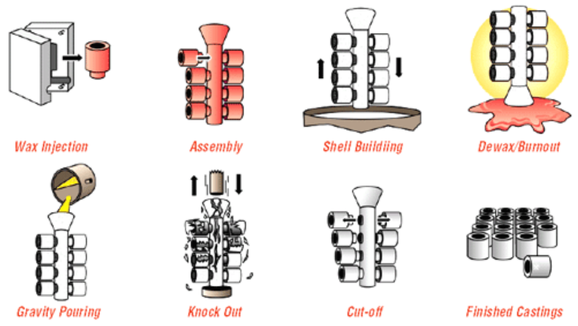


Figure 2 - Investment casting process (adapted from [5])

## 1.4 Industry 4.0

The concept Industry 4.0 (I4.0) was created by the German government around 2011, as a strategy to develop the industry and promote its computerization. In the present day, this term is also the denomination for the fourth industrial revolution [6] [7] [8].

One of the main differences that distinguishes the third from the fourth revolution is the level of automation, and its alliance to a highly customizable and flexible mass production, creating more customer-oriented products with less waste and, in the long term, less costly [9].

According to the concept, this will be accomplished through interconnection and interoperability between machines and humans, where all systems are connected with one another, communicate, share and analyse data in a common base, as happens with cloud computing. It will also depend on virtualization of factories, thus creating simulation models or virtual blueprints of the factories, enabling humans, or the machine itself, to monitor and control real-time data from the interconnected sensors around the facility(ies). Another key change that I4.0 will bring is technical assistance, decentralized decisions and real-time capability, in other words, the cyber-systems will be able to work independently, collecting real-time data, analysing that data and making decisions based on that data. For instance, if there is a malfunction with the machine, this one is able to delegate its work to another machine in the production line [10] [11].

All these factors will improve the modularity and service orientation of the industry, making it more adjustable to quick changes and more profitable.

While a standard factory needs some time to adjust to market fluctuations or product changes in manufacturing, a smart factory should be able to be more flexible regarding the customization of its products and adapt to the market’s needs easily and smoothly. This adjustment is only possible with the support of the internet of services [11].

### Cyber-physical systems

Cyber-physical systems (CPS) are described as the fusion of the physical resources, like the machinery in a factory, with computational and cyber resources and the coordination between them [12].

CPS are identified as a key research and development area as they have a big role in the development of the I4.0, they provide the connection between manufacturing and the information and communication systems. Considering this, the most recent CPS’s generation are able to store and analyse data, are network compatible and have multiple sensors and actuators, capable of quick changes in production process and with a wide range of functions [11].

### Internet of Things

The Internet of Things (IoT) has been described as an infrastructure or an inter-relational system that allows computing devices to share data with each other over a network without the need of human interaction [13] [14].

The relevant feature in the industrial world is that “Things” such as RFIDs, sensors, actuators and Human Machine Interfaces (HMIs) can collaborate and stay connected with each other.
In a nutshell the IoT, in manufacturing, works as the cyber side of the CPS [11].

#### Internet of Services

Internet of Services (IoS) is described as web-based services, that directly connect the final consumer with the manufacturer and provide support infrastructure, enabling the companies to easily get information about the usage of its products and how they can improve the users experience, and the customers to get personalized updates, or custom updates as soon as they are available [15] [16].

#### Cloud Computing

Cloud computing enables companies to have on demand computing, storage and communications capacities, through advanced automation services provided by the company that offers the cloud services. These services allow the company to have a well-supported network of services capable of retrieving, sharing, computing and analysing big quantities of data in little time, without having a big datacentre on-site and easily managing costs [17].
It becomes easy to manage a production site or a company as real-time information and production data is accessible anytime and anywhere with an internet connection and company credentials [18].

### 1.5 Zollern’s production process

In ZC, the production process starts with the injection of the wax model, which in many cases produces more than one part, this is, a branch[1] (Figure 3.1), and this stage is done automatically or semi-automatically. Thus, the most common pieces produced are produced in a highly automated and quick process, where the branch is produced, and a robot picks it up and drops it in a tray in a treadmill or directly in a box, for further pick up, without the need for human interaction. In the case of larger pieces or less common ones, an operator assembles the cast, introduces it in the injection machine, starts the injection process and in the end disassembles the cast and retrieves the wax model.

After the wax injection an operator glues the branches (Figure 3.2), or the large pieces, to a bigger wax log, and, if needed, the structure is complemented with little bits of wax that later create driving channels, this built structure is the so-called wax tree (Figure 3.3). In the end the tree is washed, covered in a protecting wax coat and sent to the next stage.

At ZC with its 6 automatic machines and 4 semi-automatic, 1000 trees are produced per day.

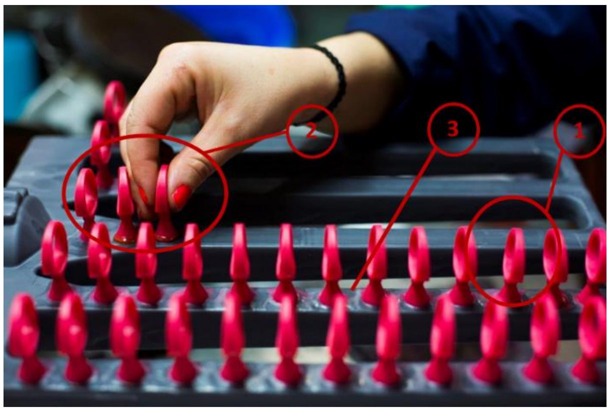


Figure 3 - Wax department - 1: Wax branch; 2: Tree being glued; 3: Wax Tree

In the ceramics department, the trees are hung in an automatic rail in platforms that can hold 5/6 trees. This rail is responsible for the movement of the trees through this entire stage.

[1] Branch – In the investment casting context, a branch is described as a set of pieces interconnected by a wax stem.

The hung trees are delivered to a robot that picks the platform and dips the trees in a ceramic bath, and then are re-hung in the same rail spot and moved around to dry. This process repeats as many times as needed until a consistent thickness is rea ched. It is also common to sprinkle or dip the tree in dry sands to improve the hardness of the ceramic structure. Both the ceramic bath and the dry sand’s composition are chosen according to the desired final result. This stage also requires a controlled environment in regard to temperature and humidity, so in ZC, sensors are spread across the department and register the values, which are sent to a Programmable Logic Controller (PLC) which controls the ventilation system and monitored by an operator.

After this stage, the trees are left resting, usually, 48 hours, and then sent to dewaxing in a Boilerclave® for about eighteen minutes at 200ºC.

Before the metal is poured, the dewaxed ceramic trees go in a rotary oven for about two and a half hours at temperatures rounding the 1000ºC. This operation “bakes” the ceramic cast, providing the resistance and the hardness required for the pouring procedure, and preheats the cast so it doesn’t break due to thermal shock.

Then, the trees leave the oven the very moment before the metal is poured into the cast.

The pouring of the metal is done manually, and requires at least four men, two holding the liquid metal container, one retrieving the trees from the oven and one opening the oven door (Figure 4.1).

At ZC the pouring stage is largely done by gravity[2], although they also have a department focused on vacuum casting[3], for more reactive metals.

In the vacuum casting department, the pouring stage is also done manually, but, as the metal quantities are smaller and the rotary oven doors can be automatically open and closed, this operation only needs two men, one operating the metal injection machine and another taking the trees out of the oven.

Next, the pieces are cooled for three hours in platforms hung in rails (Figure 4.2), and then the ceramic cast is removed with the help of a pneumatic hammer.

In the end, the pieces are cleaned, separated from the tree, final machining for deburring and quality tests are applied, and defects tended to.

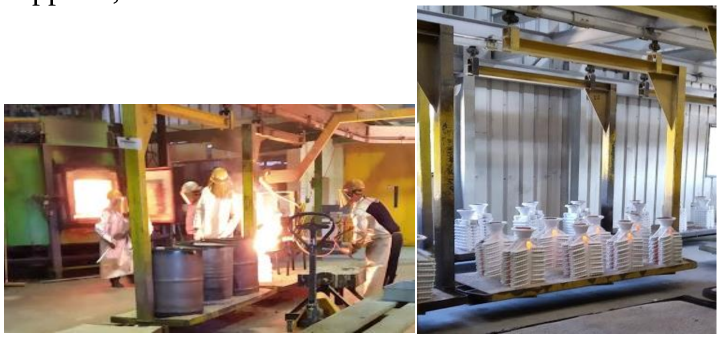


Figure 4 - Casting dapartment – 1: Gravity casting; 2: Cool room

## 1.6 Analysis of the production process

The analysis of the process described above is key to identifying the points in the process that can be improved and where the subject of this thesis would be of better service.

This analysis was made within the context of the project ZOLLERN 4.0, which focuses in automation and where the production could improve in this area.

Regarding the first stage - the wax injection, it is partially automated and partially semi-automated. This is mainly due to the bigger or complicated pieces that require an operator to assemble the mould: there usually is a sequence to assemble all the parts, making it more challenging to automate this part of the stage.

[2] Gravity casting, as one can perceive is done solely with the help of gravity to set the metal to the whole cast.

[3] Vacuum casting – Casting technique that sucks the metal into the cast over a vacuum chamber, and is, usually, used for highly reactive or oxidizable metals [62].

It should be noted that even in the automated part, very few machines are sending any data to a server.

In this stage, in the next step, the gluing of the branches or pieces to the log is done manually by trained operators. This is a very precise and sensitive work, that requires a wide range of functions and details, and since the operators glue all sizes of wax models, this task is still hard to fully automate.

In the second stage, the ceramics, although the ceramic paste is handmade, most of the process is automatic: the movement and the dipping of the trees is done without human intervention, except to set the PLC program required. In this department almost all the machines create CSV[4] files at a set time interval that are stored in a pen drive, except the robot that dips the trees, which movement and development can only be followed through the HMI present in the department.

Afterwards the handling of the ceramic casts to the shelves and from the shelves to the Boilerclave® is made over trays set in hand pushed cars. For the future, instead of hand-pushed cars it could be considered to change to automated guided vehicles (AGVs)[5] as an automated solution to move the trays.

Before the foundry stage, the tray comes out of the Boilerclave®, again helped by an operator, and one by one the trees are hung in the rotary oven, also by the hand of an operator. The foundry environment is a severe environment for humans, where the materials used and protections have to be constantly updated due to fast decaying: in the future a robot capable of enduring this environment could be considered to move the trees from the trays in and out the oven, and another robot that would be able to pour the metal into the trees, hence minimizing the risks for human health.

Both the Boilerclave® and the ovens have PLCs that register and store data, the Boilerclave® stores data to a pen drive that is rewritten every time it is full, and the ovens save a CSV file per day in the ZC’s server. The room conditions are registered by hand by an operator that consults the sensors in the department.

Lastly, the trees are taken to a larger room in safety platforms hung in rails, pushed by hand. Also, in this part of the process, the rails could be automatically powered and guided.

After this analysis and assessing the stage of ZOLLERN 4.0’s project, it was decided that the work in this thesis should focus on how the data that is now retrieved from the machines could be sent to a database, and how should it be handled in the future. This is, how to log the data that comes from the CSV files, and how could the data be stored directly from the machine in real-time.

## 1.7 Methodology

The thesis work started with the collection/aquisition of information about this thesis focus theme and getting to know INEGI’s project, including the ZC’s manufacturing facilities and its production process. This stage of the thesis was key to decide what path would be further pursued regarding the thesis objectives and the vast area that the title comprised. This stage included a few trips to the ZC’s factory for data collection, e.g. the CSV files and an inventory of all PLCs in the production area and took about six weeks.

Then the phase of development of a solution for the machine’s data collection started and went on for 14 weeks, since a few different options were considered and experimented and later adjusted.

Finally, the writing and structuration of the final document was done in a period of roughly 5 weeks.

[4] CSV file – Is a text file where the values are separated by commas, hence CSV means Comma Separated Values.

[5] AGV – Automated Guided Vehicles are robots that drive themselves along a path with the help of orientation technology, such as, wires, magnets, laser, vision, etc, and can move cargo around an industrial facility.

Through the whole process there was a weekly meeting with the thesis tutor, Professor Germano Veiga, concerning the evolution of the solutions, and a weekly Friday meeting, at INEGI regarding the evolution of the global project and its next steps.

## 1.8 Structure of the document

Next in this document, the reader will find a structured definition of the problem to be solved, or, better explained, the area in the production process of the company, that the project would like to help improve.

There will be a chapter describing the theoretical framework of the new industrial revolution and its new technologies, in the context of this thesis.

Following this chapter, there will be a thorough description of the chosen solutions, and the purpose of each one.

Finally, there will be chapter on conclusions and future considerations where the solutions are analysed, and its long-term and short-term applicability assessed. Future works in the same area and project are also considered and proposed.

# 2 Definition of the problem: Increasing the awareness of the production processes through the collection and further processing of data

As mentioned earlier, this thesis is integrated in the INEGI’s project, ZOLLERN4.0, which has the goal to improve ZC towards a more I4.0 oriented factory.

As the project is in the beginning, the first task was to study the production process in order to understand what the first steps would be.

After this study, a few primary issues arose, which included the inability to follow a piece, a branch or a tree from the beginning of the process until the end, the fact that only some machines are giving back information from the production and that this information is not treated or evaluated in any way, for example for quality control.

In the matter of identifying and following the items through the production line, the team decided that the most easy and logical item to follow would be the trees and designed a flow chart, which can be consulted in attachment A, to identify, step by step, where and when would these be identified and scanned along the process. In this case, the main issue is what technology should be used for this task, since, from the beginning until the end, these items go through many different environments, which are not easy to follow always with the same technology: apart from the funnel (which is used to make it easier to pour the metal into the cast) that is glued to the wax tree on the first stage, there is no other element that stays the same through the process, and even this funnel is susceptible to go in the ceramic bath, if the robot dives too deep. This makes the traceability of the trees during the process one of the bigger issues that the global project team is working on, so that the data that comes from the machines can be linked to each tree, and later analysed or a problem can be found, if there is a malfunction in the end.

The issue that concerns the data that comes from the machines’ PLC, is the issue that this thesis undertook and tried to solve in the context of I4.0. For this, the first step taken was to make an inventory, with the specific characteristics, of each PLC in the factory. This was not always possible due to the environment where they were being used or the inability to open the machines, but most of the PLCs characteristics, capabilities and what each records and displays is presented next.

## 2.1 Programmable Logic Controllers description

### 2.1.1 Wax injection department

In this department there are six PLCs, all in the wax injection machines, but only four of these could be reached to make an assessment, since two machines couldn’t be opened.

The following tables list and describe how/if the machines, that could be reached, are connected to a network (Table 1, Table 3, Table 5 and Table 7); what are the machines controlling and recording and the type of variable of each of these parameters and important parameters that should be saved for a better process control (Table 2, Table 4, Table 6 and Table 8).

**MPI55 – Model SASS-25-24**

Table 1 – MPI55 hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: OMRON SYSMAC CS1G (CPU42H ; CS ETN21;) | FTP/TCP/IP/Ethernet | TCP/IP/Ethernet fieldbus based | CSV file |
| HMI: BECKHOFF CP6201-0001-0035 | | | |

Table 2 – MPI55 parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| [ºC] Top wax bath | Float | Date | MM/dd/yyyy | Production Number (PN) | Integer |
| [ºC] Bottom wax bath | Float | Time end cycle | hh:mm:ss | Date and time open/close PN | DateTime |
| [m/s] Injection | Float | Receipe | Float | No. pieces to produce | Integer |
| [ºC] Injection | Float | Baseline | Float | No. producted pieces | Integer |
| [Pa] Injection | Float | Time | Float | Login operator (name e no.) | String |
| Cast position (x,y,z) | Float | Flux | Float | RFID Reader | Integer |
| [ºC] Cast | Float | Pressure | Float | Program | Integer |
| [Pa] Cast | Float | Temperature | Float | Injection time real and nominal | Float |
| | | | | Refrigerating time real and nominal | Float |
| | | | | [Pa] Injetor | Float |
| | | | | [Pa] press | Float |
| | | | | [ºC] Top and bottom wax bath | Float |
| | | | | [ºC] Tank | Float |
| | | | | [ºC] Hose and nozzle | Float |
| | | | | [ºC] fixed and moving plate | Float |
| | | | | [ºC] fixed and moving cast | Float |
| | | | | spray register | String |
| | | | | [ºC] Oil | Float |
| | | | | Alarms | String |

**MPI 100**

Table 3 – MPI 100 hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: OMRON SYSMAC CS1G (possibly) | Not connected | TCP/IP/Ethernet fieldbus based | |
| HMI: BECKHOFF CP6201-0001-0035 | | | |

Table 4 – MPI 100 parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| [ºC] Top wax bath | Float | | | Production Number (PN) | Integer |
| [ºC] Bottom wax bath | Float | | | Date and time open/close PN | DateTime |

| | | | | | |
|---|---|---|---|---|---|
| [m/s] Injection | Float | | | No. pieces to produce | Integer |
| [ºC] Injection | Float | | | No. producted pieces | Integer |
| [Pa] Injection | Float | | | Login operator (name e no.) | String |
| Cast position (x,y,z) | Float | | | RFID Reader | Integer |
| [ºC] Cast | Float | | | Program | Integer |
| [Pa] Cast | Float | | | Injection time real and nominal | Float |
| | | | | Refrigerating time real and nominal | Float |
| | | | | [Pa] Injetor | Float |
| | | | | [Pa] press | Float |
| | | | | [ºC] Top and bottom wax bath | Float |
| | | | | [ºC] Tank | Float |
| | | | | [ºC] Hose and nozzle | Float |
| | | | | [ºC] fixed and moving plate | Float |
| | | | | [ºC] fixed and moving cast | Float |
| | | | | spray register | String |
| | | | | [ºC] Oil | Float |
| | | | | Alarms | String |

**MPI 44-65 (improved and updated in the factory, both hardware and software) with a robot (KUKA Industrial Robots) attached**

Table 5 - MPI 44-65 hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7 1200 (CPU 1215C DC/DC/DC;SM 1231 TC (x2); SM 1223 DC/DC (x3);SM 1231 AI) | FTP/TCP/IP/Ethernet | TCP/IP/Ethernet fieldbus based | Not saving data |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 6 - MPI 44-65 parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Production Number (PN) | Integer | | | Production Number (PN) | Integer |
| Date and time open/close PN | DateTime | | | Date and time open/close PN | DateTime |
| No. pieces to produce | Integer | | | No. pieces to produce | Integer |
| No. defected pieces | Integer | | | No. producted pieces | Integer |
| No. pieces lacking | Integer | | | Login operator (name e no.) | String |
| No. pieces lost by robot | Integer | | | RFID Reader | Integer |
| Operating cast | Integer | | | Program | Integer |
| Login operator (name e no.) | String | | | Injection time real and nominal | Float |
| ID RFID | Integer | | | Refrigerating time real and nominal | Float |
| RFID reader | Integer | | | [Pa] Injetor | Float |

| | | | | | |
|---|---|---|---|---|---|
| Program | Integer | | | [Pa] press | Float |
| No. of cavities | Integer | | | [ºC] Top and bottom wax bath | Float |
| Injection time | Float | | | [ºC] Tank | Float |
| Refrigeration time | Float | | | [ºC] Hose and nozzle | Float |
| Alarms | String | | | [ºC] fixed and moving plate | Float |
| | | | | [ºC] fixed and moving cast | Float |
| | | | | spray register | String |
| | | | | [ºC] Oil | Float |
| | | | | Alarms | String |

**MPI 44-30 (improved and updated in the factory, both hardware and software)**

Table 7 – MPI 44-30 hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7 1200 (CPU 1214C DC/DC/DC;SM 1221 DC (x2);SM 1223 DC/DC;SM 1222 DC) | Not Connected | TCP/IP/Ethernet fieldbus based | |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 8 – MPI 44-30 parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Alarms | String | | | Production Number (PN) | Integer |
| Operating mode | String | | | Date and time open/close PN | DateTime |
| Operation | Integer | | | No. pieces to produce | Integer |
| Operation time | Float | | | No. producted pieces | Integer |
| No. pieces to produce | Integer | | | Login operator (name e no.) | String |
| No. pieces lacking | Integer | | | RFID Reader | Integer |
| Cycle time | Float | | | Program | Integer |
| Waiting time | Float | | | Injection time real and nominal | Float |
| Cycle total time | Float | | | Refrigerating time real and nominal | Float |
| Waiting total time | Float | | | [Pa] Injetor | Float |
| Injection time | Float | | | [Pa] press | Float |
| Cooling time | Float | | | [ºC] Top and bottom wax bath | Float |
| Pieces per operation | Integer | | | [ºC] Tank | Float |
| [ºC] top of tank | Float | | | [ºC] Hose and nozzle | Float |
| [ºC] tank | Float | | | [ºC] fixed and moving plate | Float |
| [ºC] Cilinder | Float | | | [ºC] fixed and moving cast | Float |
| [ºC] Hose | Float | | | spray register | String |
| [ºC] Injection nozzle | Float | | | [ºC] Oil | Float |
| [ºC] fixed plate | Float | | | Alarms | String |
| [ºC] moving plate | Float | | | | |
| [ºC] Oil | Float | | | | |

The two machines that couldn’t be open were Arburg allrounder 370 S 600kN, 290 (inj unit), which are not currently connected to any network, but are recording some data to a memory card. When the memory card is full the data is rewritten.

### 2.1.2 Ceramics department

In the ceramics department, there are three PLCs. One controls the robot that dips the trees in the ceramic bath and the air conditions in the rails in which the trees are hung, another that controls the ventilators and the movement of the rails and one that controls the Boilerclave®.

The following tables list and describe how the information is retrieved from the PLCs and how it could be done in the future (Table 9, Table 11 and Table 13), and after that, what the PLCs are controlling and recording, the type of variable of each of these parameters and important parameters that should be saved for a better process control (Table 10, Table 12 and Table 14).

**Shell-o-matic robot C**

Table 9 - Shell-o-matic robot C hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7-300: (IM174) 174-0AAI0-00AA0 (x2) | PEN Drive | TCP/IP/Ethernet fieldbus based | CSV file |
| PLC: S7 300 (CPU 315T-3PN/DP;DI4/DO8XDC24V;A18x13BIT;DI32xDC 24V (x2);DO 32xDC 24V/0,5A (x2)) | | | |
| PLC: S7-1500 (AI8xU/I/RTD/TC ST (x3);DI 32x24VDC HF (x2);DI 32x24VDC/0,5A HF) | | | |

Table 10 - Shell-o-matic robot C parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Program | Integer | Date / Time | DateTime format | In dive time | DateTime |
| Sub program | Integer | temperature 1 line X [°C] | Float | Exit dive time | DateTime |
| Transversal position nominal | Float | humidity 1 line X [%] | Float | Begin shower time | DateTime |
| Index position nominal | Float | temperature 2 line X [°C] | Float | End shower time | DateTime |
| Vertical position nominal | Float | humidity 2 line X [%] | Float | Hung back time | DateTime |
| Velocity | Float | temperature 1 line Z [°C] | Float | Viscosity | Float |
| Limit position | Float | humidity 1 line Z [%] | Float | Temperature and humidity | Float |
| Program step | Integer | temperature 2 line Z [°C] | Float | Shower tub level | Float |
| Cycles | Float | humidity 2 line Z [%] | Float | Fluidizer/shower level | Float |
| Temperature (x e z) | Float | temperature outside [°C] | Float | In rail time | DateTime |
| Humidity (x e z) | Float | humidity outside [%] | Float | Exit rail time | DateTime |
| Transversal position real | Float | temperature inflow [°C] | Float | Fluidizer pressure | Float |
| Index position real | Float | humidity inflow [%] | Float | | |

| | | | | | |
|---|---|---|---|---|---|
| Vertical position real | Float | temperature outflow [°C] | Float | | |
| Output | Float | humidity outflow [%] | Float | | |
| Line x set position real | Float | airflow inflow [m/s] | Float | | |
| Line z set position real | Float | airflow outflow [m/s] | Float | | |
| Partial program times | Float | temperature drying [°C] | Float | | |
| Max. grain | Boolean | humidity drying [%] | Float | | |
| Tub (1C,2C,3.1C) | Boolean | | | | |

These PLCs are only recording the air conditions in the rails and not the movements and information of the robot.

### Ventilation control system

Table 11 – Ventilation control system hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7-1200 (CPU 1215C DC/DC/DC;SM 1223 DC/DC (x2);SM 1221 DC (x3)) | PEN Drive | TCP/IP/Ethernet fieldbus based | CSV file |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 12 - Ventilation control system parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Alarm trigger | String | **CSV file Alarmes saves:** | | | |
| Mode | String | Date;Time;Alarm ID;Event;Description | Date time format; string; string; string; | | |
| Moviment | String | **CSV file Linha saves:** | | | |
| Weekend breaks | String | Date;Time;Line;Time_until; Event | Date time format; string; Float; string; | | |
| Date | DateTime | **CSV file Ventila saves:** | | | |
| Time | DateTime | Date;Time;Line;Event | Date time format; string; string; | | |
| Operation time | Float | **CSV file Ventiladores saves:** | | | |
| System breaks | String | Date;Time;Line;Ventilator; Event | Date time format; string; string; string; | | |
| Alarms | String | | | | |

**Boilerclave®**

Table 13 – Boilerclave hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7-1500 (DI 32x24VDC (x2);DQ 32x24VDC;AI 8xU/I/RTD/TC ST (x2)) | PEN Drive | TCP/IP/Ethernet fieldbus based | CSV file |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 14 - Boilerclave’s controlling and recording parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Program time | Float | Program | Integer | In time shelf | Date Time |
| Cut time | Float | Number of the program | Integer | Begin cycle time | Date Time |
| Active cut | String | Producing order | Integer | End cycle time | Date Time |
| Door state | String | Producing order number | Integer | Exit time shelf | Date Time |
| Program state | String | Number of trees | Integer | Program | Integer |
| Date | Date Time | Number of cycles the machine has done | Integer | Temperatures | Float |
| Time | Date Time | Execution date | Date Time | Pressures | Float |
| Program name | String | Start time | Date Time | | |
| No. | Integer | Time [s] | Date Time | | |
| Suction | Boolean | Nominal Value Autoclave [bar] | Float | | |
| Exit steam state [%] | Float | Real Value Autoclave [bar] | Float | | |
| Exit steam [Pa] | Float | Real Value Autoclave [°C] | Float | | |
| Nominal Value (autoclave) [bar] | Float | Nominal Value Steam generator [bar] | Float | | |
| Real Value (autoclave) [bar] | Float | Real Value Steam generator [bar] | Float | | |
| Real Value (autoclave) [°C] | Float | Real Value Steam generator [°C] | Float | | |
| Regulation autoclave | Float | | | | |
| Nominal Value (Steam generator) [bar] | Float | | | | |
| Real Value (Steam generator) [bar] | Float | | | | |
| Real Value (Steam generator) [°C] | Float | | | | |
| Regulation (Steam generator) | Float | | | | |
| Exit wax [°C] | Float | | | | |
| Water tank [°C] | Float | | | | |
| Filling level | Float | | | | |
| Resistances [°C] | Float | | | | |

### 2.1.3 Foundry department

In this department the PLCs' brand and model of the three rotary ovens (two in the gravity casting section and one in the vacuum casting section) couldn't be directly retrieved. It could only be acknowledged that the two ovens, in the gravity casting section, would have Siemens Simatic S7-1500.

Although the hardware couldn't be inventoried, the parameters that these PLCs control and record were listed (Table 16, Table 18 and Table 20), as well as, the important parameters to save for a better process control and the ovens' network connection (Table 15, Table 17 and Table 19).

**Rotary oven 1**

Table 15 - Rotary oven 1 hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7-1500 | FTP/TCP/IP/Ethernet | TCP/IP/Ethernet fieldbus based | CSV file |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 16 - Rotary oven 1 parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Producing order | Integer | Date and time every minute during a day | Date Time | In tree time | Date time |
| Production code (item shape and set up) | Integer | TZ 1 [°C], RZ 1 [°C], OZ 1 [°C] | Float | Position in oven | Float |
| Number of trees | Integer | TZ 2 [°C], RZ 2 [°C], OZ 2 [°C] | Float | Rotary oven movement times | Date Time |
| Ceramics weight | Float | TZ 3 [°C], RZ 3 [°C], OZ 3 [°C] | Float | 5 zones temperatures nominal | Float |
| Filling | Float | TZ 4 [°C], RZ 4 [°C], OZ 4 [°C] | Float | 5 zones temperatures real | Float |
| Ceramics height | Float | TZ 5 [°C], RZ 5 [°C], OZ 5 [°C] | Float | Exit tree time | Date Time |
| Target temperatures by oven's section (1,2,3,4,5 and total) | Float | O2 probe [%] | Float | O2 probe [%] | Float |
| Real temperatures by oven's section (1,2,3,4,5 and total) | Float | | | Gases control | Float |
| Time (1,2,3,4,5 and total) | Float | | | exhaustion gases temperature | Float |
| Gases control | Float | | | | |

TZ - Target Zone; RZ - Real Zone; OZ – Offset Zone

**Rotary oven 2**

Table 17 - Rotary oven 2 hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7-1500 | FTP/TCP/IP/Ethernet | TCP/IP/Ethernet fieldbus based | CSV file |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 18 - Rotary oven 2 parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Date and time every minute during a day | Date time format | Date and time every minute during a day | Date time format | In tree time | Date time format |
| TZ 1 [°C], RZ 1 [°C], OZ 1 [°C] | Float | TZ 1 [°C], RZ 1 [°C], OZ 1 [°C] | Float | Position in oven | Float |
| TZ 2 [°C], RZ 2 [°C], OZ 2 [°C] | Float | TZ 2 [°C], RZ 2 [°C], OZ 2 [°C] | Float | Rotary oven movement times | Date time |
| TZ 3 [°C], RZ 3 [°C], OZ 3 [°C] | Float | TZ 3 [°C], RZ 3 [°C], OZ 3 [°C] | Float | 5 zones temperatures nominal | Float |
| TZ 4 [°C], RZ 4 [°C], OZ 4 [°C] | Float | TZ 4 [°C], RZ 4 [°C], OZ 4 [°C] | Float | 5 zones temperatures real | Float |
| TZ 5 [°C], RZ 5 [°C], OZ 5 [°C] | Float | TZ 5 [°C], RZ 5 [°C], OZ 5 [°C] | Float | Exit tree time | Date time |
| O2 probe [%] | Float | O2 probe [%] | Float | O2 probe [%] | Float |
| | | | | Gases control | Float |
| | | | | exhaustion gases temperature | Float |

TZ - Target Zone; RZ - Real Zone; OZ – Offset Zone

**Rotary oven (vacuum section)**

Table 19 - Rotary oven (vacuum section) hardware and connection

| Hardware | Current connection | Desired connection | Data saving format |
|---|---|---|---|
| PLC: S7-1500 (possibly) | FTP/TCP/IP/Ethernet | TCP/IP/Ethernet fieldbus based | CSV file |
| HMI: SIEMENS SIMATIC TOUCH | | | |

Table 20 - Rotary oven (vacuum section) parameters

| What it controls | | What it saves | | What is desired it records | |
|---|---|---|---|---|---|
| Date and time every minute during a day | Date time format | Date and time every minute during a day | Date time format | In tree time | Date time format |
| TZ 1 [°C], RZ 1 [°C] | Float | TZ 1 [°C], RZ 1 [°C] | Float | Position in oven | Float |
| TZ 2 [°C], RZ 2 [°C] | Float | TZ 2 [°C], RZ 2 [°C] | Float | Rotary oven movement times | Date time |
| TZ 3 [°C], RZ 3 [°C] | Float | TZ 3 [°C], RZ 3 [°C] | Float | 5 zones temperatures nominal | Float |
| TZ TNV [°C], RZ TNV [°C] | Float | TZ TNV [°C], RZ TNV [°C] | Float | 5 zones temperatures real | Float |
| Segment indicator | Integer | O2 zone 1 [%] | Float | Exit tree time | Date time |

| | | | | | |
|---|---|---|---|---|---|
| O2 zone 1 [%] | Float | O2 zone 3 [%] | Float | O2 probe [%] | Float |
| O2 zone 3 [%] | Float | | Float | Gases control | Float |
| 2 min setpoint [%] | Float | | | exhaustion gases temperature | Float |
| Additional air [min] | Float | | | | |
| Time [min] | Float | | | | |

TZ - Target Zone; RZ - Real Zone

## 2.2 PLCs analysis

With the information gathered and presented above, it was found that not all the machines are saving data and the ones that are, don’t, usually, retrieve all the parameters they control or retain.

The first step considered by the ZOLLERN4.0’s team, was to update all the machines so they would be connected to a network and saving data from their processes to an exterior platform.

The second step was to update the information that was saved by the PLCs, in the case of the machines that already record data, the objective was to save more significant parameters, and, in the case of the machines that are not recording information, the objective was to program them to save all the significant parameters. For this, we asked the supervisor of every department to designate a list of the data that would be useful to collect from the machines, as shown in section 2.1.

The third step would be to treat and storage the data coming from the PLCs, so it can become a useful asset in quality control or process monitoring. To achieve this, a database needs to be created where every value can be related to a part of the process, an item in production, a date and time.

Later, with all the machines in the same network and communicating, it would be possible to retrieve data in real-time and create a Supervisory Control And Data Acquisition (SCADA) system.

## 2.3 Approach

Since the first two steps mentioned earlier took a very long time to finalize with regards to the time available for this thesis to be completed, due to all the administrative licenses from the mother group, ZOLLERN®, needed to implement the network connections in all the machines, and the need to gather the contacts of the managers of each machine to update/reprogram the collected data from the PLCs and wait until all the machines were reprogrammed a compromise had to be achieved.

Thus, it was agreed that the best path was to start building a bridge between the information that was already being produced and output from the PLCs and the future database to be built with all the information.

As it is, not only the writing of the data from CSV files to a database was taken in consideration, but also, the possibility to later write the values directly from the PLC to a database or from the PLCs to a cloud.

# 3 Theoretical Framework

This chapter will concern the definitions and concepts studied in the purview of this thesis, regarding the Industry 4.0 concept, summarized in Figure 5, with emphasis on the areas that were explored to produce this work.

The research will be focused on the new ways to gather and save data, and how the paradigm of the industry is changing with the velocity that information is now available instantaneously.

This chapter will also explore some important concepts needed to comprehend the developed work for this thesis, specifically device communication methods/protocols.

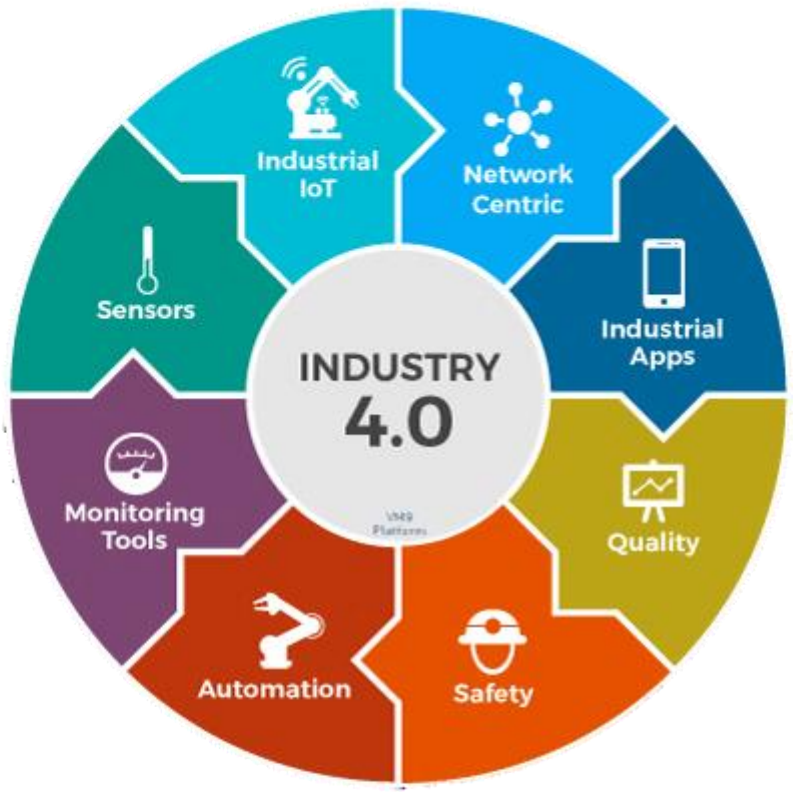


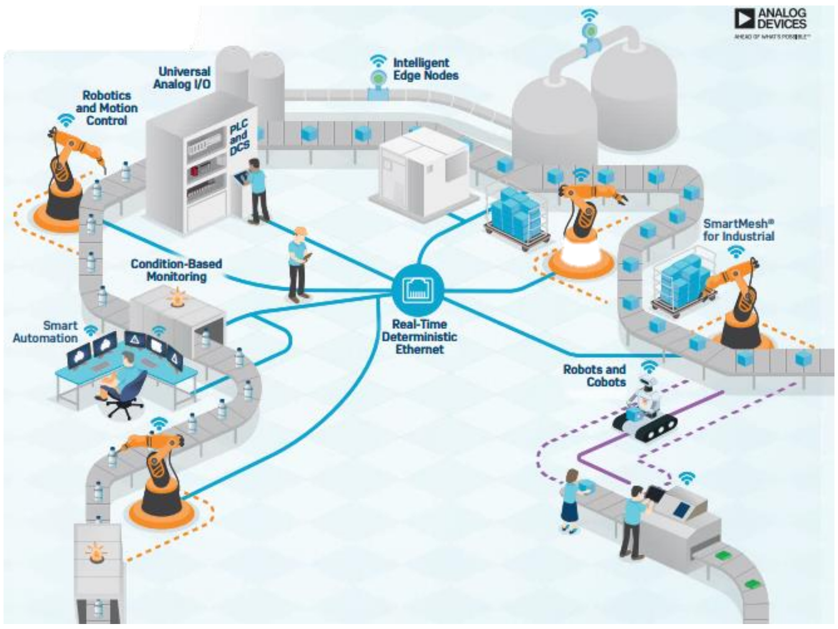
Figure 5 - Industry 4.0 in a nutshell (adapted from [19])

## 3.1 Industry 4.0 and the change of the industrial paradigm

The fourth industrial revolution, unlike any other in the past, was intentionally stimulated[6]: the companies can change their technology, more or less, at their own pace, allowing them to prepare and choose their path for the future in what concerns their

[6] This concept started being developed by a German work group, around 2011, to evolve the country’s industries, which goal was to ally the future of technology with the industrial processes, computerize the industry.

manufacturing model [20]. This transformation, although different and specific to each manufacturing can be generally described by Figure 6.


ANALOG DEVICES
Universal Analog I/O
Robotics and Motion Control
Intelligent Edge Nodes
PLC and DCS
Condition-Based Monitoring
SmartMesh® for Industrial
Smart Automation
Real-Time Deterministic Ethernet
Robots and Cobots
Figure 6 - Industry 4.0 new working model (adapted from [21])

Even though it was predicted, this technological revolution will enforce big changes, not only in the industrial sector but also in the technological world. As the industry develops and embraces the I4.0 concept, the technological world will have to keep up and grow to match the industrial needs.

As mentioned in 1.4, in Figure 6 and in Figure 7, one of the pillars of the I4.0 is the interconnectivity and interoperability between systems, machines and Man-and-machine. This is important, for example, in the decentralization of decisions, where the machines can define their next tasks, rearrange tasks within the task-force depending on the volume of work or solve a level of problems on their own. But it also means that there will have to be space for big amounts of data to travel and be analysed instantaneously, which requires systems to be faster in data processing and management and to be able to store all this information.

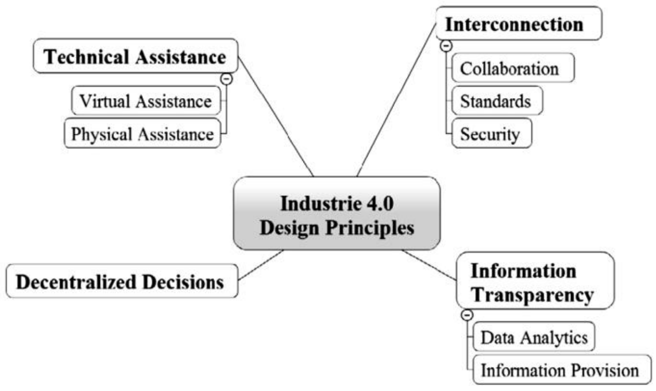


Figure 7 - Industrie 4.0 design principles (adapted from [11])

This new concept, where data is instantaneously available and used for real-time decision making, instead of existing but not being easily available, is one of the big innovations this revolution brings, as it is only with the help of immediate knowledge that machines can quickly or smoothly adapt, adjust, act on their own or even predict market fluctuations.

But this cannot be accomplished without massive computing and storage capabilities or smart data gathering. So, the way databases are structured, how they treat data and the volume of data that needs to captured is also changing.

## 3.2 Databases and Big Data - Evolution and new tendencies

### 3.2.1 Database concept and evolution

A database is a collection of data that is logically organized [22]. However, this data can be organized in many different ways, in other words, the rules that bound a database or bind its data differ with the complexity and the needs of each database.

The first database models, created around 1960, were the, so called, navigational models where to get a record (group of fields) in the database one followed other references from other records. In these models two types were created: the hierarchical and the CODASYL. In the first, the records are connected through links in a tree-like structure seen in Figure 8, the second allows a chain of events as described in Figure 9 [22].

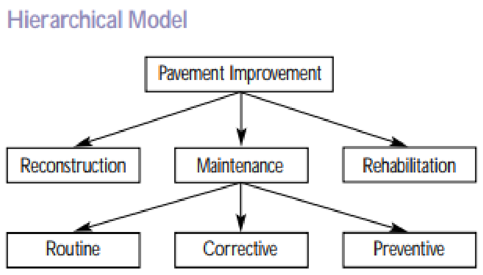


Figure 8 - Hierarchical model (adapted from [23])

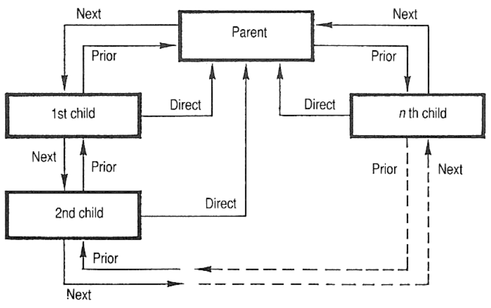
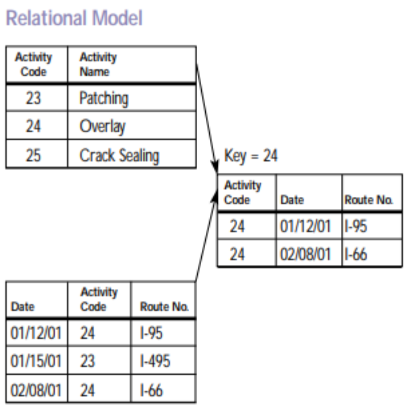


Figure 9 - CODASYL database example (adapted from [24])

The second main model, proposed by E. F. Codd in 1970 [25] still vastly used nowadays, is the relational model. In this model the data is restricted, as the name describes, by its relations. These relations are defined by primary keys, attributes (or columns), domains, foreign keys and records/tuples (or rows), also these relations are what defines a table, which is in fact the main relation that binds the data in this model [22].

This model also allows logical connections between tables whenever there are interactions between tables as acknowledged in Figure 10.



| Activity Code | Activity Name |
|---|---|
| 23 | Patching |
| 24 | Overlay |
| 25 | Crack Sealing |

| Activity Code | Date | Route No. |
|---|---|---|
| 24 | 01/12/01 | I-95 |
| 24 | 02/08/01 | I-66 |

| Date | Activity Code | Route No. |
|---|---|---|
| 01/12/01 | 24 | I-95 |
| 01/15/01 | 23 | I-495 |
| 02/08/01 | 24 | I-66 |

Figure 10 - Relational database (adapted from [23])

With this new interesting database model IBM started developing a programming sublanguage that came to be called SEQUEL (Structured English Query Language) and later, for legal reasons, SQL (Structured Query Language). This language allows the user of the database to create, change, access, delete or manage the collections, tables and entities easily [22]. Nowadays, this language is in its fourteenth normalized version (ISO/IEC 9075-14:2016).

In the 1990's, another database model started to emerge much by the influence of object-oriented programming[7] development, the object-oriented databases. These databases generate different connections between data as observed in Figure 11 [22].

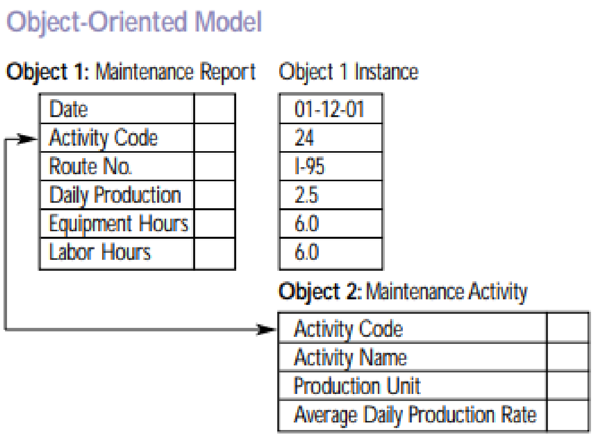


Figure 11 - Object-oriented model (adapted from [23])

Today, databases are a mixture of relational and object-oriented databases, capable of data manipulation (run procedures triggered by inputs), deduction and data analysis based on the imposed correlations and data definitions/propositions. These intelligent databases are important to obtain fast information, vital to improve productivity, decrease error, system development and maintenance [22].

Alongside this, databases are, now, developed/optimized for different types of data:

- XML Database – Allows data to be specified, stored in XML format, and later called or returned to a calling system [26].
- Semantic model –This model is usually used in software engineering, used as an abstraction, defining how the stored symbols relate to the real world.
- Content store model – Can be considered like a digital library or a storage engine, with the extra possibility that can be added, changed, deleted… Instead of just accessed.
- Event store model – Designed for the storage of real events and its corresponding data: once validated and stored the events cannot be modified.
- Time series Databases – Increasingly famous in the Industry 4.0 concept, this type of database is optimized to handle arrays of data indexed by time. This type of database allows the combination and analysis of various time series or real-time analysis, quickly getting patterns and behaviours. With the advantage of being easily deployed to a standard relational database [27].

### 3.2.2 Big Data definition and the future

In [28] the authors define Big Data as sets of data whose size is beyond the abilities of typical databases software to manage and analyse. But, the authors also say that big data is the fuel of the I4.0, in other words, this massive flux of data will produce a social network-like environment between machines and resources, creating the perfect system in a I4.0 perspective,

[7] Object-oriented programming: Is a type of programming, where the structure of the program is based on "objects" interacting with eachother, passing messages back and forth, to perform a task [60].

for, as mentioned earlier in 3.1, it is the collection and evaluation of many different types of data in due time that allows a CPS to act and make a decision in real-time.

It is also because of the constant data transfer between manufacturer and consumer, which leads to mapping behaviour and flows, that mass customization is possible. This is how the supply chain gets transparent and fully integrated [20]. A perfect example of this, is the Tesla support service, where the car receives updates via Wi-fi, and where the updates can be customized for each user.

In fact, the number of services and companies buying systems to work with big data grows day by day, and this was predicted by analysts in [29].

> *From 2005 to 2020, the digital universe will grow by a factor of 300, from 130 exabytes to 40,000 exabytes, or 40 trillion gigabytes (more than 5,200 gigabytes for every man, woman, and child in 2020). From now until 2020, the digital universe will about double every two years*

The systems capable of dealing with these amounts of data usually require cloud computing. This happens because the physical infrastructure needed to compute Big Data in real-time is too big or costly for many companies or services to acquire. Another advantage of cloud computing and storage services is the pay-as-you-go payment method, which works as follows: if the user is requiring much computing power or storage availability it will pay more, but, for example, in hours of less data traffic or work, when the user is not requiring much services, he will pay less. This creates a much easier way for companies to store and analyse their data, removing the need of a physical server and enabling the access to the data anywhere and anytime.

The main idea that the reader should retain is that the volume, velocity, variety and the value of data is rapidly increasing, causing all the infrastructure and systems that support and manage it to evolve and adapt, from the way we store it to the way we analyse and construct a database.

Nevertheless, all this data is generated somewhere, and in the industrial sector it typically comes from PLCs embedded in machines or systems.

## 3.3 PLC – A bridge between the physical world and cyber world

In third industrial revolution, when the industry started automating processes, the hardware needed to automate a process took a lot of space, there would be walls of relays (Figure 12), each one representing a function or process initiator, which when actuated would perform the associated task, turn a motor, for example [30].

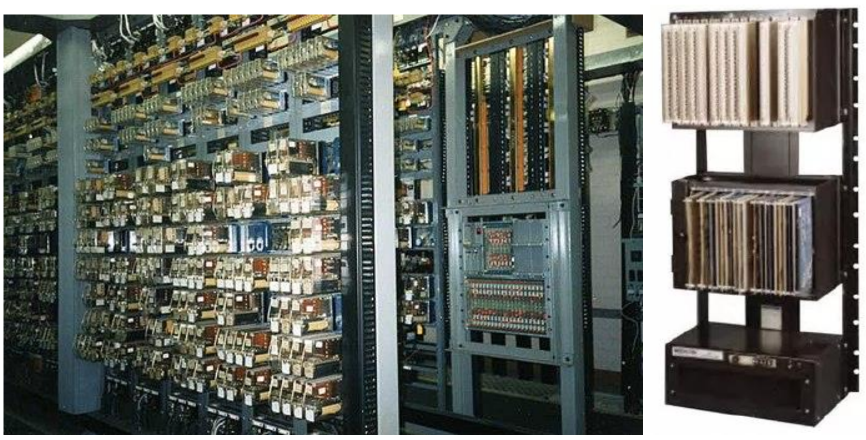


Figure 12 - Primary automated systems, Relay cabinet (adapted from [30]) (On the left) and MODICON 084 (adapted from [31]) (On the right)

As these control systems were hard to troubleshoot, complex and sometimes unreliable, and as technology evolved, in 1968, the first PLC was created. The first model was the MODICON 084 (Figure 12) [30].

Since then, companies haven’t stopped researching and creating increasingly efficient, compact and robust PLCs. They became capable of more complex programs and with a higher computational power.

Nowadays, PLC’s are able to control from small machines to entire manufacturing systems. This includes sending and retrieving information to or from processes happening in real-time and if connected to a network send this data, or receive data allowing another entity to control the processes, as happens with supervisory control and data acquisition (SCADA) systems [32].

This PLC’s ability to connect to a network and communicate with an external system is what makes this industrial computer the bridge between the physical world, sending commands to control the machine, and the cyber world, retrieving data from the sensors embedded in the machine and sending it to the pre-programmed path.

With the increasing interest in this matter, the need for the interface between the programmer and the PLC to be universal and easy to learn started growing. Today, with proper software (for instance CODESYS), basically any PLC can be programmed at a distance, as long as it is connected with the system or computer from where the program is downloaded (Figure 13), but it can also be connect to a database, a cloud or a web-browser [32].

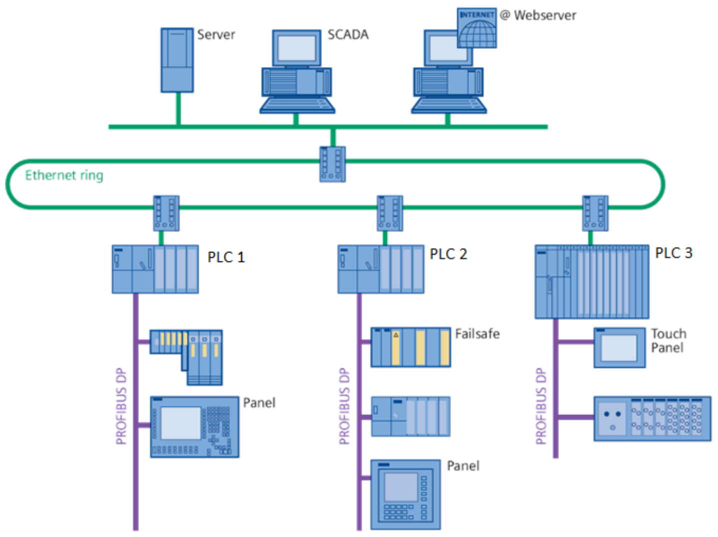


Figure 13 - PLC connection example (adapted from [33])

The first PLCs could only be programmed with Ladder logic language, whereas nowadays they can be programmed in 5 different languages: Ladder Diagram (LD), Structured Text (ST), Function Block Diagram (FBD), Instruction List (IL) and sequential function chart (SFC), thus, in order to prevent that each company had different programming, communication or overall PLC settings, the standard IEC 61131 was created [34].

### International Standard IEC 61131-3

The IEC 61131 standard regulates functional properties and characteristics, equipment requirement and tests, programming languages and its implementation, communication protocols and safety requirements. One of the most important parts of this standard is the IEC 61131-3 (the third part), which regulates semantics and syntax of all the PLC’s programmable languages enabling their interoperability, in the same program, and their portability between machines [34].

Specifically, this standard stipulates the data types, variables types (global or local) and how they should be initialized, configuration characters, configuration keywords, external data representation and program organizational units for each language [35].

## 3.4 Ethernet TCP/IP based industrial communications

A communication protocol represents a set of rules that two entities must follow to be able to interact. In this case, two or more devices, such as computers or PLCs or other factory devices, that might need to share data with the system.

In the Local Area Networks (LANs) protocols, the type communication usually used in the industrial sphere, these rules define parameters such as syntax and semantics of the message, access by multiple devices, synchronization and data flow control, failure detection and error recovery methods [36].

Beyond this characteristics, industrial networks usually also meet other criteria like being capable of supporting real-time control, being deterministic, guaranteeing high data integrity, being immune to “noise”, high reliability in heavy environments, and suitable for large installations [36]. One example of this is the Ethernet.

For a better comprehension of how these networks function, the International Standardization Organization, developed the Open System Interconnection model (OSI model), which divides a communication system into abstract layers, becoming easier understand how the information is transferred (Figure 14).

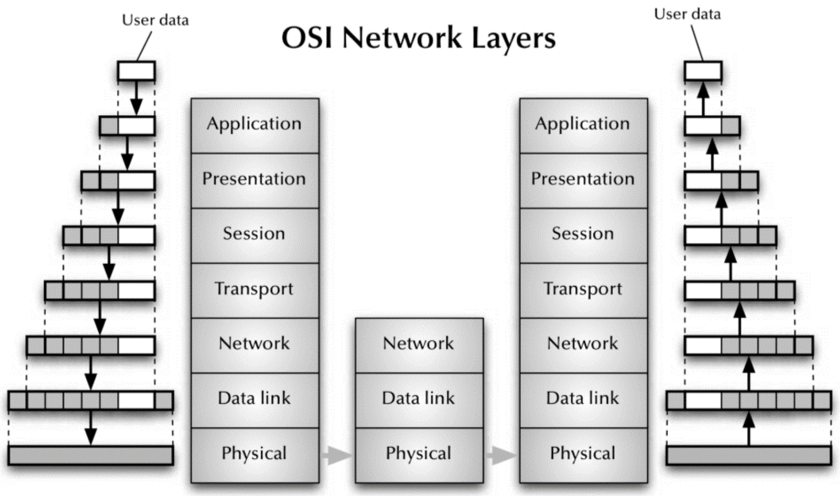


Figure 14 – OSI Network Layers (adapted from [37])

Each layer of this model is responsible for a function in the network, when transferring a message:

- Application: Data generation and end user layer
- Presentation: Data representation and encryption layer
- Session: Synch and send, establishes the connection between systems (log-in/log-off)
- Transport: Segments messages, maintains logical connections, reliable transmission of data segments between points on the network.
- Network: Routing and traffic control, data packets assembled or divided, managing connections between devices
- Data link: Reliable transmission of data frames, error detection/correction
- Physical: Transmission and reception of raw bits over physical medium (i.e. copper wire)

However, the OSI model by itself does not represent a network protocol, but a framework on which the networks are constructed. An example of a protocol widely used nowadays it’s the TCP/IP – Internet Protocol Suite, used in the internet and many computer networks like the ethernet.

**TCP/IP - Internet Protocol Suite**

This protocol follows the principles of the OSI model but joining the seven layers of the OSI model in only four, two from which its name derives [38].

Table 21 – OSI model vs. TCP/IP

| OSI Model | TCP/IP |
|---|---|
| Application | Application |
| Presentation | |
| Session | |
| Transport | Transport |
| Network | Internet |
| Data link | Network Interface |
| Physical | |

The name TCP/IP comes from its middle layer, the Transport layer implemented by Transmission Control Protocol and the Internet layer, implemented by the Internet Protocol.

The internet layer is responsible for the routing of packets from one host to another, where each packet contains the address needed for its routing [38]. However, in many cases, additional protocols are required, such as in ethernet’s case where the IP address has to be translated to a hardware address (MAC).

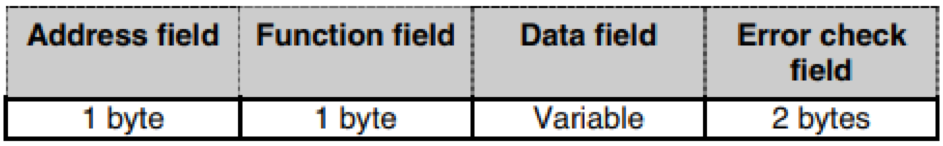
The transport layer is responsible for data integrity between sender and receiver. TCP is a very reliable connection-oriented protocol that ensures the transferring of a stream of data in byte format [38]. It can provide a delivery information on the packets transmitted to a sending point or upper-layer protocols and applications; retransmit data until a timeout state is reached or a successful message is sent; recognize duplicated messages and discard them; and buffer or control the flow of transmitted messages.

If the most common industrial network is the ethernet TCP/IP, it is often complemented with other protocols, especially at a field level or fieldbus networks. These last, are usually, transmission protocols for process control systems [38]. Examples of these network protocols are the Modbus (TCP and RTU) and Profinet. The Modbus TCP protocol is used later in the software developed for this thesis and the PROFINET protocol is the preferred fieldbus protocol used in ZC.

### 3.4.1 MODBUS TCP

Modbus began as a communication protocol mainly for serial information transmission developed by MODICON® in 1979 [39]. As it is an open protocol, manufacturers can build it into their equipment without the need to pay licenses or royalties. Modbus is usually used to transmit data from instrumentation and control devices to a main controller or data gathering system (SCADA). This is a very popular standard protocol and some estimates indicate that over 40% of industrial communication systems use Modbus. This method operates in a slave-master protocol that can have up to 247 slaves, where the master requests the information and the slave provides this information [38].

Each message frame contains four fields (Figure 15).



| Address field | Function field | Data field | Error check field |
|---|---|---|---|
| 1 byte | 1 byte | Variable | 2 bytes |

Figure 15 – Modbus message frame format (adapted from [38])

The first field, the address field, identifies the device that is sending a message or from which a message is requested. The Function field identifies the function that the device is to perform, and if performed the same value is returned. The data field contains additional information for the device to perform its task, and in the answer contains information required by the master. Lastly, the Error check field, assures the integrity of the message by calculating a numeric value and performing a cyclic redundancy check [38].

Nowadays, there are three different Modbus specifications: ASCII, RTU and TCP.

The first two are serial implementations of this protocol, and the last one created for the large implementation of the ethernet in the industrial sector, following the OSI model.

The MODBUS TCP protocol defines message presentation and application levels of the OSI model (Figure 16).

As mentioned earlier, the Modbus protocol is defined as a master and slave protocol. However, in the TCP version of this protocol this relation becomes less obvious since this network allows peer-to-peer communication. Being an Ethernet based network, these concepts are switched with client and server concepts, where the master becomes the client and the slave becomes the server. [40] This particular Modbus protocol enables the Modbus messaging protocol in an ethernet environment using the TCP/IP protocols.

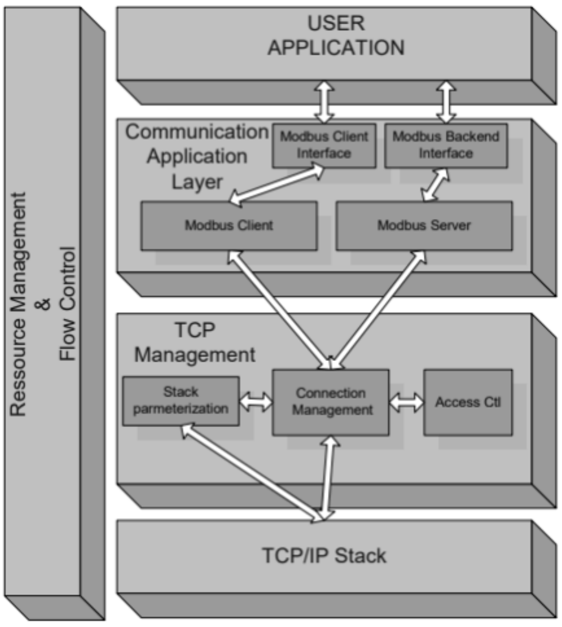


Figure 16 – Modbus TCP architecture (adapted from [41])

- Communication layer: Four areas can compose the interface between a Modbus client and server, input and output discrete (coils) and input and output registers. At this layer, it is always necessary to have a pre-mapping of these areas between the interface and the application [41].

### 3.4.2 PROFINET

This standard network protocol developed by Siemens was created to collect data from equipment and controlling it, with very tight cycle times [42]. Like Modbus TCP, this protocol works with ethernet. Any ethernet capable device can be integrated into a PROFINET network via a free port.

This protocol, much like Modbus TCP, derives from another network protocol that didn't feature the ethernet, the Profibus protocol [43], which also uses a master slave protocol.

This method divides the types of information in three different kinds known as Communication channels (Figure 17): Real-Time (RT), for information to be immediately delivered; Non-Real-Time (NRT), when the safety of the information is critical; Isochronous Real-Time (IRT), for large amounts of information that is only sent once [44].

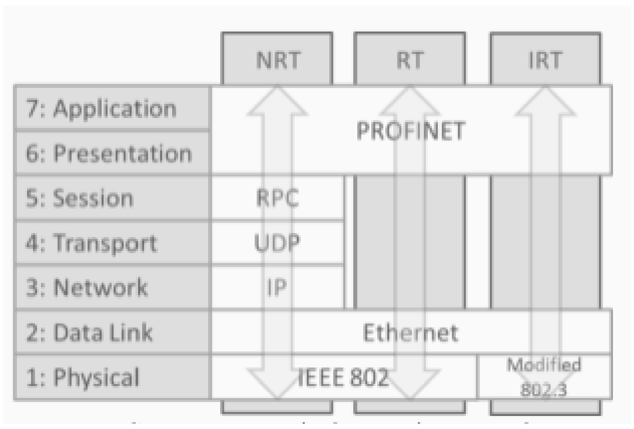


Figure 17 – Protocols used by Profinet according to OSI’s model (adapted from [44])

The RT communication channel skips the encapsulation steps in the network (transport and session layer). This ensures a faster information exchange, but they have no IP address so, they can’t be routed between LANs. On the other hand, the NRT channel uses all the OSI layers and has IP addresses, allowing supervisors to access devices from across routing boundaries or even over the internet, but the message takes longer to be transported. Lastly, the IRT channel, provides a tool that eliminates delays due to high network traffic, changing/adding traffic rules whenever it is necessary [44].

Profinet exchanges data quickly and deterministically and its data travelling speeds depend on the application, as explained above. Although the speed varies, this protocol is still faster and more deterministic than most comparable protocol, such as Modbus TCP [45].

## 3.5 OPC UA – Industrial Communication Standardization

OPC stands for Open Platform Communications and standardizes the interface between clients and servers, as well as, servers and servers (Figure 18) including how the data is transferred and accessed. This standard was created so it could work as a “middle-man” between PLC specific protocols (such Modbus or Profinet) and controls systems, such HMIs or SCADAs, converting and re-converting requests between these systems even if they are set with different protocols [46]. In the end, this standard, enhances the interoperability between systems.

The older versions of this standard separated specifications that defined how data should be accessed and exchanged, how alarms and events are exchanged and how query methods and analytical processes are applied to data [47].

Whereas the OPC version established in 2008 known as unified architecture (OPC UA) combines all the specifications of the former versions and offers [48]:

- Platform independence - can work in a PLC, a cloud-based server, traditional PC… and in multiple operating systems such as Microsoft Windows, Apple OSX, Linux…;
- Security - being firewall friendly and providing suite controls like transport, session encryption, message signing, sequenced messaging, authentication, user control and auditing;
- Extensibility - can incorporate new transport protocols, security algorithms, encoding standards, application-services or innovative technologies maintaining backwards compatibility;
- Information modelling - since it has full object-oriented capabilities, it can model or extend any data framework as complex as the structure can be.

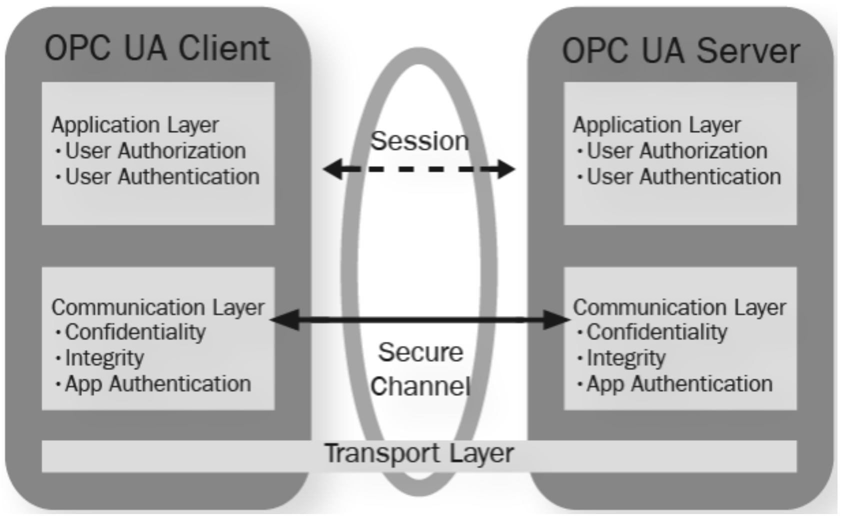


Figure 18 – OPC UA Communication Layers (Adapted from [49])

# 4 Developed Solutions

This chapter will contain a detailed description of the possible solutions considered for implementation in ZC regarding the lack of data collection and analysis mentioned in chapter 2, and also the developed solutions and tasks done in the scope of this thesis.

After acknowledging and studying the issues, a few options were considered, separating from the beginning two different topics: the untreated data that is already being stored in the company’s server in files; and the possibility of eliminating the “middle man” and storing the PLCs data directly to a database.

On the matter of the untreated data already being stored, as the data was being saved to CSV files, it was considered that the best approach was to develop an application that could read these files, collect the useful information and store it in a mock-up database, since the final database is not yet functioning, as the SQL server is also yet to be assembled.

On the matter of saving the data directly to a database three different approaches were considered. The first was to create a PLC program that connected the PLC directly to a database, in this case, the same mock-up database mentioned above. The second approach considered was to create an application that could connect the machine to a storage cloud, a mere simulation of how the writing of values to the cloud would work. The third approach connects a PLC to a storage cloud through an application.

These approaches, complete three possible different ways of extracting data from a PLC (Figure 19).

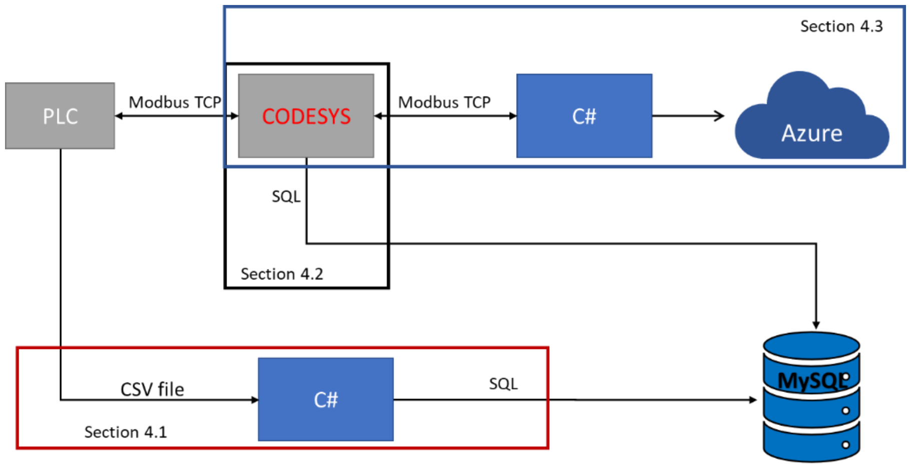


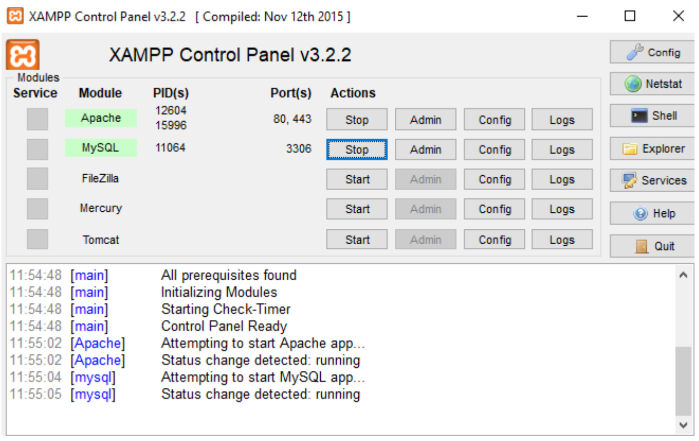
Figure 19 – Approach scheme

The reader can find all the programs described below in the following link: https://1drv.ms/f/s!Apt473Y05zqnhcN_eb4UeEdK3cDX5A.

## 4.1 CSV File To Database - C# Application

Before the development of the application, the first step was to build the mock-up database. For this, the MySQL™ services were chosen. This is an Oracle© product, that is easy to obtain, and well documented [50].

For the web server, a free open-source cross-platform was used, specifically an APACHE HTTP server product (Figure 20), XAMPP[8], which interprets scripts written in PHP or Perl languages [51]. In this case the PHP language was used, which is a general-purpose scripting language designed for web development [52].

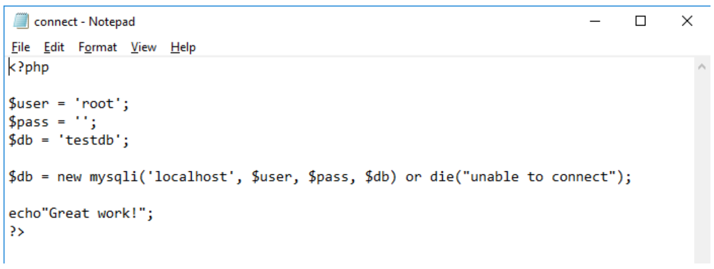

XAMPP Control Panel v3.2.2 [ Compiled: Nov 12th 2015 ]
XAMPP Control Panel v3.2.2
Config
Netstat
Shell
Explorer
Services
Help
Quit

| Modules Service | Module | PID(s) | Port(s) | Actions | | | |
|---|---|---|---|---|---|---|---|
| | Apache | 12604 15996 | 80, 443 | Stop | Admin | Config | Logs |
| | MySQL | 11064 | 3306 | Stop | Admin | Config | Logs |
| | FileZilla | | | Start | Admin | Config | Logs |
| | Mercury | | | Start | Admin | Config | Logs |
| | Tomcat | | | Start | Admin | Config | Logs |

```
11:54:48  [main]      All prerequisites found
11:54:48  [main]      Initializing Modules
11:54:48  [main]      Starting Check-Timer
11:54:48  [main]      Control Panel Ready
11:55:02  [Apache]    Attempting to start Apache app...
11:55:02  [Apache]    Status change detected: running
11:55:04  [mysql]     Attempting to start MySQL app...
11:55:05  [mysql]     Status change detected: running
```

Figure 20 - XAMPP Control Panel

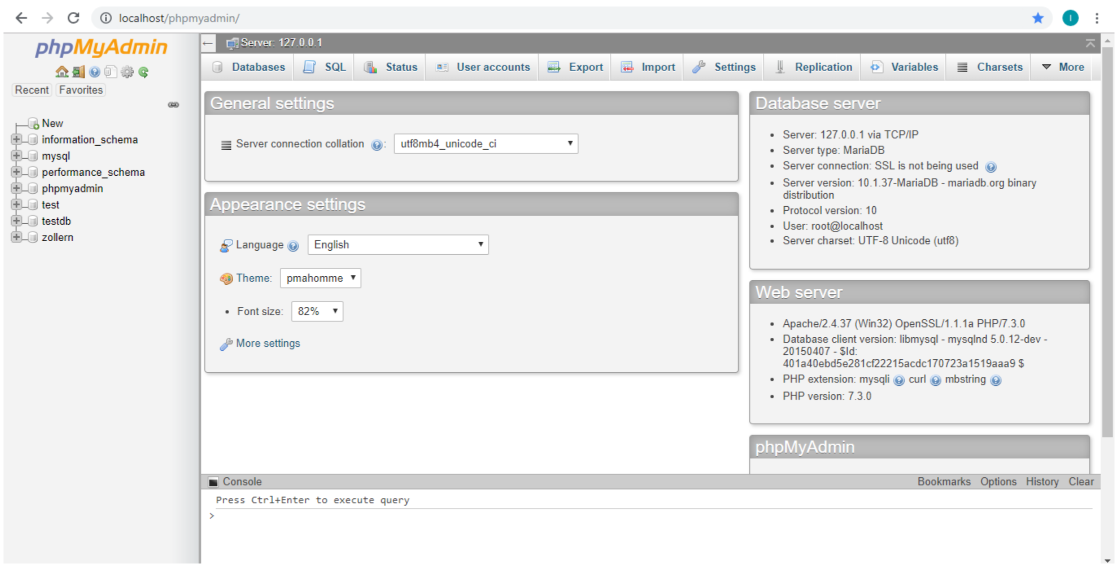
To connect to MySQL with PHP in XAMPP for the first time and create a test database, it was needed to create a PHP file with the connection parameters (Figure 21).



```php
<?php

$user = 'root';
$pass = '';
$db = 'testdb';

$db = new mysqli('localhost', $user, $pass, $db) or die("unable to connect");

echo"Great work!";
?>
```

Figure 21 – MySQL Connection configuration

After this, to access the MySQL database, the software phpMyAdmin (Figure 22) was used. This software is written in PHP and designed to administrate and manage MySQL over the web [53].



General settings

Server connection collation: utf8mb4_unicode_ci

Appearance settings

Language: English

Theme: pmahomme

Font size: 82%

More settings

Database server

- Server: 127.0.0.1 via TCP/IP
- Server type: MariaDB
- Server connection: SSL is not being used
- Server version: 10.1.37-MariaDB - mariadb.org binary distribution
- Protocol version: 10
- User: root@localhost
- Server charset: UTF-8 Unicode (utf8)

Web server

- Apache/2.4.37 (Win32) OpenSSL/1.1.1a PHP/7.3.0
- Database client version: libmysql - mysqlnd 5.0.12-dev - 20150407 - $Id: 401a40ebd5e281cf22215acdc170723a1519aaa9 $
- PHP extension: mysqli curl mbstring
- PHP version: 7.3.0

phpMyAdmin

```
Press Ctrl+Enter to execute query
>
```

Figure 22 – phpMyAdmin dashboard

[8] XAMPP: X – meaning cross-platform; A – Apache; M – MySQL or MariaDB; PP – PHP or Perl languages

But even after studying the SQL language and how to build a simple database, it was necessary to study the CSV files retrieved from the machines, in order to consider how the tables should be set up, since each was different from the other, not only in the amount of data recorded but also in data types and in syntax. For a complete picture of the mentioned differences between files it is advisable that the reader consults the attachment B where the first page of each file is exhibited.

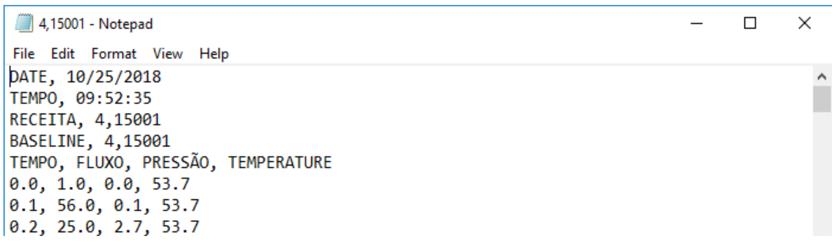
In the mock-up database setup a few decisions were made: the name of the tables would correspond to the department where the machine/system is implemented, followed by the designation of the machine/system from which the data was being retrieved, and the table structure would be as faithful to the CSV file as possible, even though this was not always possible due to the CSV’s syntax. For example, the header of the CSV files from the MPI55 is quite difficult to store in a standard table, so, during the development of the application these values were written with a different structure in order to be part of the table.

**Database setup**

As this database’s purpose was only to ensure that the data from the CSV files was being, in fact, saved in this structure, the relations between data were kept simple and only the data inside the same table was interconnected. The reader will find in attachment C a primary scheme of what this database will look like in the future.

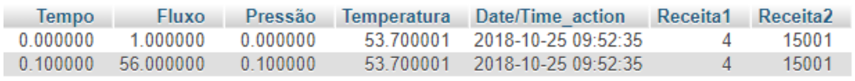
This experimental database was given the name zollern, and it features eight tables, one for each example of CSV file provided from ZC. There are files from the following machines/systems: MPI55 (one file), ceramics ventilation control system (four different files), Robot C (one file), Rotary oven 1 (two files from different working days), Rotary oven - vacuum section (one file).

The first table created was the “cera_mpi55”:  which is the table where the data structure is most different from the CSV file due to the header’s structure (Figure 23). To transfer the first four values of this file to the table, which were four rows in the beginning of the CSV file, these were written in the last columns of every row in the new table (Figure 24).



```
DATE, 10/25/2018
TEMPO, 09:52:35
RECEITA, 4,15001
BASELINE, 4,15001
TEMPO, FLUXO, PRESSÃO, TEMPERATURE
0.0, 1.0, 0.0, 53.7
0.1, 56.0, 0.1, 53.7
0.2, 25.0, 2.7, 53.7
```

Figure 23 – MPI55 CSV file header

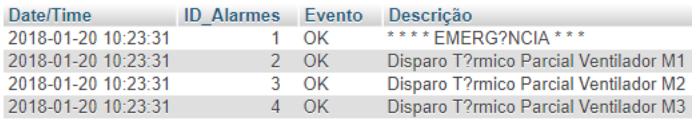


| Tempo | Fluxo | Pressão | Temperatura | Date/Time_action | Receita1 | Receita2 |
|---|---|---|---|---|---|---|
| 0.000000 | 1.000000 | 0.000000 | 53.700001 | 2018-10-25 09:52:35 | 4 | 15001 |
| 0.100000 | 56.000000 | 0.100000 | 53.700001 | 2018-10-25 09:52:35 | 4 | 15001 |

Figure 24 - cera_mpi55 table header

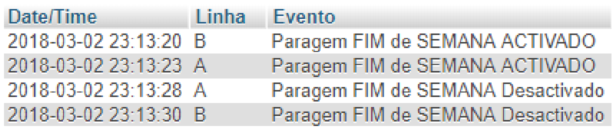
Since the values of “RECEITA” and “BASELINE” are always the same, these values were saved only once in the table.

Next, the ceramics ventilation control system tables were created. These four tables were named “ceramica_alarmes” (Figure 25), “ceramica_ctrlfds” (Figure 26), “ceramica_linha” (Figure 27), “ceramica_ventiladores” (Figure 28). In these cases, the date and time recorded in the CSV files is saved in separate columns, but, because it is usually easier to look for date and time together, the date and time in the tables were merged as a timestamp.



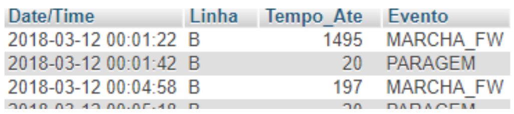
| Date/Time | ID_Alarmes | Evento | Descrição |
|---|---|---|---|
| 2018-01-20 10:23:31 | 1 | OK | * * * * EMERG?NCIA * * * |
| 2018-01-20 10:23:31 | 2 | OK | Disparo T?rmico Parcial Ventilador M1 |
| 2018-01-20 10:23:31 | 3 | OK | Disparo T?rmico Parcial Ventilador M2 |
| 2018-01-20 10:23:31 | 4 | OK | Disparo T?rmico Parcial Ventilador M3 |

Figure 25 – ceramica_alarmes table header



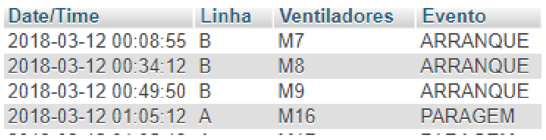
| Date/Time | Linha | Evento |
|---|---|---|
| 2018-03-02 23:13:20 | B | Paragem FIM de SEMANA ACTIVADO |
| 2018-03-02 23:13:23 | A | Paragem FIM de SEMANA ACTIVADO |
| 2018-03-02 23:13:28 | A | Paragem FIM de SEMANA Desactivado |
| 2018-03-02 23:13:30 | B | Paragem FIM de SEMANA Desactivado |

Figure 26 – ceramica_ctrlfds table header



| Date/Time | Linha | Tempo_Ate | Evento |
|---|---|---|---|
| 2018-03-12 00:01:22 | B | 1495 | MARCHA_FW |
| 2018-03-12 00:01:42 | B | 20 | PARAGEM |
| 2018-03-12 00:04:58 | B | 197 | MARCHA_FW |

Figure 27 – ceramica_linha table header



| Date/Time | Linha | Ventiladores | Evento |
|---|---|---|---|
| 2018-03-12 00:08:55 | B | M7 | ARRANQUE |
| 2018-03-12 00:34:12 | B | M8 | ARRANQUE |
| 2018-03-12 00:49:50 | B | M9 | ARRANQUE |
| 2018-03-12 01:05:12 | A | M16 | PARAGEM |

Figure 28 – ceramica_ventiladores table header

Afterwards, the “ceramica_robotc” was created were the header was kept the same as the CSV file, with the exception of a typo in the CSV file (the file mentions temperature 1 line X and temperature 1 line Z twice but with the same timestamp and different values, so it was deduced that the second pair of values refers to the measuring point 2), which was corrected in the header of the new table.

Finally, the two rotary oven tables were created: “fusao_fornogrande” and “fusao_fornopequeno”, which headers correspond to the ones in the CSV files.

After the setup of the database came the actual development of the C# application.

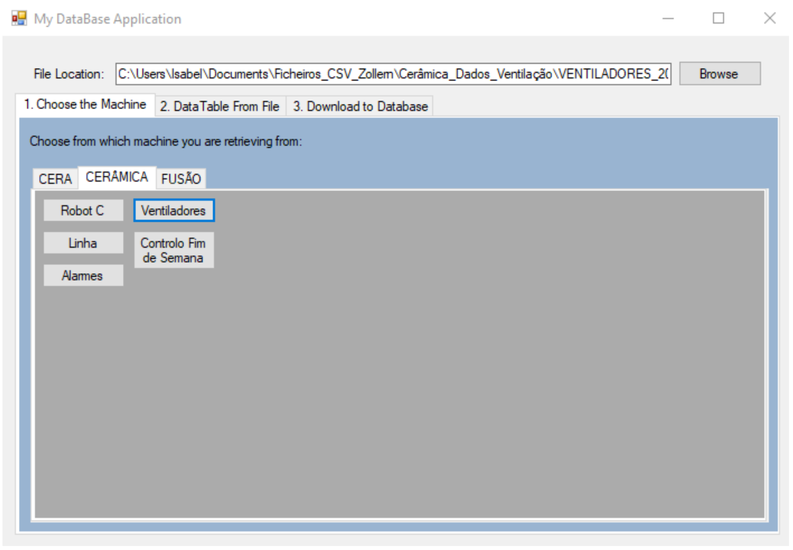
### C# Application development

Designed by Microsoft, the C# programming language is easy to learn, yet very useful. This general-purpose language is a multi-paradigm programming language, although it is intended to be object-oriented. C# simplifies programming through its Extensible Markup Language (XML) and Simple Object Access Protocol (SOAP), this allows programmers to access programming objects or methods and develop on existing code, without the need to duplicate it [54].

The selected software to develop this application was The Microsoft Visual Studio 2017 Integrated Development Environment (IDE), as IDE is a multi-web and multi Windows platform developer. The chosen format for the application was the Windows Form, which can be configured to present a user-friendly layout for anyone that usually works with Microsoft Windows system, which is ZC’s case.

This application will do three different actions: read a chosen CSV file; preview that file’s data, depending on the chosen machine, in a table alike the database’s; and write the data to the chosen table in the database.

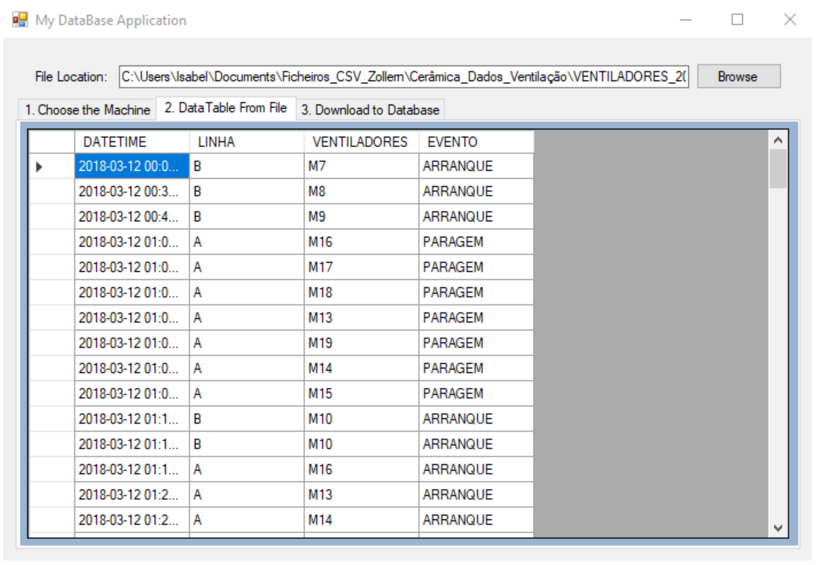
For this effect, three different areas in the Form were created, like the steps the user must take to complete the task (Figure 29, Figure 30 and Figure 31). In the first area, the user must choose the file to read and the machine/system from which the file corresponds. In the second area, the user can preview the table and confirm the data to send. In the third area, the user can choose the table corresponding to the machine and send the data.


My DataBase Application

File Location: C:\Users\Isabel\Documents\Ficheiros_CSV_Zollern\Cerâmica_Dados_Ventilação\VENTILADORES_2(

Choose from which machine you are retrieving from:

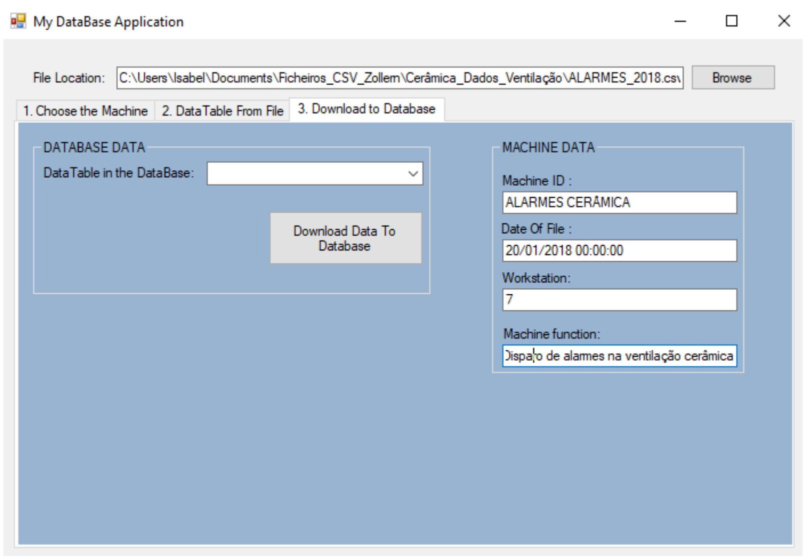
Figure 29 – First area example


My DataBase Application

File Location: C:\Users\Isabel\Documents\Ficheiros_CSV_Zollern\Cerâmica_Dados_Ventilação\VENTILADORES_2(

| DATETIME | LINHA | VENTILADORES | EVENTO |
|---|---|---|---|
| 2018-03-12 00:0... | B | M7 | ARRANQUE |
| 2018-03-12 00:3... | B | M8 | ARRANQUE |
| 2018-03-12 00:4... | B | M9 | ARRANQUE |
| 2018-03-12 01:0... | A | M16 | PARAGEM |
| 2018-03-12 01:0... | A | M17 | PARAGEM |
| 2018-03-12 01:0... | A | M18 | PARAGEM |
| 2018-03-12 01:0... | A | M13 | PARAGEM |
| 2018-03-12 01:0... | A | M19 | PARAGEM |
| 2018-03-12 01:0... | A | M14 | PARAGEM |
| 2018-03-12 01:0... | A | M15 | PARAGEM |
| 2018-03-12 01:1... | B | M10 | ARRANQUE |
| 2018-03-12 01:1... | B | M10 | ARRANQUE |
| 2018-03-12 01:1... | A | M16 | ARRANQUE |
| 2018-03-12 01:2... | A | M13 | ARRANQUE |
| 2018-03-12 01:2... | A | M14 | ARRANQUE |

Figure 30 – Second area example


My DataBase Application

File Location: C:\Users\Isabel\Documents\Ficheiros_CSV_Zollern\Cerâmica_Dados_Ventilação\ALARMES_2018.csv

Machine ID : ALARMES CERÂMICA

Date Of File : 20/01/2018 00:00:00

Workstation: 7

Machine function: Disparo de alarmes na ventilação cerâmica

Figure 31 – Third area example

In the first area, the buttons that correspond to a machine/system, have different source codes behind them, because each must read the respective machine's file and build the table according to different syntaxes or file structures. The structure of the CSV file that is extracted from each machine/system is constant. Thus for each file structure, a Class was created where the data is sorted according to the respective file and, if necessary, converted to fit the database's data formats/types as it happens often when sorting date and time values (Figure 32).

These classes are helpful since they make it easier to change the reading and assembly process of the data of a specific machine's file if necessary. One must keep in mind that most of these files will, in a short period, be altered, in terms of what they are saving, to include data which at present time is not being registered to the CSV file, so it was important to maintain an easy to modify program structure.

```
public void BindDataCSV_cerampi55(string FileName)
        {
            DataTable dt = new DataTable();
            string[] lines = System.IO.File.ReadAllLines(FileName);
            List<string> lines3 = new List<string>();

            int y = 0, m = 0, d = 0, sec, min, hour;
            //Leitura primeira linha data
            string linhaData = lines[0];
            string[] columns = linhaData.Split(',');
            y = int.Parse(columns[1].Split('/')[2]);
            d = int.Parse(columns[1].Split('/')[1]);
            m = int.Parse(columns[1].Split('/')[0]);
            //Leitura
            string linhaTempo = lines[1];
            string[] columnsl = linhaTempo.Split(',');
            sec = int.Parse(columnsl[1].Split(':')[2]);
            min = int.Parse(columnsl[1].Split(':')[1]);
            hour = int.Parse(columnsl[1].Split(':')[0]);
            this.date = new DateTime(y, m, d, hour, min, sec);
            string createddate = Convert.ToDateTime(date).ToString("yyyy-MM-dd hh:mm:ss");
            createddate.ToString();
```

Figure 32 - Date and time format converting example

Even though each machine has a specific data assembly Class, each machine/system had characteristics which change in value but not in structure, for example, name or function. These two characteristics or objects are common to every machine and are always described with a string of characters. This led to the creation of the Class "Maquina" (Figure 33), which features a few machine characteristics. This class doesn't have many objects since these machines differ a lot in function and in the parameters, they need to record. However, it is possible that with the development of the ZOLLERN4.0 project and the attempt of uniformization of the CSV files structure, that this Class can have more objects or even that the data assembly and management Classes are reduced to one, also Classes like "Cacho", that are for now only representative of a possible evolution, may start to be useful if these start to be traced.

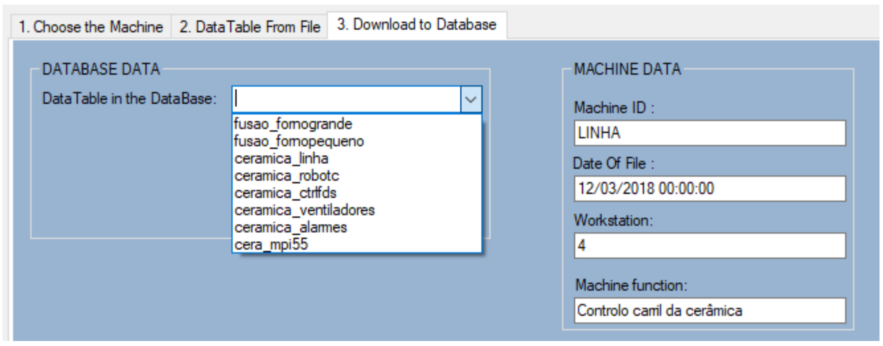
```
internal class Maquina
    {
        public DateTime datetime_maq;
        public int workstation;
        public string id_maquina;
        public string function;
    }
```

Figure 33 – Class "Maquina" program code

Summarizing, when the user presses a button in the first area, after choosing the file, the data is sorted according to the type of file and appears in the second area in table format as it will appear in the database table.

Afterwards, in the third area, after choosing the corresponding table from the list (Figure 34), when the user presses "Download Data To Database" the process of sending the data to the MySQL database begins. It should be noted that, to avoid errors when sending the data, the information about the CSV file and machine/system are displayed in a section to the right of the table list (Figure 34).

Initially, the application will attempt to connect to the database, showing a message so the user acknowledges the phase of the process, and will show another message when the connection succeeds or an error message in case it can’t. After this step, it will convert the table from the second area in a list so it can be embedded in a SQL query format (Figure 35) and sent as a command to the MySQL database.



DataTable in the DataBase:

fusao_fornogrande
fusao_fornopequeno
ceramica_linha
ceramica_robotc
ceramica_ctrlfds
ceramica_ventiladores
ceramica_alarmes
cera_mpi55

Machine ID : LINHA
Date Of File : 12/03/2018 00:00:00
Workstation: 4
Machine function: Controlo carril da cerâmica

Figure 34 – Third area table list and machine/system information

```
public void ViewToDB(MySqlConnection connection, string mySqlTable, List<string[]> Data)
        {
            var queryBuilder = new StringBuilder("INSERT INTO " + mySqlTable + " VALUES");
            var stringFormatterBuilder = new StringBuilder("(");
            for (int i = 0; i < Data[0].Length - 1; i++)
            {
                // All the items except the last one, With comma
                stringFormatterBuilder.Append("'{")
                    .Append(i)
                    .Append("}',");
            }
            if (Data[0].Length >= 1)
            {
                // Last item, without comma
                                stringFormatterBuilder.Append("'{")
                    .Append(Data[0].Length - 1)
                    .Append("}'");
            }
            stringFormatterBuilder.Append(")");
            var stringFormatter = stringFormatterBuilder.ToString();
            var rows = new List<string>();

            for (int i = 0; i < Data.Count; i++)
                rows.Add(string.Format(stringFormatter, Data[i].ToArray()));
            queryBuilder.Append(string.Join(",", rows));
            queryBuilder.Append(";");
            var query = queryBuilder.ToString();
            using (var command = new MySqlCommand(query, connection))
            {
                command.CommandType = CommandType.Text;
                command.ExecuteNonQuery();
            }
        }
```

Figure 35 – Query builder instance

When the process is complete, the program sends a message informing the user that the download is complete or an error message if something went wrong. Then, the whole process can be repeated for another CSV file or the application closed.

## 4.2 From the PLC To Database - CODESYS program

With this program the goal is to show how the direct communication between the PLC of a machine or system and a database would work. Therefore, eliminating the need to store CSV files.

The platform where the program is developed is different, however, the database and its connection are the same, so the web server (XAMPP) and the platform that manages the database (phpMyAdmin) can be retained.

This program was developed in CODESYS, since this controller development environment system develops programs according to the standard IEC 61131-3, hence working in any PLC. An alternate software, TIA portal® was considered, a software specialized to control SIEMENS PLCs, but, although most of the ZC’s PLCs are SIEMENS, not all of them are, so the final choice for the program development was the CODESYS.

Since there wasn’t a real PLC or machine where the program could be tested or implemented in, it was decided that a simulation of a machine would be made with the help of a softPLC[9] working with CODESYS.

The machine chosen to simulate was the MPI55. The application simulates the same values as the ones saved by the CSV files (Figure 36).

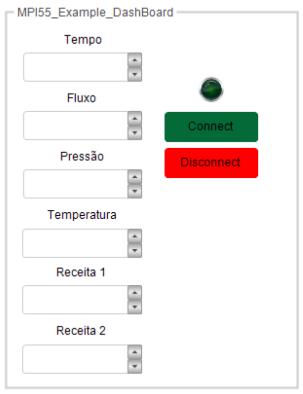
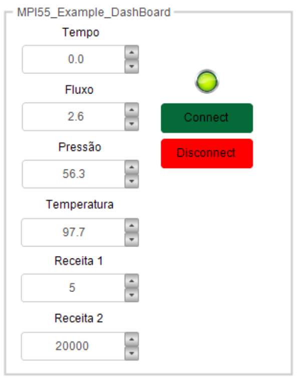


Figure 36 - MPI55 CODESYS dashboard

With the web server connected to the database and the softPLC running, the first step is to initiate the connection between the PLC and the database pushing the green button “Connect” (Figure 36), when connected the simulated green LED is turned on and the program will send the registered values every second (Figure 37).



Tempo: 0.0
Fluxo: 2.6
Pressão: 56.3
Temperatura: 97.7
Receita 1: 5
Receita 2: 20000

Figure 37 – Application running and connected

The application is started with preset values, as these are standard values for this machine, but to simulate this machine functioning, one must increase the value “Tempo” and increase or decrease the other values depending on the phase of the injection pretended to simulate. To stop the dispatch of values to the database the user must only press “Disconnect” and the communication stops.

This PLC program is written in two different languages, according to IEC 61131-3. It has two functions written in Structured Text (ST) language and the main program written in Function Block Diagram (FBD) language.

The two functions are used to convert the application variables to the SQL query that is sent to the database.

In the main program, each block corresponds to a part of the data sending process, one block containing all the parameters and functions needed to connect to the database (Figure 38.a), one block that assembles the information and sends it when commanded (Figure 38.b), one that breaks the connection when commanded (Figure 38.c), and two blocks that are actuated when the PLC is connected to the database and initiate the timer that commands the data

[9] SoftPLC: This technology is basically a software that emulates the function of a physical PLC. The CODESYS software packages already provides a softPLC of its brand to allow programmers to test their programs.

assembly block every second (Figure 39). To make the application more realistic, there is a block that initiates the “Tempo” variable when the system is connected and stops when the system is disconnected, as to simulate an injection cycle, and a function block that formats the “Tempo” variable to float instead of the CODESYS time format, so the database recognizes it (Figure 39).

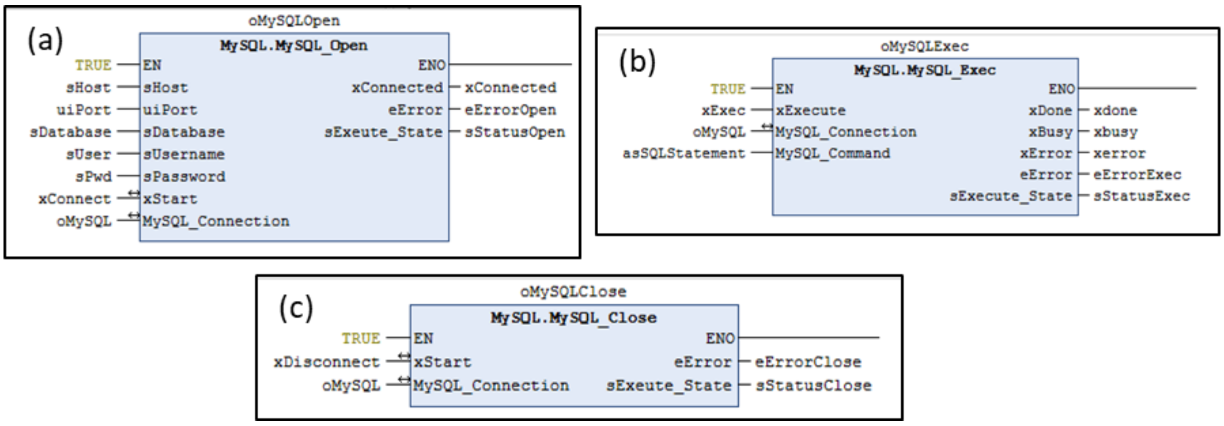


Figure 38 – (a) Opening connection Block; (b) Executing query Block; (c) Closing connection Block

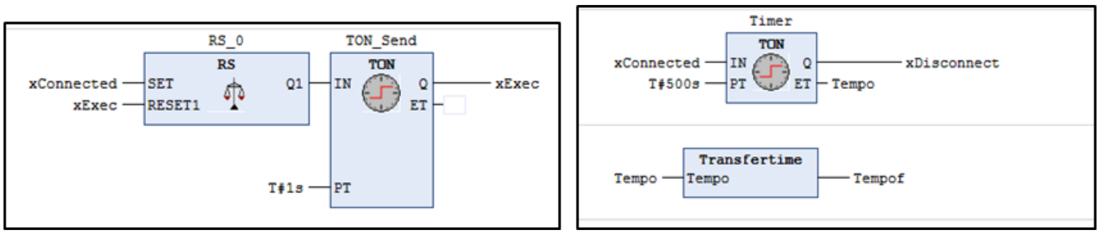


Figure 39 – Initializing timer Blocks (On the left); “Tempo” variable automated set up (On the right)

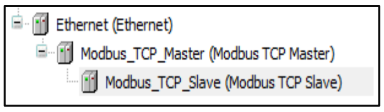
However, the Function Blocks regarding the connection and command execution to the MySQL database, are not available in the standard CODESYS environment, so, it was necessary to download a specific MySQL library containing these functions.

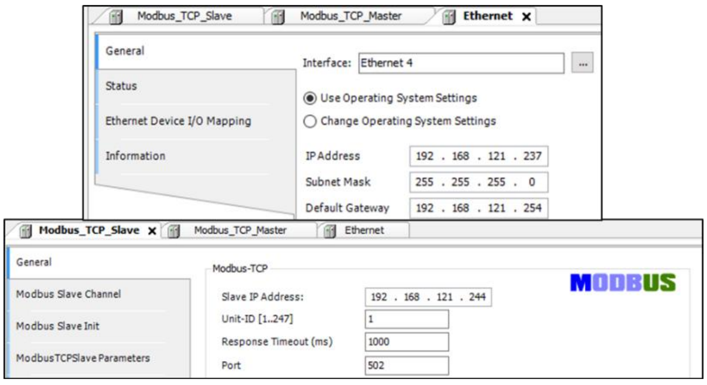
The difference between this process and the C# CSV reader application is that the data is updated to the database table in real-time, which is visible in the “Date/Time_action” column where, in this case, the value is the real timestamp when the data was sent and not the date and time when the CSV file was created. With this information, if this was always the case, one could even argue if the “Tempo” column would still be needed.

### Connection between a real PLC and the CODESYS

The connection between a PLC and the CODESYS program would be done through a Modbus TCP connection, where the PLC would be the server and the CODESYS program the client (Figure 40).

For this connection to work correctly, the network connection parameters of the PLC and the computer on the other end, must be configured in the program (Figure 41). This includes IP addresses, ports, the transferred variables and response times.


Ethernet (Ethernet)
Modbus_TCP_Master (Modbus TCP Master)
Modbus_TCP_Slave (Modbus TCP Slave)
Figure 40 – CODESYS Master Slave configuration



Modbus_TCP_Slave | Modbus_TCP_Master | Ethernet

General
Status
Ethernet Device I/O Mapping
Information

Interface: Ethernet 4

[✓ Use Operating System Settings]
Change Operating System Settings

IP Address: 192 . 168 . 121 . 237
Subnet Mask: 255 . 255 . 255 . 0
Default Gateway: 192 . 168 . 121 . 254

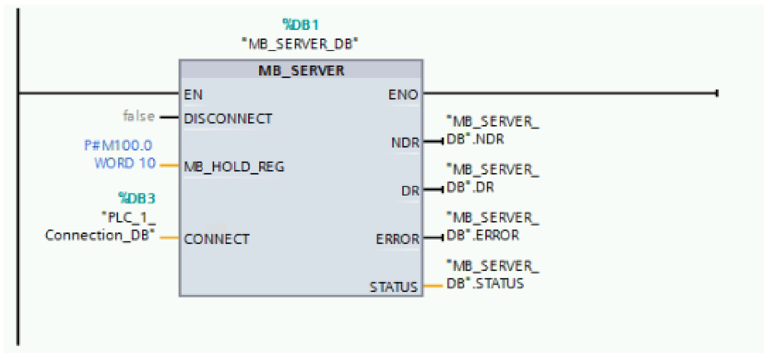
Modbus_TCP_Slave | Modbus_TCP_Master | Ethernet

General
Modbus Slave Channel
Modbus Slave Init
ModbusTCPSlave Parameters

Modbus-TCP
Slave IP Address: 192 . 168 . 121 . 244
Unit-ID [1..247]: 1
Response Timeout (ms): 1000
Port: 502

Figure 41 – Network connection parameters configuration: computer’s parameters (top); PLC’s parameters (bottom)

On the PLC’s side, this must also be programmed and configured depending on the PLC’s brand and model. But, for example, with the TIA Portal, for a S7 PLC, this is a very simple task, where only one FBD/LD is needed to program this connection (Figure 42).



```
%DB1
"MB_SERVER_DB"
MB_SERVER
EN                                  ENO
false — DISCONNECT
                                    NDR — "MB_SERVER_DB".NDR
P#M100.0
WORD 10 — MB_HOLD_REG
                                    DR — "MB_SERVER_DB".DR
%DB3
"PLC_1_Connection_DB" — CONNECT
                                    ERROR — "MB_SERVER_DB".ERROR
                                    STATUS — "MB_SERVER_DB".STATUS
```

Figure 42 – Example of a Modbus server configuration in a S7 PLC

## 4.3 Connecting To The Cloud

To follow the evolutionary steps of I4.0, it was a considered a logical step in this thesis development to also simulate the behaviour of a machine saving its data to a cloud database.

So, in this section were developed two C# applications and a CODESYS program to show how this can be achieved. The first application served as an intermediate exercise to study how the connection to the cloud would work.

### 4.3.1 From the machine To the CLOUD - C# Application to Azure

A C# application embedded with a MPI55 simulation was developed to connect to a Microsoft’s Azure cloud storage account, simulating a cloud database.

In this case the connection to the database is direct, like connecting to an email account, so the web server is not needed in this process.
For this a Windows Form was created to simulate the mentioned machine (Figure 43).

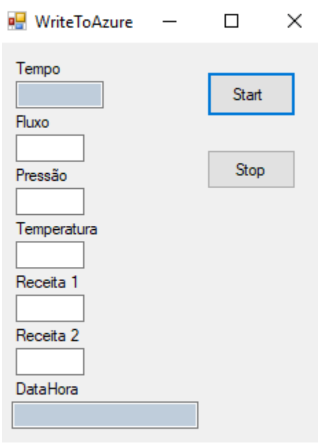


Figure 43 – MPI55: C# to Azure simulation

In this case, the value “Tempo” will, also, run as a stopwatch at the press the “Start” button, stopping and resetting at the press of “Stop” button, hence simulating a cycle of wax injection.

To recreate the MPI55 data collection, a Class called “MpiEntity” (Figure 44) was created. In this class, all the parameters that the table must save are defined and set up, meaning, the parameters that each column will contain.

```
public class MpiEntity : TableEntity
        {
            public MpiEntity(string Dapartment, int num, DateTime DataHora)
            {
                string machinest = num.ToString();
                this.PartitionKey = Dapartment;
                this.RowKey = machinest;
                this.Timestamp = DataHora;
            }
            public MpiEntity() { }
            public string Tempo { get; set; }
            public string Fluxo { get; set; }
            public string Pressao { get; set; }
            public string Temperatura { get; set; }
            public string Receita1 { get; set; }
            public string Receita2 { get; set; }

        }
```

Figure 44 – MpiEntity Class

Regarding the Class elements, the “PartitionKey Department” represents the name of the machine that is saving data to the table, in this case only the MPI55, the “RowKey machinest” represents the index of the row in the table and the “Timestamp DataHora” the date and time of the upload to the database.

The following objects correspond to the values that change during the process, the same showing in Figure 43, except the lowest value.

When the application is running, before pressing the “Start” button, the user can insert values in the blank spaces to simulate MPI55 values, however, the upper and lowest values are automatic.

Pressing the “Start” button will connect the application with the Microsoft Azure Cloud Storage Account setup in the program, and once connected the values of “Tempo” and “DataHora” will start running (Figure 45) and the application saving to the designated table. This process is done by executing a table query, which creates a new row every second with the values in the application and according to the “MpiEntity” Class (Figure 46).

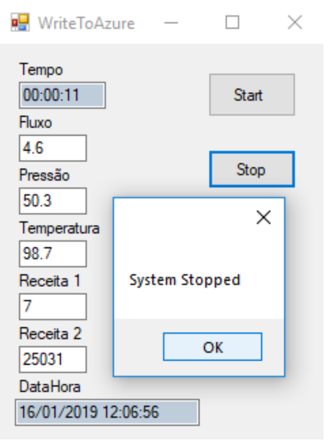
```
private void btnSend_Click(object sender, EventArgs e)
        {
            int j = 0;
            // Retrieve the storage account from the connection string.
            CloudStorageAccount storageAccount = CloudStorageAccount.Parse("UseDevelopmentStorage=true;");
            // Create the table client.
            CloudTableClient tableClient = storageAccount.CreateCloudTableClient();
            // Retrieve a reference to the table.
            CloudTable table = tableClient.GetTableReference("Zollern");
            // Create the table if it doesn't exist.
            table.CreateIfNotExists();
            // Começar Timer
            stopwatch1.Start();
            aTimer.Start();
            txtDateTime.Text = DateTime.Now.ToString();
            txttime.Text = stopwatch1.Elapsed.ToString("hh\\:mm\\:ss");
```

Figure 45 – Microsoft Azure Cloud Storage account connection, stopwatch and timestamp initiation

```
          TableQuery<MpiEntity> query = new TableQuery<MpiEntity>();
            foreach (MpiEntity entity in table.ExecuteQuery(query))
            {
                j++;
            }
            // Create a new customer entity.
            MpiEntity mpi1 = new MpiEntity("MPI55", j+1, DateTime.Now);
            mpi1.Tempo = txttime.Text;
            mpi1.Fluxo = txtflux.Text;
            mpi1.Pressao = txtpressure.Text;
            mpi1.Temperatura = txtheat.Text;
            mpi1.Receita1 = txtrec1.Text;
            mpi1.Receita2 = txtrec2.Text;
            // Create the TableOperation object that inserts the customer entity.
            TableOperation insertOperation = TableOperation.InsertOrReplace(mpi1);
            // Execute the insert operation.
            table.Execute(insertOperation);
```

Figure 46 – Table query and new table entity creation

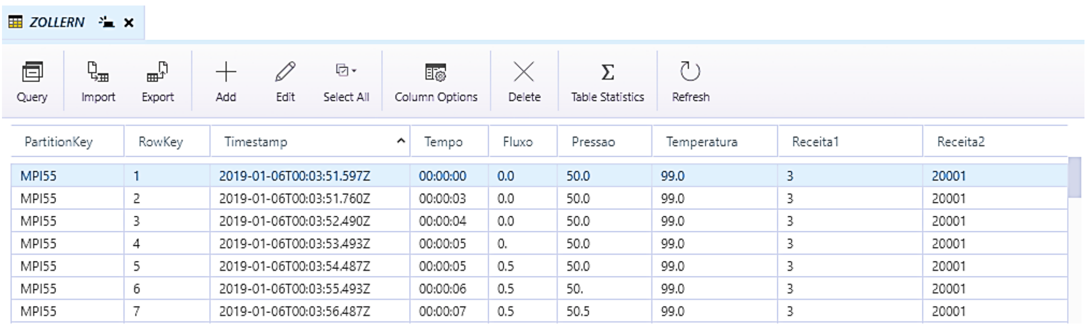
When the button “Stop” is pressed, the timestamp and the timer will stop, as mentioned earlier, and the program will stop sending data to the table and send a message informing that the system has stopped (Figure 47).



Tempo
00:00:11
Fluxo
4.6
Pressão
50.3
Temperatura
98.7
Receita 1
7
Receita 2
25031
DataHora
16/01/2019 12:06:56
System Stopped

Figure 47 – Application stopped

The table produced by this application will have nine columns, the first three are data identifiers, this means, columns that help sorting or finding data, and the following six columns correspond to the process data to be saved (Figure 48).



| PartitionKey | RowKey | Timestamp | Tempo | Fluxo | Pressao | Temperatura | Receita1 | Receita2 |
|---|---|---|---|---|---|---|---|---|
| MPI55 | 1 | 2019-01-06T00:03:51.597Z | 00:00:00 | 0.0 | 50.0 | 99.0 | 3 | 20001 |
| MPI55 | 2 | 2019-01-06T00:03:51.760Z | 00:00:03 | 0.0 | 50.0 | 99.0 | 3 | 20001 |
| MPI55 | 3 | 2019-01-06T00:03:52.490Z | 00:00:04 | 0.0 | 50.0 | 99.0 | 3 | 20001 |
| MPI55 | 4 | 2019-01-06T00:03:53.493Z | 00:00:05 | 0. | 50.0 | 99.0 | 3 | 20001 |
| MPI55 | 5 | 2019-01-06T00:03:54.487Z | 00:00:05 | 0.5 | 50.0 | 99.0 | 3 | 20001 |
| MPI55 | 6 | 2019-01-06T00:03:55.493Z | 00:00:06 | 0.5 | 50. | 99.0 | 3 | 20001 |
| MPI55 | 7 | 2019-01-06T00:03:56.487Z | 00:00:07 | 0.5 | 50.5 | 99.0 | 3 | 20001 |

Figure 48 – Microsoft Azure Cloud Storage Account Table

This application could only be tested in the Microsoft Azure Storage Emulator due to problems with the student’s Microsoft Azure account. However, this doesn’t invalidate the application, since the only necessary change to its code would be to change the connection string that appears in Figure 45, and in the “App.config” code (Figure 49).

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
    </startup>
    <appSettings>
      <add key="StorageConnectionString" value="UseDevelopmentStorage=true;" />
    </appSettings>
```

Figure 49 – App.config code and connection string

This is the path another user would have to follow to use this application: he would have to substitute the connection string to its storage account connection string in the two points mentioned above.

### 4.3.2 From the PLC to the Cloud - C# Application

For this exercise, two different platforms were used, CODESYS for establishing the PLC variables that would go on the cloud storage table, and Microsoft Visual Studio 2017 IDE for C# language to read or write values to the PLC and send the read values to the cloud’s table.

As it is, a simple interface (Figure 50) was created to change the PLC variables, where the “Start/Stop” button initiates the time variable. After this, follows the variables mapping, this establishes which variables are sent to the C# application (Figure 51), in this case, are the same one shown in Figure 50.

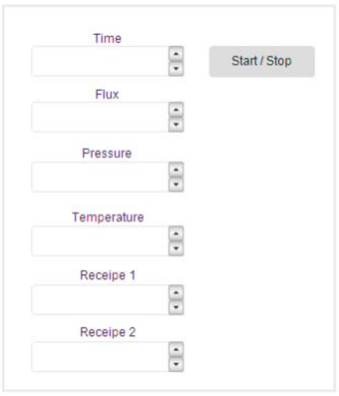
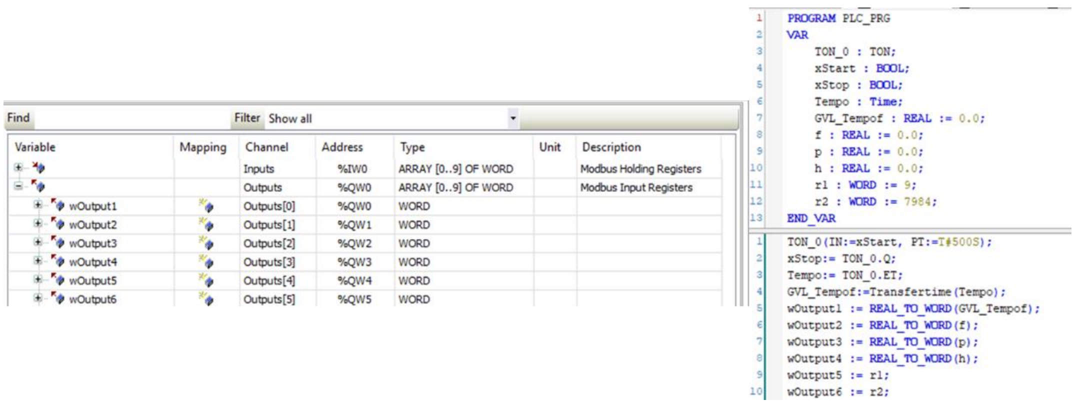


Figure 50 – PLC to Cloud application HMI



Find | Filter Show all

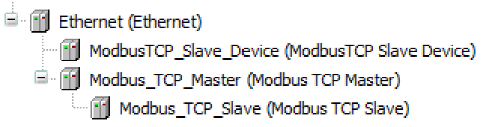
| Variable | Mapping | Channel | Address | Type | Unit | Description |
|---|---|---|---|---|---|---|
| | | Inputs | %IW0 | ARRAY [0..9] OF WORD | | Modbus Holding Registers |
| | | Outputs | %QW0 | ARRAY [0..9] OF WORD | | Modbus Input Registers |
| wOutput1 | | Outputs[0] | %QW0 | WORD | | |
| wOutput2 | | Outputs[1] | %QW1 | WORD | | |
| wOutput3 | | Outputs[2] | %QW2 | WORD | | |
| wOutput4 | | Outputs[3] | %QW3 | WORD | | |
| wOutput5 | | Outputs[4] | %QW4 | WORD | | |
| wOutput6 | | Outputs[5] | %QW5 | WORD | | |

```
PROGRAM PLC_PRG
VAR
    TON_0 : TON;
    xStart : BOOL;
    xStop : BOOL;
    Tempo : Time;
    GVL_Tempof : REAL := 0.0;
    f : REAL := 0.0;
    p : REAL := 0.0;
    h : REAL := 0.0;
    r1 : WORD := 9;
    r2 : WORD := 7984;
END_VAR
```

```
TON_0(IN:=xStart, PT:=T#500S);
xStop:= TON_0.Q;
Tempo:= TON_0.ET;
GVL_Tempof:=Transfertime(Tempo);
wOutput1 := REAL_TO_WORD(GVL_Tempof);
wOutput2 := REAL_TO_WORD(f);
wOutput3 := REAL_TO_WORD(p);
wOutput4 := REAL_TO_WORD(h);
wOutput5 := r1;
wOutput6 := r2;
```

Figure 51 – Mapping output variables (On the left); Corresponding PLC variables (On the right)

The connection between a real PLC and the CODESYS program would be a Modbus TCP connection, similar to the described in 4.2, where the PLC would be the server and the CODESYS the client. However, the connection used between the CODESYS and the C# application is, also, a Modbus TCP connection, but this time CODESYS is a Modbus slave. This means that CODESYS needs to be configured as master and slave (Figure 52) to be able to communicate with both sides.



```
Ethernet (Ethernet)
    ModbusTCP_Slave_Device (ModbusTCP Slave Device)
    Modbus_TCP_Master (Modbus TCP Master)
        Modbus_TCP_Slave (Modbus TCP Slave)
```

Figure 52 – CODESYS connections configuration

Thus, the created C# application (Windows Form) includes a Modbus client, using a C# EasyModbus library, which connects to the softPLC through the ethernet IP address and a designated port (Figure 53). With this connection, the C# application is capable of reading the mapped output from the PLC and writing inputs to the PLC, that if assigned to a PLC variable can control it from a distance. Although, the input variables in the C# application are not assigned to any task, they were kept to demonstrate this useful tool.

In the C# application, to connect to the PLC, the IP address and port must be specified followed by pressing “connect”. When the system is connected, the label in front of the button will change its value to “Connected”, and the PLC’s values will appear in the application in the corresponding places. At the same time, the application will be writing these values to a Azure Cloud Storage table named “Zollern2”, similar to the one presented in section 4.3 (Figure 44, Figure 45, Figure 46 and Figure 48), in fact the code to establish the query process, the connection to the cloud table and creating a table entity is the same.

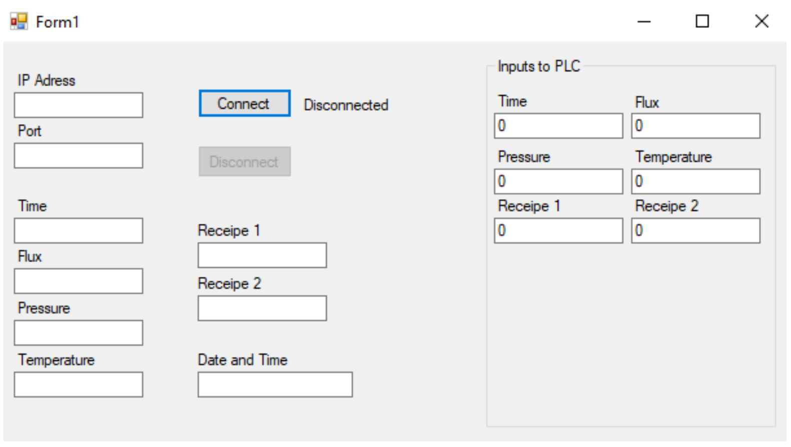


Figure 53 – PLC to Cloud C# application

When the application is reading the values, it registers them in an array of integers (Figure 54), as they come as “word” data type from the PLC and assembles them in a string array to be shown in the Form’s text fields.

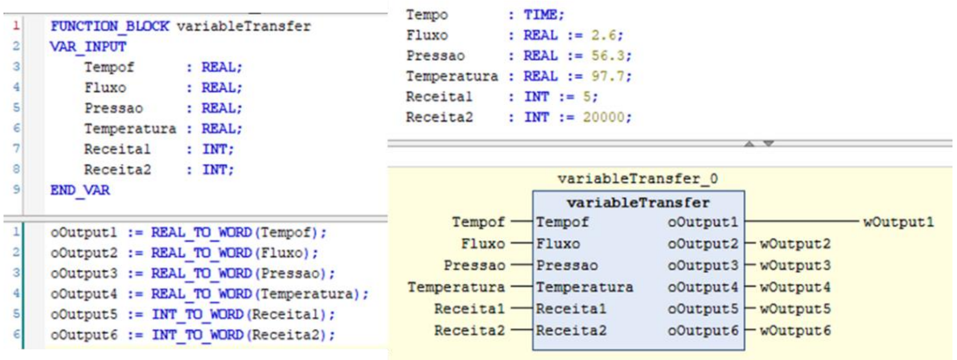
```
private void Read()
        {
            int[] myReadInputRegisters2;
            if (myModbusClient.Connected == true)
            {
                myReadInputRegisters2 = myModbusClient.ReadInputRegisters(0,10);
                string[] QW = new string[6];
                QW[0] = myReadInputRegisters2[0].ToString();
                QW[1] = myReadInputRegisters2[1].ToString();
                QW[2] = myReadInputRegisters2[2].ToString();
                QW[3] = myReadInputRegisters2[3].ToString();
                QW[4] = myReadInputRegisters2[4].ToString();
                QW[5] = myReadInputRegisters2[5].ToString();
                txttime.Text = QW[0];
                txtflux.Text = QW[1];
                txtpress.Text = QW[2];
                txtheat.Text = QW[3];
                txtreceipe1.Text = QW[4];
                txtreceipe2.Text = QW[5];
            }
        }
```

Figure 54 – Application reading PLC’s outputs

When the user presses the “Disconnect” button, it will stop reading or writing values to the PLC and sending values to the cloud table.

In the end, to combine both database transferring methods, this is, to a classic database and a cloud database, a function (Figure 55) was added to the CODESYS application presented in section 4.2 to assign mapped variables to those of PLC program’s variables, also creating a similar ethernet connection.

This allows that, when the PLC program is running, the C# application mentioned above connected and the XAMPP web server, also connected, the data coming from the PLC program is saved in both the MySQL corresponding table and the Azure cloud storage.



```
FUNCTION_BLOCK variableTransfer
VAR_INPUT
    Tempof      : REAL;
    Fluxo       : REAL;
    Pressao     : REAL;
    Temperatura : REAL;
    Receita1    : INT;
    Receita2    : INT;
END_VAR
```

```
oOutput1 := REAL_TO_WORD(Tempof);
oOutput2 := REAL_TO_WORD(Fluxo);
oOutput3 := REAL_TO_WORD(Pressao);
oOutput4 := REAL_TO_WORD(Temperatura);
oOutput5 := INT_TO_WORD(Receita1);
oOutput6 := INT_TO_WORD(Receita2);
```

```
Tempo       : TIME;
Fluxo       : REAL := 2.6;
Pressao     : REAL := 56.3;
Temperatura : REAL := 97.7;
Receita1    : INT := 5;
Receita2    : INT := 20000;
```

Figure 55 – Mapped variables function: Configuration (On the left) and Implementation (On the right)

# 5 Conclusions, future work perspectives and final remarks

## 5.1 Conclusions

The intention of this thesis was to implement the Industry 4.0 concept in ZC’s factory through the development of tools or solutions that could improve this factory’s process.

In the initial study of the manufacturing process, it became clear that ZC’s problem was not so much, the lack of sensorial equipment, as the HMIs of most machines show most of the important controlling and monitoring parameters of their process, but that this important data is not being saved or the little data that is, in fact, extracted is not treated or analysed in any way.

The extraction and analysis of this data is key to improve the efficiency of the manufacturing process, as this analysis will enable an optimization of the parameters and set points used. This optimization will result in a more efficient manufacturing process, increased product quality, and finally a positive impact on the economic results of the unit.

For the data analysis it is important to have a structured database where the data can be quickly analysed or consulted, but is also important to have data to consult, and so this work focused on finding a few alternatives for the data collection and integration process.

The first software tool is ready to be implemented if the company chooses to continue saving data to CSV files and later importing them to a database. But most of these CSV files still lack a lot of vital information from their corresponding processes, so one important issue that the project Zollern 4.0 together with ZC are trying to improve is the reprograming of the PLCs in order for them to store more useful information: Although this task is time consuming, it is also a vital one to position the factory in the I4.0 path.

If the company keeps the data gathering from the PLCs via CSV file, the first solution can be standardized through the reprograming of the PLCs and standardization of the syntax of the headers and the data types, this is, for example, writing float numbers always with the same decimal separator in all the files. Also, the scheduling of the transfer of CSV files would be important to keep the database updated.

However, since the reprogramming of the PLCs is already a step to take within the Zollern 4.0 project, this factory would benefit if instead of each machine saving data to a CSV record, it could save all the data directly, in real-time to a designated database, which could be altered when this task is performed. This change would be a paradigm shift towards the creation of a digital twin of the plant according to the Industrie 4.0 tenants, open numerous opportunities, such as the creation of real time monitoring through a plant central dashboard.

In this logic, the second software tool was developed to demonstrate one possible connection between a PLC and a database that is able to transfer real-time data without an intermediate file. Since the data would be directly logged on the database in real-time, with the proper software tools, the system could give real-time analysis of the process and prevent errors or defects.

This solution could be implemented in two different ways:

In the first approach, if the PLC has enough computing power, it can be directly connected to the database via ethernet, or else, if the PLC doesn’t have that ability, another PLC or bridge would be needed.

In the second approach to this solution, one option would be a low-power open-source computer like BeagleBone Black or Raspberry Pi, implemented in each machine. The low-power would run a CODESYS PLC runtime and would be connected to the machine’s PLC through the most appropriate fieldbus. The low power computer would therefore be responsible to save the data to the database, functioning as a bridge between the PLC and the database. In this approach, the interface between the database and the machine could be standardized since it is not directly downloaded to the machine’s PLC. These platforms, due to their flexibility, create a lot of different possibilities, some of them detailed in the future work section.

The third tool simulates the communication between a machine and a cloud database, this was considered an important area to research, as this type of storage and analysis solutions are becoming ever seen as the future for large database storage, especially for companies with worldwide locations, such as ZC. This kind of solution provides benefits in migration speed, database performance, access, protection, availability, scalability and risk mitigation, as the cloud services companies are responsible for keeping the information secured [55]. This tool also shows how easily a communication between a cloud and a machine is established, as easy as the communication between a machine and a standard database or even less arduous.

Thus, in terms of database lodging, although a standard SQL server will be enough to store, access the extracted data and to analyse this data, this kind of solution is getting outdated, especially in companies that don’t already have these systems. Instead of this, companies now prefer cloud databases, that, with all the advantages listed above, also can very quickly start working in network with other factories or companies for an integrated production system, don’t need to have physical servers installed, taking up space and only pay for the cloud services when they are using them. For example, when the company is not using data analysis software in the cloud, this service will not be charged; this is not possible if the company chooses the standard solution.

With this tendency in mind, the fourth tool was created, with capability of connecting the PLC to a cloud database through a CODESYS program and a C# application. In this solution, the CODESYS program would be implemented in the system in the same way as the second solution, and both implementing approaches described above would be valid. The C# application would run in the other end of the connection whether it would be the company’s server or a central computer, which would, by its turn, connect to the cloud via internet or intranet. An easier and faster way to do this would be to use an already existing tool, not available at the time of the development of this thesis, which much like in the second solution connects the CODESYS program directly with the Azure cloud, for example.

## 5.2 Future works

To improve the first solution, it would be necessary to normalize it through the normalization of the CSV file structure and schedule file transfers according to each machine’s process timings, as mentioned above. For example, if machines like the rotary ovens only writes one file per day, thus would only be necessary to upload its information to the database once a day, other machines like the wax injectors, will write several files per day, which would be more laborious.

In the second solutions case, if the low-power computer path is followed, it would be advantageous, for example, for the implementation of read and write data points in the wax tree gluing sector, this could help when the traceability system is implemented, since a simple way to increment the first read of the tree branch would be in the tree wax gluing sector, where the operator would have a small interface where the operator would input the wax tree’s ID, what type of piece and how many branches were glued, and even when the tree started to be assembled and was finished. All of this could be done with a simple CODESYS program and an HMI like the one shown in Figure 56 and its installation would not be costly.

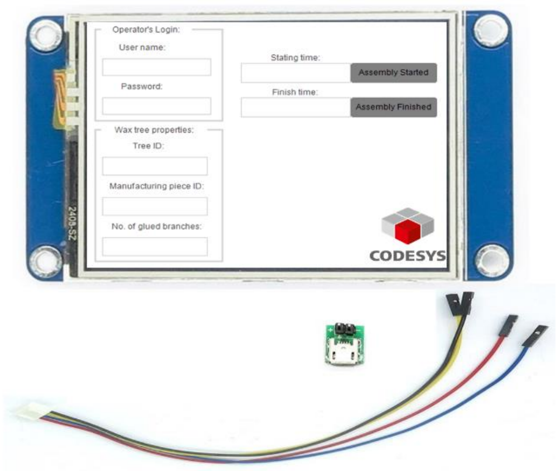


Figure 56 – Data introduction point – gluing sector – Mock-up

Moreover, between having to implement another PLC to run the CODESYS program or implementing this low-power computer, this second approach will cost less, taking much less space and be more flexible both software and hardware wise. In fact, implementing updates to the system would be easier and the low-power computers can operate in, barely, any machine.

In any of the described cases above, or even if another path is chosen, the company should consider following the OPC UA standard, when setting up its industrial communications. This will enhance the interoperability between machines and systems, as well as build a system that can easily be continuously updated and improved, always being able to exchange data between embedded objects regardless of the technology or platform used between these objects.

Lastly, to complete the possibilities of PLC data extraction, it is only missing the direct link between the PLC and the cloud through a CODESYS program (Figure 57), this tool already exists as a CODESYS library that can be purchased (Azure IoT Hub Client [56]), providing a rather simple way to connect to these platforms.

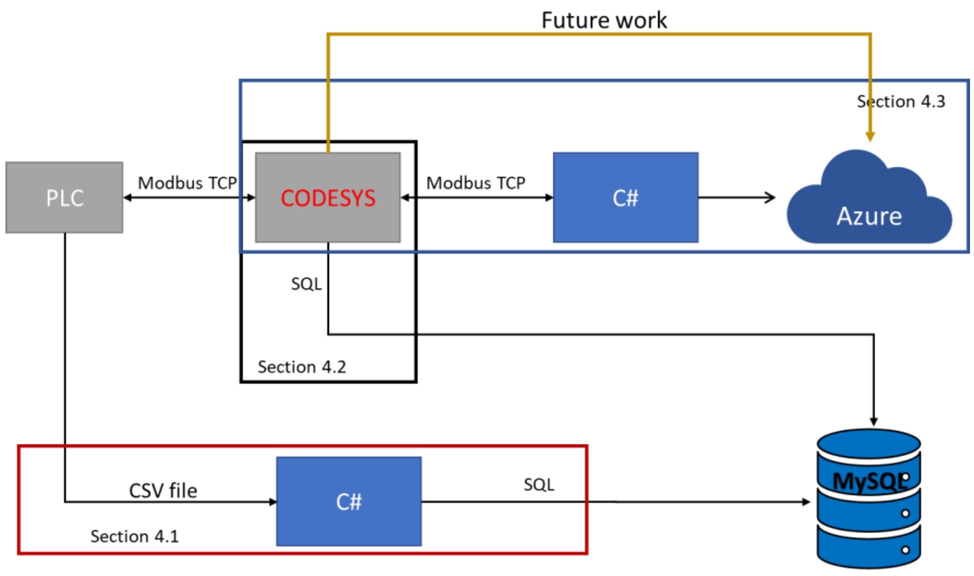


Figure 57 – Complete approach scheme

## 5.3 Final remarks

The solution presented in this thesis builds on top of a group of building blocks selected from a large number of options available today. The developed software applications show some of the possibilities available to develop a good data gathering system, that will be ever more important, and it will soon be indispensable, to move forward in the industrial technology. An additional characteristic to consider when choosing the most convenient path is the flexibility between hardware and software within the factory and the ability to share data and to manage it at a distance.

## References

[1] Zollern, “Zollern History,” Zollern, [Online]. Available: https://www.zollern.com/en/company/history.html. [Acedido em 27 1 2019].

[2] Zollern, “Zollern Worldwide,” Zollern, [Online]. Available: https://www.zollern.com/en/company/zollern-worldwide.html. [Acedido em 27 1 2019].

[3] Zollern, “Fundição por cera perdida,” Zollern, [Online]. Available: https://www.zollern.com/br/fundicao-forjado/fundicao-por-cera-perdida.html. [Acedido em 27 1 2019].

[4] L. Technologies, “Boilerclave®,” LBBC Technologies, [Online]. Available: http://lbbctechnologies.com/product/. [Acedido em 27 1 2019].

[5] “Investment Casting Process,” [Online]. Available: http://www.castingquality.com/casting-technology/investment-casting-tech/investment-casting-process.html. [Acedido em 27 1 2019].

[6] CLEVERISM, “Industry 4.0: Definition, Design Principles, Challenges, and the Future of Employment,” CLEVERISM, [Online]. Available: https://www.cleverism.com/industry-4-0/. [Acedido em 27 1 2019].

[7] i-SCOOP, “Industry 4.0: the fourth industrial revolution – guide to Industrie 4.0,” i-SCOOP, [Online]. Available: https://www.i-scoop.eu/industry-4-0/. [Acedido em 27 1 2019].

[8] W. D. L. W. HENNING KAGERMANN, “Industry 4.0: With the Internet of Things on the way to the 4th industrial revolution,” vdi nachrichten, [Online]. Available: https://www.vdi-nachrichten.com/Technik-Gesellschaft/Industrie-40-Mit-Internet-Dinge-Weg-4-industriellen-Revolution. [Acedido em 27 1 2019].

[9] B. Marr, “What is Industry 4.0? Here's A Super Easy Explanation For Anyone,” Forbes, [Online]. Available: https://www.forbes.com/sites/bernardmarr/2018/09/02/what-is-industry-4-0-heres-a-super-easy-explanation-for-anyone/#3ca57caa9788. [Acedido em 27 1 2019].

[10] B. Marr, “What Everyone Must Know About Industry 4.0,” Forbes, [Online]. Available: https://www.forbes.com/sites/bernardmarr/2016/06/20/what-everyone-must-know-about-industry-4-0/#f03d9f4795f7. [Acedido em 27 1 2019].

[11] T. P. B. O. Mario Hermann, “Design Principles for Industrie 4.0 Scenarios,” em *49th Hawaii International Conference on System Sciences*, 2016.

[12] J. D. M. Siddhartha Kumar Khaitan, “Design Techniques and Applications of CyberPhysical Systems: A Survey,” IEEE.

[13] C. F. Friedemann Mattern, “From the Internet of Computers to the Internet of Things,” Institute for Pervasive Computing, ETH Zurich, Zurich.

[14] M. L. R. R. Michael Chui, “The Internet of Things,” McKinsey, [Online]. Available: https://www.mckinsey.com/industries/high-tech/our-insights/the-internet-of-things. [Acedido em 27 1 2019].

[15] “The Internet of Services—Intro to our Tech,” Medium, [Online]. Available: https://medium.com/@iostoken/the-internet-of-services-intro-to-our-tech-e91abfb13b8c. [Acedido em 27 1 2019].

[16] “The internet of services in industrie4.0,” Concept Systems, [Online]. Available: https://conceptsystemsinc.com/the-internet-of-services-in-industrie-4-0/. [Acedido em 27 1 2019].

[17] E. Knorr, “What is cloud computing? Everything you need to know now,” InfoWorld, [Online]. Available: https://www.infoworld.com/article/2683784/cloud-computing/what-is-cloud-computing.html. [Acedido em 27 1 2019].

[18] AWS, “What is Cloud Computing?,” Amazon, [Online]. Available: https://aws.amazon.com/what-is-cloud-computing/. [Acedido em 27 1 2019].

[19] V. IT, “INDUSTRY 4.0 IS THE NEXT INDUSTRIAL REVOLUTION.,” VM9 IT, [Online]. Available: https://www.vm9it.com/industry.html. [Acedido em 27 1 2019].

[20] F. Almada-Lobo, “The Industry 4.0 revolution and the future of Manufacturing Execution Systems (MES),” *Journal of Innovation Management,* pp. 16-21, 2015.

[21] I. Analog Devices, “Accelerating the Path to Industry 4.0,” Analog Devices, Inc., [Online]. Available: https://www.analog.com/en/applications/markets/industrial-automation-technology-pavilion-home/industry-4-pt-0.html. [Acedido em 27 1 2019].

[22] P. Beynon-Davies, Database Systems, Palgrave MacMillan, 2004.

[23] Office of asset management, Data Integration Glossary, U.S. Department of Transportation, 2001.

[24] J. B. WALDNER, CIM—Principles of Computer Integrated Manufacturing, Chichester: John Wiley & Sons,, 1992.

[25] E. F. CODD, “A Relational Model of Data for Large Shared Data Banks,” IBM Research Laboratory, San Jose, California, 1970.

[26] O. a. i. affiliates, “XML DB Developer's Guide,” Oracle, [Online]. Available: https://docs.oracle.com/cd/B19306_01/appdev.102/b14259/xdb05sto.htm. [Acedido em 27 1 2019].

[27] I. InfluxData, “InfluxDB,” InfluxData, Inc., [Online]. Available: https://www.influxdata.com/time-series-platform/influxdb/. [Acedido em 27 1 2019].

[28] O. K. SHEN YIN, “Big Data for Modern Industry:Challenges and Trends,” *IEEE,* vol. 103, nº 2, pp. 143-146, 2015.

[29] J. G. a. D. Reinsel, “THE DIGITAL UNIVERSE IN 2020: Big Data, Bigger Digital Shadow s, and Biggest Grow th in the Far East,” IDC, 2012.

[30] L. a. AutomationDirect.com, “History of the PLC,” AutomationDirect.com, [Online]. Available: https://library.automationdirect.com/history-of-the-plc/. [Acedido em 27 1 2019].

[31] J. A. G. Barroso, “New Section: Automation photos,” [Online]. Available: https://www.noeju.com/tag/modicon-084/. [Acedido em 27 1 2019].

[32] unitronics, “What is the definition of "PLC"?,” unitronics, [Online]. Available: https://unitronicsplc.com/what-is-plc-programmable-logic-controller/. [Acedido em 27 1 2019].

[33] P. Service, “Automation software,” Project Service, [Online]. Available: https://www.projectservice.com/servizi/16/AutomazioneUK/UK. [Acedido em 27 1 2019].

[34] M. T. Karl-Heinz John, “IEC 61131-3: Programming industrial automation systems,” Springer, 2001.

[35] I. E. Commission, “IEC 61131 Part 3: Programming languages”. Universal Patente IEC 61131-3:2003(E) , 01 2003.

[36] L. B. a. E. Bryan, Programmable Controllers: THEORY AND IMPLEMENTATION, Atlanta, Georgia: Industrial Text Company, 1997.

[37] D. B. Swift, “Computer Organisation & Program Execution,” [Online]. Available: https://cs.anu.edu.au/courses/comp6300/lectures/week-10/#/42. [Acedido em 27 1 2019].

[38] E. W. D. R. ,. J. P. Steve Mackay, Practical Industrial Data Networks: Design, Installation and Troubleshooting, Perth, Australia : IDC Technologies, 2004.

[39] S. Modbus, “Simply Modbus,” Simply Modbus, [Online]. Available: http://www.simplymodbus.ca/faq.htm#Modbus. [Acedido em 27 1 2019].

[40] I. Control Solutions, “Modbus 101 - Introduction to Modbus,” Control Solutions, Inc., [Online]. Available: https://www.csimn.com/CSI_pages/Modbus101.html. [Acedido em 27 1 2019].

[41] Modbus-IDA, MODBUS Messaging on TCP/IP Implementation Guide V1.0b, Modbus-IDA , 2006.

[42] Siemens, Easy PROFINET implementation, Siemens.

[43] P. N. America, “SYSTEM DESCRIPTION: PROFINET TECHNOLOGY AND APPLICATION,” PI North America, [Online]. Available: https://us.profinet.com/documentation/system-description-profinet-technology-and-application/. [Acedido em 27 1 2019].

[44] P. University, “PROFINET Communication Channels,” PROFINET University, [Online]. Available: https://profinetuniversity.com/profinet-basics/profinet-communication-channels/. [Acedido em 27 1 2019].

[45] P. University, “What is PROFINET?,” PROFINET University, [Online]. Available: https://profinetuniversity.com/profinet-basics/definition-profinet/. [Acedido em 27 1 2019].

[46] O. foundation, “What is OPC?,” OPC Foundation, 2019. [Online]. Available: https://opcfoundation.org/about/what-is-opc/. [Acedido em 18 2 2019].

[47] O. Foundation, “OPC Classic,” OPC Foundation, [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-classic/. [Acedido em 18 2 2019].

[48] O. Foundation, “Unified Architecture,” OPC Foundation, 2019. [Online]. Available: https://opcfoundation.org/about/opc-technologies/opc-ua/. [Acedido em 18 2 2019].

[49] N. Corp., “OPC and OPC UA explained,” Novotek Corp., 2019. [Online]. Available: https://www.novotek.com/en/solutions/kepware-communication-platform/opc-and-opc-ua-explained. [Acedido em 18 2 2019].

[50] G. Guanabara, “Curso MySQL,” [Online]. Available: https://www.youtube.com/watch?v=Ofktsne-utM&list=PLHz_AreHm4dkBs-795Dsgvau_ekxg8g1r. [Acedido em 27 1 2019].

[51] A. Friends, “About,” Apache Friends, [Online]. Available: https://www.apachefriends.org/about.html. [Acedido em 27 1 2019].

[52] T. P. Group, “PHP,” The PHP Group, [Online]. Available: http://php.net/. [Acedido em 27 1 2019].

[53] phpMyAdmin, “Bringing MySQL to the web,” phpMyAdmin, [Online]. Available: https://www.phpmyadmin.net/. [Acedido em 27 1 2019].

[54] M. Rouse, “Definition of C#,” TechTarget, [Online]. Available: https://searchwindevelopment.techtarget.com/definition/C. [Acedido em 27 1 2019].

[55] N. C. D. M. Services, “7 Benefits of Using a Cloud Database Service,” Hosting, [Online]. Available: https://www.ntirety.com/7-benefits-of-using-a-cloud-database-service/. [Acedido em 27 1 2019].

[56] 3.-S. S. S. GmbH, “Azure IoT Hub Client,” 3S-Smart Software Solutions GmbH, [Online]. Available: https://store.codesys.com/codesys-azure-iot-hub-client.html?___store=en&___from_store=default. [Acedido em 27 1 2019].

[57] C. E. B. Thomas M. Connolly, Database Systems: A Practical Approach to Design, Implementation, and Management, Addison Wesley 4th edition.

[58] W. K. P. L. Nigel Derrett, “Some Aspects of Operations in an Object-Oriented Database,” *quarterly bulletin of the IEEE computer society technical committee on Database Engineering,* vol. VOL. 8, nº no.4, pp. 66-74, 1986.

[59] M. L. P. G. M. W. J. J. P. E. a. M. H. Michael Rüßmann, “Industry 4.0: The Future of Productivity and Growth in Manufacturing Industries,” Munich, 2015.

[60] D. Clark, Beginning C# Object-Oriented Programming, Apress.

[61] E. A. Lee, “Cyber Physical Systems: Design Challenges,” University of California , Berkeley, 2008.

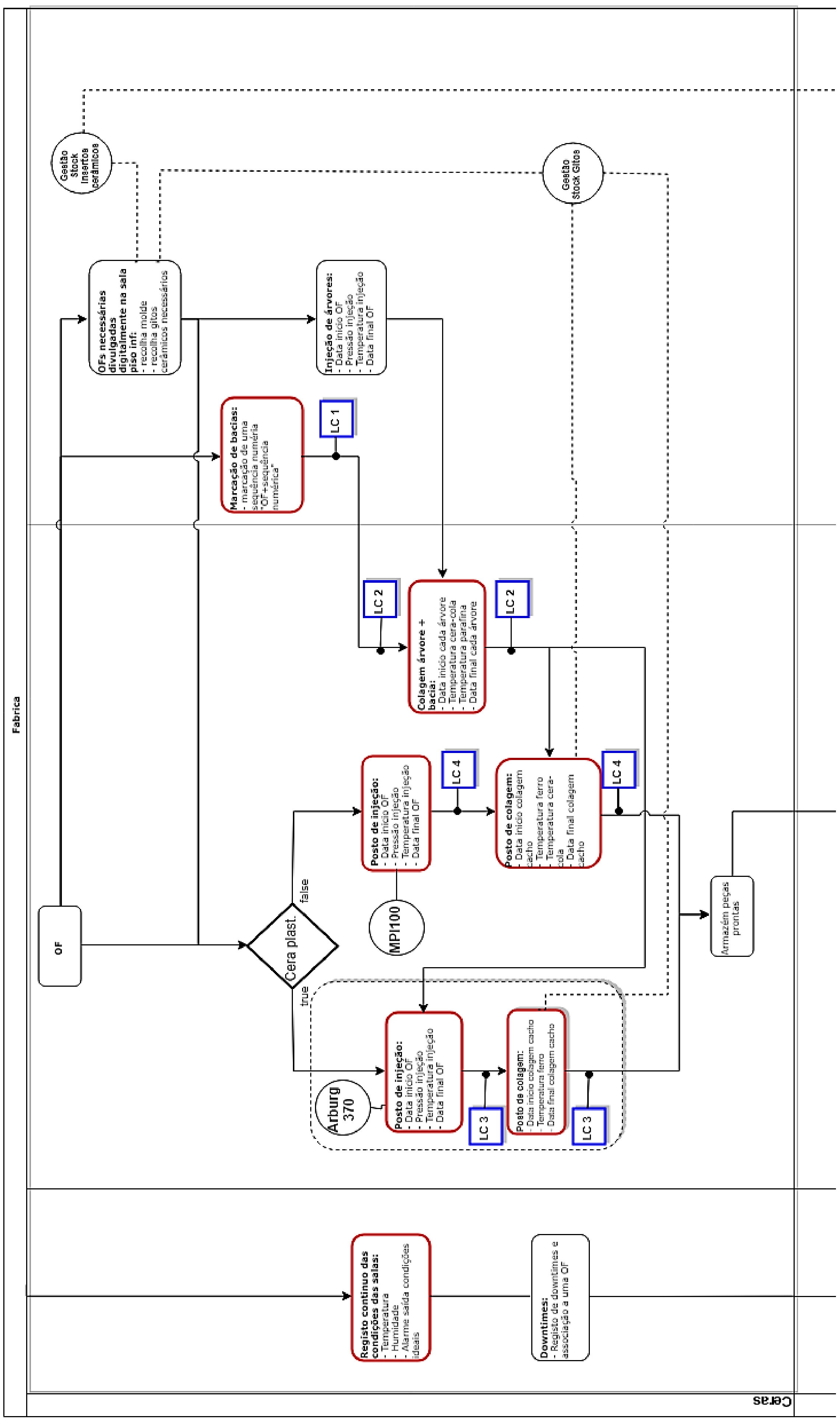
[62] J.L.DionR.K.BuhrM.Sahoo, “Vacuum pouring,” em *F. Weinberg International Symposium on Solidification Processing,* Ontario, 1990.

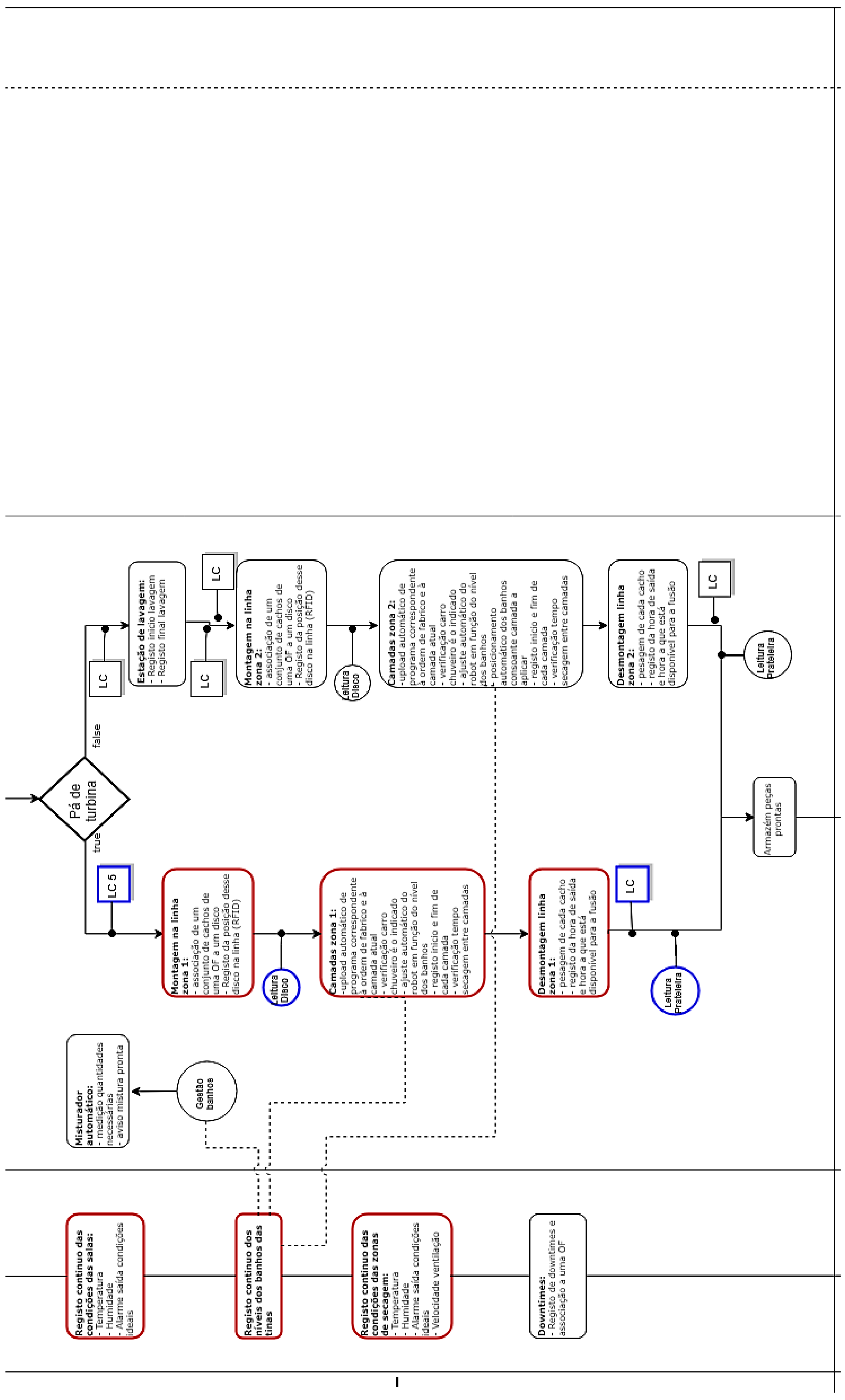
[63] “File:Relational database terms.svg,” [Online]. Available: https://commons.wikimedia.org/wiki/File:Relational_database_terms.svg. [Acedido em 27 1 2019].

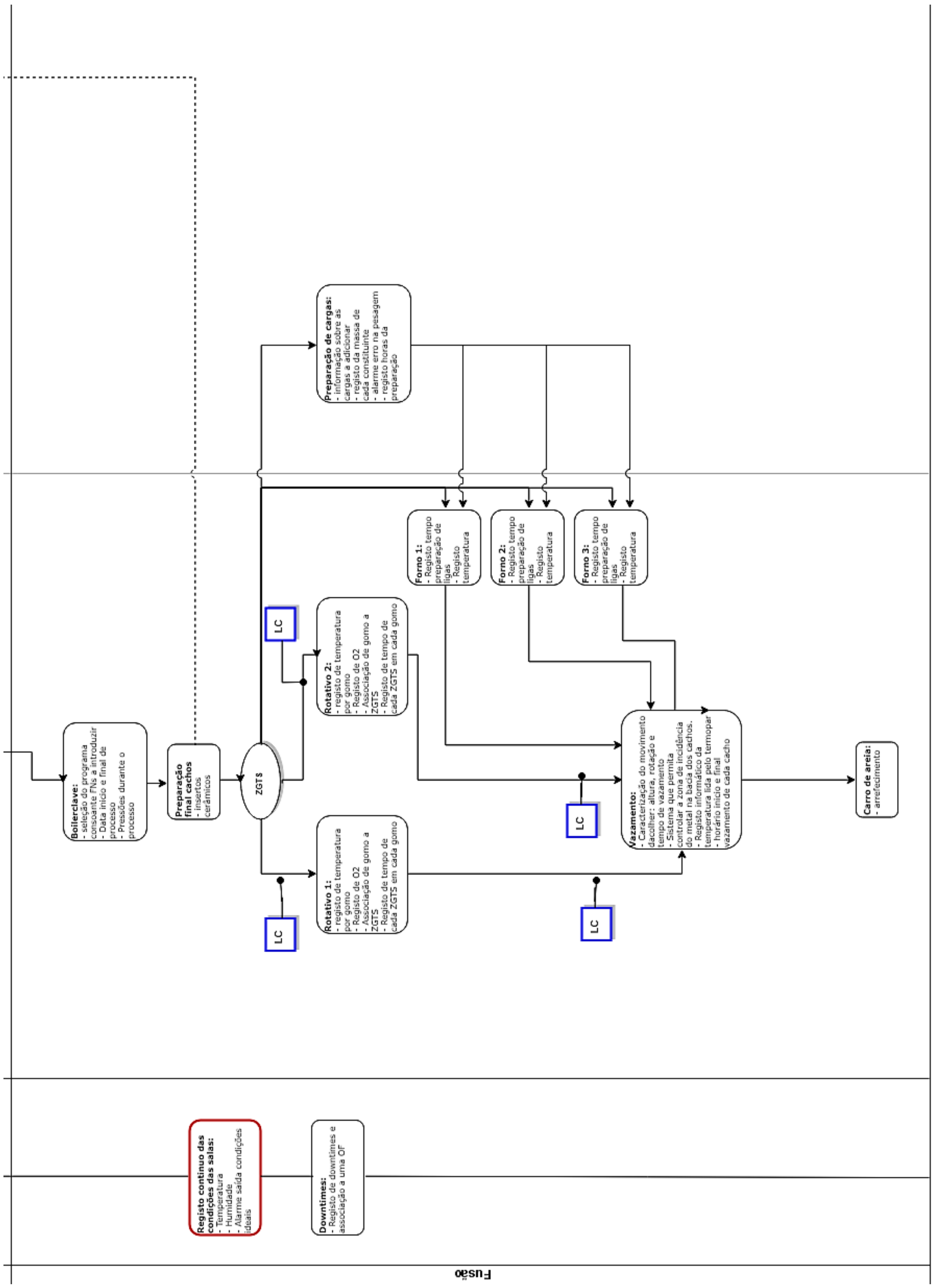
[64] PLCopen, “PLCopen,” PLCopen, [Online]. Available: http://plcopen.org/. [Acedido em 27 1 2019].

[65] Zollern, “Zollern Porto,” Zollern, [Online]. Available: https://www.zollern.com/en/company/zollern-worldwide/zollern-porto.html. [Acedido em 27 1 2019].

## ATTACHMENT A: Zollern’s process flowchart


Fabrica
Ceras
OF
Gestão Stock Insertos cerâmicos
OFs necessárias divulgadas digitalmente na sala piso inf:
- recolha molde
- recolha gitos cerâmicos necessários
Gestão Stock Gitos
Injeção de árvores:
- Data início OF
- Pressão injeção
- Temperatura injeção
- Data final OF
Marcação de bacias:
- marcação de uma sequência numérica "OF+sequência numérica"
LC 1
LC 2
LC 2
Colagem árvore + bacia:
- Data início cada árvore
- Temperatura cera-cola
- Temperatura parafina
- Data final cada árvore
Cera plast.
false
true
Posto de injeção:
- Data início OF
- Pressão injeção
- Temperatura injeção
- Data final OF
LC 4
MPI100
Posto de colagem:
- Data início colagem cacho
- Temperatura ferro
- Temperatura cera-cola
- Data final colagem cacho
LC 4
Armazém peças prontas
Arburg 370
Posto de injeção:
- Data início OF
- Pressão injeção
- Temperatura injeção
- Data final OF
LC 3
Posto de colagem:
- Data início colagem cacho
- Temperatura ferro
- Data final colagem cacho
LC 3
Registo contínuo das condições das salas:
- Temperatura
- Humidade
- Alarme saída condições ideais
Downtimes:
- Registo de downtimes e associação a uma OF


Pá de turbina
true
false
LC 5
LC
Estação de lavagem:
- Registo início lavagem
- Registo final lavagem
Montagem na linha zona 1:
- associação de um conjunto de cachos de uma OF a um disco
- Registo da posição desse disco na linha (RFID)
Montagem na linha zona 2:
- associação de um conjunto de cachos de uma OF a um disco
- Registo da posição desse disco na linha (RFID)
Leitura Disco
Camadas zona 1:
-upload automático de programa correspondente à ordem de fabrico e à camada atual
- verificação carro chuveiro é o indicado
- ajuste automático do robot em função do nível dos banhos
- registo início e fim de cada camada
- verificação tempo secagem entre camadas
Camadas zona 2:
-upload automático de programa correspondente à ordem de fabrico e à camada atual
- verificação carro chuveiro é o indicado
- ajuste automático do robot em função do nível dos banhos
- posicionamento automático dos banhos consoante camada a aplicar
- registo início e fim de cada camada
- verificação tempo secagem entre camadas
Desmontagem linha zona 1:
- pesagem de cada cacho
- registo da hora de saída e hora a que está disponível para a fusão
Desmontagem linha zona 2:
- pesagem de cada cacho
- registo da hora de saída e hora a que está disponível para a fusão
Leitura Prateleira
Armazém peças prontas
Misturador automático:
- medição quantidades necessárias
- aviso mistura pronta
Gestão banhos
Registo contínuo das condições das salas:
- Temperatura
- Humidade
- Alarme saída condições ideais
Registo contínuo dos níveis dos banhos das tinas
Registo contínuo das condições das zonas de secagem:
- Temperatura
- Humidade
- Alarme saída condições ideais
- Velocidade ventilação
Downtimes:
- Registo de downtimes e associação a uma OF


Preparação de cargas:
- informação sobre as cargas a adicionar
- registo da massia de cada constituinte
- alarme erro na pesagem
- registo horas da preparação
Forno 1:
- Registo tempo preparação de ligas
- Registo temperatura
Forno 2:
- Registo tempo preparação de ligas
- Registo temperatura
Forno 3:
- Registo tempo preparação de ligas
- Registo temperatura
Boilerclave:
- seleção do programa consoante FNs a introduzir
- Data início e final de processo
- Pressões durante o processo
Preparação final cachos
- insertos cerâmicos
ZGTS
LC
Rotativo 2:
- registo de temperatura por gomo
- Registo de O2
- Associação de gomo a ZGTS
- Registo de tempo de cada ZGTS em cada gomo
Rotativo 1:
- registo de temperatura por gomo
- Registo de O2
- Associação de gomo a ZGTS
- Registo de tempo de cada ZGTS em cada gomo
LC
LC
LC
Vazamento:
- Caracterização do movimento dacolher: altura, rotação e tempo de vazamento
- Sistema que permita controlar a zona de incidência do metal na bacia dos cachos.
- Registo informático da temperatura lida pelo termopar
- horário início e final vazamento de cada cacho
Carro de areia:
- arrefecimento
Registo contínuo das condições das salas:
- Temperatura
- Humidade
- Alarme saída condições ideais
Downtimes:
- Registo de downtimes e associação a uma OF
Fusão

## ATTACHMENT B: CSV files

***Ceramics department – Alarms***

```
DATA;HORA;ID_ALARME;EVENTO;DESCRIÇÃO
2018-01-20;10:23:31;1;OK;* * * * EMERGÊNCIA * * *
2018-01-20;10:23:31;2;OK;Disparo Térmico Parcial Ventilador M1
2018-01-20;10:23:31;3;OK;Disparo Térmico Parcial Ventilador M2
2018-01-20;10:23:31;4;OK;Disparo Térmico Parcial Ventilador M3
2018-01-20;10:23:31;5;OK;Disparo Térmico Parcial Ventilador M4
2018-01-20;10:23:31;6;OK;Disparo Térmico Parcial Ventilador M5
2018-01-20;10:23:31;7;OK;Disparo Térmico Parcial Ventilador M6
2018-01-20;10:23:31;8;OK;Disparo Térmico Parcial Ventilador M7
2018-01-20;10:23:31;9;OK;Disparo Térmico Parcial Ventilador M8
2018-01-20;10:23:31;10;OK;Disparo Térmico Parcial Ventilador M9
2018-01-20;10:23:31;11;OK;Disparo Térmico Parcial Ventilador M10
2018-01-20;10:23:31;12;OK;Disparo Térmico Parcial Ventilador M11
2018-01-20;10:23:31;13;OK;Disparo Térmico Parcial Ventilador M12
2018-01-20;10:23:31;14;OK;Disparo Térmico Parcial Ventilador M13
2018-01-20;10:23:31;15;OK;Disparo Térmico Parcial Ventilador M14
2018-01-20;10:23:31;16;OK;Disparo Térmico Parcial Ventilador M15
2018-01-20;10:23:31;17;OK;Disparo Térmico Parcial Ventilador M16
2018-01-20;10:23:31;18;OK;Disparo Térmico Parcial Ventilador M17
2018-01-20;10:23:31;19;OK;Disparo Térmico Parcial Ventilador M18
2018-01-20;10:23:31;20;OK;Disparo Térmico Parcial Ventilador M19
2018-01-20;10:23:31;21;OK;Disparo Térmico Parcial Ventilador M20
2018-01-20;10:23:31;53;OK;SEQUÊNCIA DE FASE
2018-01-20;17:06:36;1;OK;* * * * EMERGÊNCIA * * *
2018-01-20;17:06:36;2;OK;Disparo Térmico Parcial Ventilador M1
2018-01-20;17:06:36;3;OK;Disparo Térmico Parcial Ventilador M2
2018-01-20;17:06:36;4;OK;Disparo Térmico Parcial Ventilador M3
2018-01-20;17:06:36;5;OK;Disparo Térmico Parcial Ventilador M4
2018-01-20;17:06:36;6;OK;Disparo Térmico Parcial Ventilador M5
2018-01-20;17:06:36;7;OK;Disparo Térmico Parcial Ventilador M6
2018-01-20;17:06:36;8;OK;Disparo Térmico Parcial Ventilador M7
2018-01-20;17:06:36;9;OK;Disparo Térmico Parcial Ventilador M8
2018-01-20;17:06:36;10;OK;Disparo Térmico Parcial Ventilador M9
2018-01-20;17:06:36;11;OK;Disparo Térmico Parcial Ventilador M10
2018-01-20;17:06:36;12;OK;Disparo Térmico Parcial Ventilador M11
2018-01-20;17:06:36;13;OK;Disparo Térmico Parcial Ventilador M12
2018-01-20;17:06:36;14;OK;Disparo Térmico Parcial Ventilador M13
2018-01-20;17:06:36;15;OK;Disparo Térmico Parcial Ventilador M14
2018-01-20;17:06:36;16;OK;Disparo Térmico Parcial Ventilador M15
2018-01-20;17:06:36;17;OK;Disparo Térmico Parcial Ventilador M16
2018-01-20;17:06:36;18;OK;Disparo Térmico Parcial Ventilador M17
2018-01-20;17:06:36;19;OK;Disparo Térmico Parcial Ventilador M18
2018-01-20;17:06:36;20;OK;Disparo Térmico Parcial Ventilador M19
2018-01-20;17:06:36;21;OK;Disparo Térmico Parcial Ventilador M20
2018-01-20;17:06:36;53;OK;SEQUÊNCIA DE FASE
2018-02-01;15:53:56;1;ALARME ;* * * * EMERGÊNCIA * * *
2018-02-01;15:55:23;1;OK;* * * * EMERGÊNCIA * * *
2018-02-03;17:57:02;1;ALARME ;* * * * EMERGÊNCIA * * *
2018-02-04;23:36:21;1;OK;* * * * EMERGÊNCIA * * *
2018-02-24;14:41:10;1;ALARME ;* * * * EMERGÊNCIA * * *
2018-02-25;12:55:43;53;ALARME ;SEQUÊNCIA DE FASE
```

***Ceramics department – Line***

```
DATA;HORA;LINHA;TEMPO_ATE;EVENTO
2018-03-12;00:01:22;B;1495;MARCHA_FW
2018-03-12;00:01:42;B;20;PARAGEM
2018-03-12;00:04:58;B;197;MARCHA_FW
2018-03-12;00:05:18;B;20;PARAGEM
2018-03-12;00:08:35;B;196;MARCHA_FW
2018-03-12;00:08:55;B;20;PARAGEM
2018-03-12;00:12:51;B;237;MARCHA_FW
2018-03-12;00:13:11;B;20;PARAGEM
2018-03-12;00:16:33;B;201;MARCHA_FW
2018-03-12;00:16:53;B;20;PARAGEM
2018-03-12;00:20:14;B;201;MARCHA_FW
2018-03-12;00:20:34;B;20;PARAGEM
2018-03-12;00:26:41;B;367;MARCHA_FW
2018-03-12;00:27:01;B;20;PARAGEM
2018-03-12;00:30:22;B;202;MARCHA_FW
2018-03-12;00:30:43;B;20;PARAGEM
2018-03-12;00:33:28;A;8866;MARCHA_FW
2018-03-12;00:33:52;B;190;MARCHA_FW
2018-03-12;00:33:54;A;27;PARAGEM
2018-03-12;00:33:59;A;5;MARCHA_FW
2018-03-12;00:34:12;B;20;PARAGEM
2018-03-12;00:34:28;A;29;PARAGEM
2018-03-12;00:34:50;A;22;MARCHA_FW
2018-03-12;00:35:33;A;43;PARAGEM
2018-03-12;00:37:21;B;188;MARCHA_FW
2018-03-12;00:37:41;B;20;PARAGEM
2018-03-12;00:39:13;A;220;MARCHA_FW
2018-03-12;00:40:38;A;85;PARAGEM
2018-03-12;00:41:28;B;228;MARCHA_FW
2018-03-12;00:41:48;B;20;PARAGEM
2018-03-12;00:45:29;B;221;MARCHA_FW
2018-03-12;00:45:49;B;20;PARAGEM
2018-03-12;00:49:31;B;221;MARCHA_FW
2018-03-12;00:49:50;B;20;PARAGEM
2018-03-12;00:53:32;B;222;MARCHA_FW
2018-03-12;00:53:51;B;19;PARAGEM
2018-03-12;00:57:33;B;222;MARCHA_FW
2018-03-12;00:57:53;B;20;PARAGEM
2018-03-12;01:01:34;B;221;MARCHA_FW
2018-03-12;01:01:54;B;19;PARAGEM
2018-03-12;01:05:35;B;222;MARCHA_FW
2018-03-12;01:05:55;B;19;PARAGEM
2018-03-12;01:10:25;B;270;MARCHA_FW
2018-03-12;01:10:44;B;20;PARAGEM
2018-03-12;01:14:00;B;196;MARCHA_FW
2018-03-12;01:14:20;B;20;PARAGEM
2018-03-12;01:19:55;A;2357;MARCHA_FW
2018-03-12;01:19:55;B;334;MARCHA_FW
2018-03-12;01:20:14;B;20;PARAGEM
2018-03-12;01:20:23;A;27;PARAGEM
```

***Ceramics department – Ventilators***

```
DATA;HORA;LINHA;VENTILADOR;EVENTO
2018-03-12;00:08:55;B;M7;ARRANQUE
2018-03-12;00:34:12;B;M8;ARRANQUE
2018-03-12;00:49:50;B;M9;ARRANQUE
2018-03-12;01:05:12;A;M16;PARAGEM
2018-03-12;01:05:12;A;M17;PARAGEM
2018-03-12;01:05:12;A;M18;PARAGEM
2018-03-12;01:05:13;A;M13;PARAGEM
2018-03-12;01:05:13;A;M19;PARAGEM
2018-03-12;01:05:14;A;M14;PARAGEM
2018-03-12;01:05:14;A;M15;PARAGEM
2018-03-12;01:10:44;B;M10;ARRANQUE
2018-03-12;01:10:45;B;M10;ARRANQUE
2018-03-12;01:19:57;A;M16;ARRANQUE
2018-03-12;01:20:02;A;M13;ARRANQUE
2018-03-12;01:20:07;A;M14;ARRANQUE
2018-03-12;01:20:12;A;M15;ARRANQUE
2018-03-12;01:20:17;A;M17;ARRANQUE
2018-03-12;01:20:22;A;M18;ARRANQUE
2018-03-12;01:20:27;A;M19;ARRANQUE
2018-03-12;01:31:53;B;M11;ARRANQUE
2018-03-12;02:04:25;A;M13;PARAGEM
2018-03-12;02:21:14;A;M14;PARAGEM
2018-03-12;02:46:20;A;M15;PARAGEM
2018-03-12;03:03:04;A;M16;PARAGEM
2018-03-12;03:23:32;A;M17;PARAGEM
2018-03-12;03:59:44;A;M18;PARAGEM
2018-03-12;04:16:44;A;M18;ARRANQUE
2018-03-12;05:56:25;A;M13;ARRANQUE
2018-03-12;06:11:14;A;M14;ARRANQUE
2018-03-12;06:32:48;A;M18;PARAGEM
2018-03-12;06:36:15;A;M13;PARAGEM
2018-03-12;06:36:15;A;M14;PARAGEM
2018-03-12;06:36:15;A;M19;PARAGEM
2018-03-12;06:43:30;A;M19;ARRANQUE
2018-03-12;06:43:35;A;M13;ARRANQUE
2018-03-12;06:43:40;A;M14;ARRANQUE
2018-03-12;06:54:22;A;M15;ARRANQUE
2018-03-12;07:15:37;A;M16;ARRANQUE
2018-03-12;07:29:45;A;M17;ARRANQUE
2018-03-12;07:40:22;A;M18;ARRANQUE
2018-03-12;08:14:57;B;M1;PARAGEM
2018-03-12;08:21:50;B;M1;ARRANQUE
2018-03-12;08:22:18;B;M2;PARAGEM
2018-03-12;08:22:19;B;M2;PARAGEM
2018-03-12;08:38:05;B;M2;ARRANQUE
2018-03-12;08:52:27;A;M19;PARAGEM
2018-03-12;09:47:37;B;M1;PARAGEM
2018-03-12;09:53:55;B;M2;PARAGEM
2018-03-12;10:00:48;B;M1;ARRANQUE
2018-03-12;10:14:33;B;M2;ARRANQUE
```

***Ceramics department – Weekend breaks***

```
DATA;HORA;LINHA;EVENTO
2018-03-02;23:13:20;B;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:13:23;A;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:13:28;A;Paragem FIM de SEMANA Desactivado
2018-03-02;23:13:30;B;Paragem FIM de SEMANA Desactivado
2018-03-02;23:13:35;B;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:13:37;A;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:14:01;A;Paragem FIM de SEMANA Desactivado
2018-03-02;23:14:03;B;Paragem FIM de SEMANA Desactivado
2018-03-02;23:14:17;B;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:14:20;A;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:14:22;A;Paragem FIM de SEMANA Desactivado
2018-03-02;23:14:24;B;Paragem FIM de SEMANA Desactivado
2018-03-02;23:14:28;B;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:14:30;A;Paragem FIM de SEMANA ACTIVADO
2018-03-02;23:14:45;A;Paragem FIM de SEMANA Desactivado
2018-03-02;23:14:49;A;Paragem FIM de SEMANA ACTIVADO
2018-03-03;06:14:27;B;Paragem de FIM de SEMANA
2018-03-03;06:14:48;A;Paragem de FIM de SEMANA
2018-03-03;10:20:18;A;Paragem FIM de SEMANA Desactivado
2018-03-03;10:20:18;A;ARRANQUE de FIM de SEMANA
2018-03-03;10:20:22;B;Paragem FIM de SEMANA Desactivado
2018-03-03;10:20:22;B;ARRANQUE de FIM de SEMANA
2018-03-03;11:16:33;B;Paragem FIM de SEMANA ACTIVADO
2018-03-03;11:16:36;A;Paragem FIM de SEMANA ACTIVADO
2018-03-03;18:16:32;B;Paragem de FIM de SEMANA
2018-03-03;18:16:35;A;Paragem de FIM de SEMANA
2018-03-04;23:08:12;A;Paragem FIM de SEMANA Desactivado
2018-03-04;23:08:12;A;ARRANQUE de FIM de SEMANA
2018-03-04;23:08:14;B;Paragem FIM de SEMANA Desactivado
2018-03-04;23:08:14;B;ARRANQUE de FIM de SEMANA
2018-03-05;01:11:18;A;PARAGEM - MANUAL
2018-03-05;01:11:18;B;PARAGEM - MANUAL
2018-03-05;01:17:43;A;ARRANQUE - AUTO
2018-03-05;01:17:43;B;ARRANQUE - AUTO
2018-03-05;03:00:16;A;PARAGEM - MANUAL
2018-03-05;03:00:16;B;PARAGEM - MANUAL
2018-03-05;03:19:25;A;ARRANQUE - AUTO
2018-03-05;03:19:25;B;ARRANQUE - AUTO
2018-03-08;12:52:12;A;PARAGEM - MANUAL
2018-03-08;12:52:12;B;PARAGEM - MANUAL
2018-03-08;12:52:27;A;ARRANQUE - AUTO
2018-03-08;12:52:27;B;ARRANQUE - AUTO
2018-03-08;12:52:32;B;Paragem FIM de SEMANA ACTIVADO
2018-03-08;12:52:41;B;Paragem FIM de SEMANA Desactivado
2018-03-08;12:52:46;A;PARAGEM - MANUAL
2018-03-08;12:52:46;B;PARAGEM - MANUAL
2018-03-08;13:42:36;A;ARRANQUE - AUTO
2018-03-08;13:42:36;B;ARRANQUE - AUTO
2018-03-08;13:42:44;B;Paragem FIM de SEMANA ACTIVADO
2018-03-08;13:42:46;A;Paragem FIM de SEMANA ACTIVADO
```

***Ceramics department – Robot C***

aircondition robot C

date / time;temperature 1 line X [°C];humidity 1 line X [%];temperature 1 line X [°C];humidity 2 line X [%];temperature 1 line Z [°C];humidity 1 line Z [%];temperature 1 line Z [°C];humidity 2 line Z [%];temperature outside [°C];humidity outside [%];temperature inflow [°C];humidity inflow [%];temperature outflow [°C];humidity outflow [%];airflow inflow [m/s];airflow outflow [m/s];temperature drying [°C];humidity drying [%];

```
11.10.2018
00:00:01;24,2513;42,05006;25,08861;40,40075;25,21521;41,09158;24,88426;41,03371;19,2
5817;61,73322;22,24031;45,71759;24,78841;41,82581;7,351346;5,352647;25,50275;41,7173
11.10.2018
00:01:01;24,20971;43,67043;25,11755;40,45501;25,19531;41,06988;24,89873;40,98307;19,
30519;61,87066;22,3253;51,33464;24,83001;41,82581;6,798322;4,979745;25,57689;41,4930
5
11.10.2018
00:02:01;24,29832;42,7228;25,08861;41,08435;25,2351;41,04818;24,84447;41,57624;19,24
371;62,07682;22,41392;44,84592;24,84628;42,06814;5,764974;5,020255;25,6203;41,52922
11.10.2018
00:03:01;24,21875;42,2309;25,07414;40,3682;25,17542;41,0952;24,90234;41,1603;19,2853;
62,0081;21,7249;47,53689;24,84447;41,69922;7,754991;4,78588;25,57509;41,64135
11.10.2018
00:04:01;24,25673;43,05194;25,04702;40,70095;25,23691;41,02286;24,86437;41,14945;19,
29977;61,98278;22,04319;52,14844;24,8282;41,9307;8,135127;4,621672;25,54977;41,78241
11.10.2018
00:05:01;24,42311;41,88368;25,14106;41,10966;25,16819;41,04818;24,89873;41,68837;19,
31243;61,87428;22,24935;45,26186;24,8065;42,15857;8,899378;5,355903;25,50094;41,9053
8
11.10.2018
00:06:01;24,48821;41,1386;25,0416;40,37182;25,17542;41,10243;24,87883;41,3954;19,321
47;62,04427;21,95638;46,27459;24,7649;41,82581;9,225984;5,259693;25,53349;42,00304
11.10.2018
00:07:01;24,49544;42,27792;25,07957;40,56351;25,16095;41,10605;24,84628;41,182;19,25
998;62,12022;22,79188;49,83362;24,72692;42,09346;8,24689;5,300564;25,50275;42,01389
11.10.2018
00:08:02;24,6998;40,96499;25,10489;40,78415;25,18265;41,17115;24,85894;41,54369;19,2
9977;62,05512;23,03422;43,33044;24,74501;42,06453;8,601707;5,292969;25,45935;41,9741
11.10.2018
00:09:01;24,72511;40,75521;25,0633;40,20906;25,18808;41,17115;24,92766;41,15668;19,3
1966;61,88874;22,75571;47,22222;24,75043;41,76794;7,489511;5,003617;25,48828;41,9379
3
11.10.2018
00:10:01;24,7649;41,9741;25,07957;40,89265;25,18808;41,10243;24,86798;41,22902;19,31
424;61,6645;23,44293;47,58753;24,75405;42,12601;8,705512;4,76454;25,51179;41,84389
11.10.2018
00:11:03;24,83905;40,97584;25,12117;40,9252;25,22244;41,12775;24,89511;41,59071;19,3
1604;61,55599;23,23676;42,93258;24,77575;42,09346;8,377821;5,150825;25,50456;41,7390
1
11.10.2018
00:12:02;24,70703;41,43518;25,08138;40,40075;25,20797;41,20008;24,88245;41,33391;19,
31966;61,46195;22,04499;46,9618;24,77575;41,82581;7,713397;5,150825;25,5136;41,73177
```

```
11.10.2018
00:13:01;24,56055;41,85836;25,07957;40,3248;25,18989;41,25072;24,83001;41,18924;19,3
2509;61,58131;21,69415;48,65451;24,74501;41,76794;7,144097;5,343605;25,57328;41,6196
5
11.10.2018
00:14:01;24,48097;41,90538;25,07414;40,47671;25,15191;41,182;24,88426;41,1603;19,312
43;61,34621;21,65437;50,05063;24,69437;41,87645;8,849103;5,211589;25,50275;41,6522
11.10.2018
00:15:01;24,55874;41,69922;25,06149;40,6684;25,1718;41,18562;24,85352;41,22902;19,39
2;61,4294;21,69054;47,74305;24,67448;42,07176;7,917752;5,150825;25,50636;41,71369
11.10.2018
00:16:01;24,51715;41,20008;25,0416;40,15119;25,17904;41,06264;24,8083;40,99031;19,37
03;61,38599;21,43374;48,86791;24,66544;41,90538;7,460576;5,001447;25,45935;41,76432
11.10.2018
00:17:01;24,50268;41,76794;25,05787;40,34288;25,1284;41,17477;24,83724;41,06264;19,3
4498;61,50174;22,19329;49,13556;24,67448;42,03559;7,888817;5,648148;25,47924;41,8945
3
11.10.2018
00:18:02;24,62384;41,1169;25,08138;40,58883;25,15915;41,12775;24,85532;41,25796;19,4
0828;61,05324;22,95465;44,98698;24,68352;42,07176;8,000217;5,059318;25,46297;41,8004
9
11.10.2018
00:19:01;24,62384;40,98307;25,06149;40,34288;25,10127;41,04094;24,82458;41,15668;19,
40104;60,96644;23,01613;47,30903;24,71788;41,86198;7,124566;4,879196;25,46297;41,579
86
11.10.2018
00:20:01;24,81011;42,25622;25,08681;40,72266;25,20255;41,12775;24,84447;41,39902;19,
40647;61,13281;23,73409;48,77387;24,7432;42,15133;7,3159;4,896557;25,49009;41,37732
11.10.2018
00:21:01;24,81915;41,12413;25,03074;41,28328;25,19893;41,1169;24,86437;41,95602;19,4
1008;61,07856;23,51888;42,41175;24,7649;42,22367;7,919198;5,445602;25,51722;41,45327
11.10.2018
00:22:01;24,81735;40,97584;25,06149;40,89265;25,22063;41,20008;24,89149;41,46774;19,
43902;60,9375;22,46636;48,09751;24,7649;41,86198;8,368417;5,404007;25,47201;41,47135
11.10.2018
00:23:01;24,87883;42,21643;25;41,00477;25,22063;41,24349;24,85352;41,54369;19,37211;
60,87601;22,82624;49,78299;24,75405;42,11878;8,215061;5,465133;25,50094;41,57263
11.10.2018
00:24:01;24,88788;41,57624;24,94213;41,27243;25,18989;41,25072;24,90054;41,76432;19,
41008;60,73134;22,56402;45,84418;24,73597;42,26707;8,98112;5,404731;25,48286;41,5545
4
11.10.2018
00:25:01;24,75767;41,59433;24,99096;41,03371;25,16095;41,34838;24,8933;41,60518;19,4
3902;60,63729;21,90574;49,32002;24,71788;42,13325;8,765915;5,089699;25,50456;41,5183
8
11.10.2018
00:26:01;24,6781;42,52749;24,94756;41,11328;25,2351;41,34115;24,8933;41,49305;19,439
02;60,56134;21,99436;50,38339;24,67448;42,4009;8,327908;5,201823;25,49732;41,48582
11.10.2018
00:27:01;24,64193;41,54731;24,9783;41,14584;25,21521;41,40263;24,85532;41,61965;19,4
4264;60,3588;21,76468;47,10287;24,6365;42,34665;8,86972;4,947917;25,50456;41,41348
```

***Foundry department – Rotary oven 1***

O2 rotation furnase 1

```
Datum Uhrzeit;Sollwert Zone 1 [°C];Isttemperatur Zone 1 [°C];Offset Zone 1 [°C];Sollwert
Zone 2 [°C];Isttemperatur Zone 2 [°C];Offset Zone 2 [°C];Sollwert Zone 3 [°C];Isttemperatur
Zone 3 [°C];Offset Zone 3 [°C];Sollwert Zone 4 [°C];Isttemperatur Zone 4 [°C];Offset Zone 4
[°C];Sollwert Zone 5 [°C];Isttemperatur Zone 5 [°C];Offset Zone 5 [°C];O2 Sonde [%];
Data/Hora do dia;Alvo Zona 1 [°C];Real Zona 1 [°C];Offset Zone 1 [°C];Alvo Zona 2
[°C];Real Zona 2 [°C];Offset Zone 2 [°C];Alvo Zona 3 [°C];Real Zona 3 [°C];Offset Zone 3
[°C];Alvo Zona 4 [°C];Real Zona 4 [°C];Offset Zone 4 [°C];Alvo Zona 5 [°C];Real Zona 5
[°C];Offset Zone 5 [°C];O2 Sonda [%];

12.10.2018 00:00:01;600;690,3;-60;600;747,9;-13;600;699,8;0;600;712,9;-20;600;658;-
50;12,60872
12.10.2018 00:01:01;600;689,9;-60;600;747,5;-13;600;699,5;0;600;712,6;-20;600;657,6;-
50;12,55651
12.10.2018 00:02:01;600;689,5;-60;600;747,2;-13;600;699,2;0;600;712;-20;600;657,1;-
50;12,50496
12.10.2018 00:03:01;600;689,1;-60;600;746,8;-13;600;698,6;0;600;711,5;-20;600;656,6;-
50;12,55736
12.10.2018 00:04:01;600;688,7;-60;600;746,3;-13;600;698;0;600;711;-20;600;656,1;-
50;12,57716
12.10.2018 00:05:01;600;688,2;-60;600;745,9;-13;600;697,6;0;600;710,5;-20;600;655,7;-
50;12,47723
12.10.2018 00:06:01;600;687,8;-60;600;745,6;-13;600;696,9;0;600;709,8;-20;600;655,2;-
50;12,4999
12.10.2018 00:07:01;600;687,4;-60;600;745,2;-13;600;696,7;0;600;709,4;-20;600;654,6;-
50;12,47517
12.10.2018 00:08:01;600;687;-60;600;744,8;-13;600;696,2;0;600;709,1;-20;600;654,2;-
50;12,40832
12.10.2018 00:09:01;600;686,7;-60;600;744,3;-13;600;695,5;0;600;708,8;-20;600;653,8;-
50;12,31719
12.10.2018 00:10:01;600;686,2;-60;600;743,8;-13;600;695,4;0;600;708,5;-20;600;653,3;-
50;12,43368
12.10.2018 00:11:01;600;685,9;-60;600;743,6;-13;600;694,9;0;600;707,9;-20;600;652,9;-
50;12,36215
12.10.2018 00:12:01;600;685,3;-60;600;743;-13;600;694,1;0;600;707,4;-20;600;652,3;-
50;12,30453
12.10.2018 00:13:01;600;684,8;-60;600;742,4;-13;600;693,3;0;600;706,8;-20;600;651,6;-
50;12,40871
12.10.2018 00:14:01;600;684,4;-60;600;742;-13;600;692,9;0;600;706,3;-20;600;651,2;-
50;12,38341
12.10.2018 00:15:01;600;684;-60;600;741,5;-13;600;692,8;0;600;705,9;-20;600;650,7;-
50;12,33103
12.10.2018 00:16:01;600;683,6;-60;600;741,1;-13;600;692,1;0;600;705,7;-20;600;650,2;-
50;12,23083
12.10.2018 00:17:01;600;683,2;-60;600;740,7;-13;600;691,5;0;600;705,4;-20;600;649,8;-
50;12,30402
12.10.2018 00:18:01;600;682,8;-60;600;740,3;-13;600;690,9;0;600;705;-20;600;649,3;-
50;12,2459
12.10.2018 00:19:01;600;682,4;-60;600;739,9;-13;600;690,7;0;600;704,5;-20;600;648,9;-
50;12,24299
```

***Foundry department – Rotary oven 2***

O2 rotation furnase 1

```
Datum Uhrzeit;Sollwert Zone 1 [°C];Isttemperatur Zone 1 [°C];Offset Zone 1 [°C];Sollwert
Zone 2 [°C];Isttemperatur Zone 2 [°C];Offset Zone 2 [°C];Sollwert Zone 3 [°C];Isttemperatur
Zone 3 [°C];Offset Zone 3 [°C];Sollwert Zone 4 [°C];Isttemperatur Zone 4 [°C];Offset Zone 4
[°C];Sollwert Zone 5 [°C];Isttemperatur Zone 5 [°C];Offset Zone 5 [°C];O2 Sonde [%];
Data/Hora do dia;Alvo Zona 1 [°C];Real Zona 1 [°C];Offset Zone 1 [°C];Alvo Zona 2
[°C];Real Zona 2 [°C];Offset Zone 2 [°C];Alvo Zona 3 [°C];Real Zona 3 [°C];Offset Zone 3
[°C];Alvo Zona 4 [°C];Real Zona 4 [°C];Offset Zone 4 [°C];Alvo Zona 5 [°C];Real Zona 5
[°C];Offset Zone 5 [°C];O2 Sonda [%];

23.05.2018 00:00:01;1000;616,8;-50;1000;657,8;-17;1130;675,6;-40;1130;674,5;-
49,5;1130;625,4;-76;20,3138
23.05.2018 00:01:01;1000;632,3;-50;1000;659,4;-17;1130;673,7;-40;1130;672,4;-
49,5;1130;616,2;-76;20,3138
23.05.2018 00:02:01;1000;616,6;-50;1000;655;-17;1130;672,7;-40;1130;671,6;-
49,5;1130;626,5;-76;20,3138
23.05.2018 00:03:01;1000;629,7;-50;1000;657;-17;1130;672,2;-40;1130;671,5;-
49,5;1130;628,4;-76;20,3138
23.05.2018 00:04:01;1000;622,9;-50;1000;654,4;-17;1130;671;-40;1130;669,7;-
49,5;1130;615,8;-76;20,3138
23.05.2018 00:05:01;1000;626,6;-50;1000;654,5;-17;1130;670,1;-40;1130;668,7;-
49,5;1130;622,6;-76;20,3138
23.05.2018 00:06:01;1000;630,5;-50;1000;653,8;-17;1130;669,4;-40;1130;668,9;-
49,5;1130;627,7;-76;20,3138
23.05.2018 00:07:01;1000;622;-50;1000;651,4;-17;1130;667,4;-40;1130;667,3;-
49,5;1130;615,9;-76;20,3138
23.05.2018 00:08:01;1000;633,5;-50;1000;652,3;-17;1130;667,2;-40;1130;666,4;-
49,5;1130;625;-76;20,3138
23.05.2018 00:09:01;1000;618,8;-50;1000;649,2;-17;1130;665,9;-40;1130;665,7;-
49,5;1130;619,7;-76;20,3138
23.05.2018 00:10:01;1000;632,8;-50;1000;651,3;-17;1130;663,5;-40;1130;663,3;-
49,5;1130;609;-76;20,3138
23.05.2018 00:11:01;1000;615,8;-50;1000;647,1;-17;1130;664,3;-40;1130;664,6;-
49,5;1130;628,6;-76;20,3138
23.05.2018 00:12:01;1000;629,6;-50;1000;648;-17;1130;663;-40;1130;663;-
49,5;1130;618,9;-76;20,3138
23.05.2018 00:13:01;1000;623,8;-50;1000;645,1;-17;1130;660,3;-40;1130;661,4;-
49,5;1130;626,3;-76;20,3138
23.05.2018 00:14:01;1000;624,7;-50;1000;646,3;-17;1130;659,3;-40;1130;661,1;-
49,5;1130;629,7;-76;20,3138
23.05.2018 00:15:01;1000;632,3;-50;1000;646,8;-17;1130;659,4;-40;1130;659,7;-
49,5;1130;616,2;-76;20,3138
23.05.2018 00:16:01;1000;619;-50;1000;644,2;-17;1130;657,8;-40;1130;658,8;-
49,5;1130;624,1;-76;20,3138
23.05.2018 00:17:01;1000;632,6;-50;1000;644,8;-17;1130;658,3;-40;1130;659,2;-
49,5;1130;627,4;-76;20,3138
23.05.2018 00:18:01;1000;616,2;-50;1000;642,2;-17;1130;657,1;-40;1130;657,5;-
49,5;1130;612;-76;20,3138
23.05.2018 00:19:01;1000;628,5;-50;1000;642,7;-17;1130;654,3;-40;1130;656,7;-
49,5;1130;626,9;-76;20,3138
```

***Foundry department – Rotary oven (vacuum sector)***

Temperature IOB WA6370

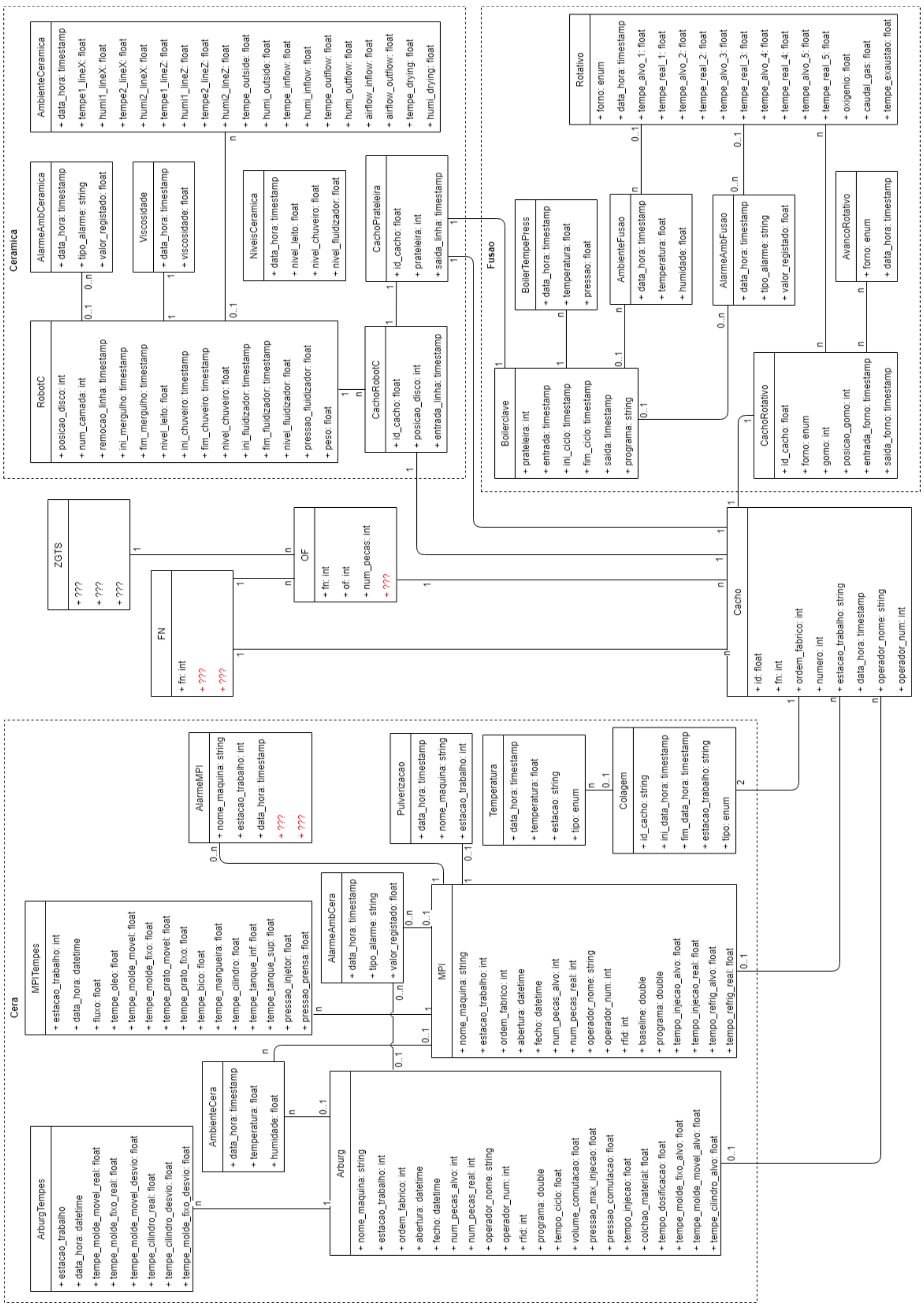
Datum Uhrzeit;Sollwert Zone 1 [°C];Isttemperatur Zone 1 [°C];Sollwert Zone 2 [°C];Isttemperatur Zone 2 [°C];Sollwert Zone 3 [°C];Isttemperatur Zone 3 [°C];Sollwert Zone TNV [°C];Isttemperatur Zone TNV [°C];O2 Zone 1 [%];O2 Zone 3 [%];
Datum Uhrzeit;Alvo Zona 1 [°C];Real Zona 1 [°C];Alvo Zona 2 [°C];Real Zona 2 [°C];Alvo Zona 3 [°C];Real Zona 3 [°C];Alvo Zona TNV [°C];Real Zona TNV [°C];O2 Zona 1 [%];O2 Zona 3 [%];

```
17.01.2018 00:00:01;650;652,1;650;664,7;650;664,7;650;670,3;0;4,193689
17.01.2018 00:01:02;650;651,4;650;663,8;650;663,8;650;652,9;20,31379;20,31379
17.01.2018 00:02:01;650;651,4;650;663,7;650;663,7;650;632,3;20,31379;20,31379
17.01.2018 00:03:01;650;651,4;650;663,7;650;663,7;650;611,2;13,57171;20,13318
17.01.2018 00:04:01;650;651,5;650;663,7;650;663,7;650;638,8;12,00866;17,22153
17.01.2018 00:05:01;650;651,7;650;663,7;650;663,7;650;677;11,73824;16,33927
17.01.2018 00:06:01;650;651,6;650;663,6;650;663,6;650;670,5;11,78397;16,11622
17.01.2018 00:07:01;650;651,4;650;663,5;650;663,5;650;653,6;11,93123;16,14127
17.01.2018 00:08:02;650;651,3;650;663,5;650;663,5;650;631,6;12,04952;16,13608
17.01.2018 00:09:02;650;651,3;650;663,4;650;663,4;650;611,3;12,15669;16,27245
17.01.2018 00:10:01;650;651,1;650;663,3;650;663,3;650;642,9;12,2472;16,36465
17.01.2018 00:11:01;650;651,1;650;663,2;650;663,2;650;676,5;12,31618;16,38985
17.01.2018 00:12:02;650;651;650;663;650;663;650;666,8;12,396;16,34285
17.01.2018 00:13:01;650;650,8;650;662,9;650;662,9;650;653,5;12,48194;16,39247
17.01.2018 00:14:01;650;650,6;650;662,7;650;662,7;650;633,4;12,52737;16,48036
17.01.2018 00:15:01;650;650,5;650;662,6;650;662,6;650;612,9;12,59454;16,49396
17.01.2018 00:16:02;650;650,3;650;662,4;650;662,4;650;641,6;12,67904;16,50344
17.01.2018 00:17:01;650;653,1;650;663,5;650;663,5;650;678,1;9,809351;14,29685
17.01.2018 00:18:01;650;653,6;650;663,5;650;663,5;650;670,7;11,40149;14,79188
17.01.2018 00:19:01;650;653,2;650;663,2;650;663,2;650;653;11,88024;15,30934
17.01.2018 00:20:01;650;652,7;650;662,7;650;662,7;650;631,5;12,35873;15,77209
17.01.2018 00:21:01;650;652,2;650;662,4;650;662,4;650;611,2;12,6672;16,12262
17.01.2018 00:22:01;650;651,7;650;662;650;662;650;647,5;12,85119;16,41908
17.01.2018 00:23:02;650;651,2;650;661,6;650;661,6;650;677,6;13,00587;16,57324
17.01.2018 00:24:02;650;650,7;650;661,3;650;661,3;650;666,6;13,12477;16,68997
17.01.2018 00:25:01;650;650,5;650;661;650;661;650;649,8;13,22922;16,87796
17.01.2018 00:26:01;650;653,9;650;662,7;650;662,7;650;631,2;10,71326;14,89133
17.01.2018 00:27:01;650;653,6;650;662,6;650;662,6;650;612,2;11,82944;15,43751
17.01.2018 00:28:01;650;652,9;650;662,1;650;662,1;650;649,1;12,379;15,97555
17.01.2018 00:29:01;650;652,2;650;661,6;650;661,6;650;679,4;12,6945;16,36234
17.01.2018 00:30:01;650;651,5;650;661,2;650;661,2;650;667,7;12,90914;16,50344
17.01.2018 00:31:03;650;650,9;650;660,7;650;660,7;650;666,6;13,10519;16,72583
17.01.2018 00:32:01;650;650,5;650;660,4;650;660,4;650;663,2;13,18167;16,82195
17.01.2018 00:33:01;650;653,1;650;661,4;650;661,4;650;645,6;10,48669;14,67687
17.01.2018 00:34:02;650;653,3;650;661,4;650;661,4;650;624,8;11,81187;15,21729
17.01.2018 00:35:02;650;652,7;650;661;650;661;650;607,9;12,35348;15,74963
17.01.2018 00:36:01;650;652,2;650;660,6;650;660,6;650;654,1;12,71713;16,18066
17.01.2018 00:37:02;650;651,4;650;660,2;650;660,2;650;679;12,96716;16,38769
17.01.2018 00:38:02;650;650,9;650;659,9;650;659,9;650;665,2;13,20132;16,59417
17.01.2018 00:39:01;650;650,4;650;659,5;650;659,5;650;647,4;13,29992;16,71782
17.01.2018 00:40:01;650;653,6;650;661,3;650;661,3;650;628,3;10,74542;14,67803
17.01.2018 00:41:01;650;653,6;650;661,1;650;661,1;650;614,2;11,98405;15,30165
```

***Wax injection department – MPI55***

```
DATE, 10/25/2018
TEMPO, 09:52:35
RECEITA, 4,15001
BASELINE, 4,15001
TEMPO, FLUXO, PRESSÃO, TEMPERATURE
0.0, 1.0, 0.0, 53.7
0.1, 56.0, 0.1, 53.7
0.2, 25.0, 2.7, 53.7
0.3, 31.0, 7.2, 53.7
0.4, 34.0, 10.9, 53.7
0.5, 34.0, 14.3, 53.7
0.6, 28.0, 16.2, 53.7
0.7, 19.0, 17.3, 53.7
0.8, 13.0, 17.7, 53.7
0.9, 9.0, 17.8, 53.7
1.0, 8.0, 18.0, 53.7
1.1, 5.0, 18.1, 53.7
1.2, 5.0, 18.1, 53.7
1.3, 5.0, 18.1, 53.7
1.4, 4.0, 18.0, 53.7
1.5, 4.0, 18.1, 53.7
1.6, 3.0, 18.1, 53.7
1.7, 3.0, 18.0, 53.7
1.8, 4.0, 18.1, 53.7
1.9, 2.0, 18.2, 53.7
2.0, 3.0, 18.1, 53.7
2.1, 2.0, 18.1, 53.7
2.2, 2.0, 18.1, 53.7
2.3, 2.0, 18.2, 53.7
2.4, 3.0, 18.2, 53.7
2.5, 3.0, 18.2, 53.7
2.6, 2.0, 18.2, 53.7
2.7, 2.0, 18.1, 53.7
2.8, 2.0, 18.2, 53.7
2.9, 2.0, 18.2, 53.7
3.0, 3.0, 18.2, 53.7
3.1, 2.0, 18.2, 53.7
3.2, 1.0, 18.2, 53.7
3.3, 1.0, 18.2, 53.7
3.4, 0.0, 18.2, 53.7
3.5, 1.0, 18.2, 53.7
3.6, 0.0, 18.2, 53.7
3.7, 0.0, 18.2, 53.7
3.8, 0.0, 18.2, 53.7
3.9, 0.0, 18.3, 53.7
4.0, 0.0, 18.3, 53.7
4.1, 1.0, 18.2, 53.7
4.2, 0.0, 18.2, 53.7
4.3, 0.0, 18.2, 53.7
4.4, 1.0, 18.2, 53.7
```

## ATTACHMENT C: Database scheme


Ceramica
AmbienteCeramica
+ data_hora: timestamp
+ tempe1_lineX: float
+ humi1_lineX: float
+ tempe2_lineX: float
+ humi2_lineX: float
+ tempe1_lineZ: float
+ humi1_lineZ: float
+ tempe2_lineZ: float
+ humi2_lineZ: float
+ tempe_outside: float
+ humi_outside: float
+ tempe_inflow: float
+ humi_inflow: float
+ tempe_outflow: float
+ humi_outflow: float
+ airflow_inflow: float
+ airflow_outflow: float
+ tempe_drying: float
+ humi_drying: float
AlarmeAmbCeramica
+ data_hora: timestamp
+ tipo_alarme: string
+ valor_registado: float
Viscosidade
+ data_hora: timestamp
+ viscosidade: float
NiveisCeramica
+ data_hora: timestamp
+ nivel_leito: float
+ nivel_chuveiro: float
+ nivel_fluidizador: float
CachoPrateleira
+ id_cacho: float
+ prateleira: int
+ saida_linha: timestamp
RobotC
+ posicao_disco: int
+ num_camada: int
+ remocao_linha: timestamp
+ ini_mergulho: timestamp
+ fim_mergulho: timestamp
+ nivel_leito: float
+ ini_chuveiro: timestamp
+ fim_chuveiro: timestamp
+ nivel_chuveiro: float
+ ini_fluidizador: timestamp
+ fim_fluidizador: timestamp
+ nivel_fluidizador: float
+ pressao_fluidizador: float
+ peso: float
CachoRobotC
+ id_cacho: float
+ posicao_disco: int
+ entrada_linha: timestamp
Fusao
Rotativo
+ forno: enum
+ data_hora: timestamp
+ tempe_alvo_1: float
+ tempe_real_1: float
+ tempe_alvo_2: float
+ tempe_real_2: float
+ tempe_alvo_3: float
+ tempe_real_3: float
+ tempe_alvo_4: float
+ tempe_real_4: float
+ tempe_alvo_5: float
+ tempe_real_5: float
+ oxigenio: float
+ caudal_gas: float
+ tempe_exaustao: float
BoilerTempePress
+ data_hora: timestamp
+ temperatura: float
+ pressao: float
AmbienteFusao
+ data_hora: timestamp
+ temperatura: float
+ humidade: float
AlarmeAmbFusao
+ data_hora: timestamp
+ tipo_alarme: string
+ valor_registado: float
AvancoRotativo
+ forno: enum
+ data_hora: timestamp
Boilerclave
+ prateleira: int
+ entrada: timestamp
+ ini_ciclo: timestamp
+ fim_ciclo: timestamp
+ saida: timestamp
+ programa: string
CachoRotativo
+ id_cacho: float
+ forno: enum
+ gomo: int
+ posicao_gomo: int
+ entrada_forno: timestamp
+ saida_forno: timestamp
ZGTS
+ ???
+ ???
+ ???
OF
+ fn: int
+ of: int
+ num_pecas: int
+ ???
FN
+ fn: int
+ ???
+ ???
Cacho
+ id: float
+ fn: int
+ ordem_fabrico: int
+ numero: int
+ estacao_trabalho: string
+ data_hora: timestamp
+ operador_nome: string
+ operador_num: int
Cera
AlarmeMPI
+ nome_maquina: string
+ estacao_trabalho: int
+ data_hora: timestamp
+ ???
+ ???
Pulverizacao
+ data_hora: timestamp
+ nome_maquina: string
+ estacao_trabalho: int
Temperatura
+ data_hora: timestamp
+ temperatura: float
+ estacao: string
+ tipo: enum
Colagem
+ id_cacho: string
+ ini_data_hora: timestamp
+ fim_data_hora: timestamp
+ estacao_trabalho: string
+ tipo: enum
MPITempes
+ estacao_trabalho: int
+ data_hora: datetime
+ fluxo: float
+ tempe_oleo: float
+ tempe_molde_movel: float
+ tempe_molde_fixo: float
+ tempe_prato_movel: float
+ tempe_prato_fixo: float
+ tempe_bico: float
+ tempe_mangueira: float
+ tempe_cilindro: float
+ tempe_tanque_inf: float
+ tempe_tanque_sup: float
+ pressao_injetor: float
+ pressao_prensa: float
AlarmeAmbCera
+ data_hora: timestamp
+ tipo_alarme: string
+ valor_registado: float
MPI
+ nome_maquina: string
+ estacao_trabalho: int
+ ordem_fabrico: int
+ abertura: datetime
+ fecho: datetime
+ num_pecas_alvo: int
+ num_pecas_real: int
+ operador_nome: string
+ operador_num: int
+ rfid: int
+ baseline: double
+ programa: double
+ tempo_injecao_alvo: float
+ tempo_injecao_real: float
+ tempe_refrig_alvo: float
+ tempe_refrig_real: float
AmbienteCera
+ data_hora: timestamp
+ temperatura: float
+ humidade: float
ArburgTempes
+ estacao_trabalho
+ data_hora: datetime
+ tempe_molde_movel_real: float
+ tempe_molde_fixo_real: float
+ tempe_molde_movel_desvio: float
+ tempe_cilindro_real: float
+ tempe_cilindro_desvio: float
+ tempe_molde_fixo_desvio: float
Arburg
+ nome_maquina: string
+ estacao_trabalho: int
+ ordem_fabrico: int
+ abertura: datetime
+ fecho: datetime
+ num_pecas_alvo: int
+ num_pecas_real: int
+ operador_nome: string
+ operador_num: int
+ rfid: int
+ programa: double
+ tempo_ciclo: float
+ volume_comutacao: float
+ pressao_max_injecao: float
+ pressao_comutacao: float
+ tempo_injecao: float
+ colchao_material: float
+ tempo_dosificacao: float
+ tempe_molde_fixo_alvo: float
+ tempe_molde_movel_alvo: float
+ tempe_cilindro_alvo: float