U.PORTO

FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

Information Systems - SINF 2020/2021

PL PROJET OVERVIEW

Application scenario and main goals

Gil Gonçalves - gil@fe.up.pt
Luis Neto - lcneto@fe.up.pt
João Reis - jpcreis@fe.up.pt
Vítor Pinto - vitorpinto@fe.up.pt

1. Introduction

Industry 4.0 is the digital transformation of manufacturing/production and represents a new stage in the organization and control of the industrial value chain.

Cyber-physical systems form the basis of Industry 4.0 (e.g., 'smart machines'). They use modern control systems, have embedded software systems and dispose of an Internet address to connect and be addressed via IoT (the Internet of Things). This way, means of production (i.e. equipment) and products get networked and can 'communicate', enabling new ways of production and real-time optimization. Cyber-physical systems create the capabilities needed for smart factories. These are the same capabilities we know from the Industrial Internet of Things, like remote monitoring or track and trace, just to mention two.

According to the German Plattform Industrie 4.0, "Industry 4.0 refers to the intelligent networking of machines and processes for industry with the help of information and communication technology".

Figure 1: Intelligent networking of machines and processes for industry (Industry 4.0)

Among the huge variety of tools and innovations that are part of Industry 4.0, the extensive connectivity, the generation of huge amounts of data, cloud computing or process simulation stand out for their importance in new industrial processes. In this sense, the evolution that Cyber-physical systems will enable has a very solid foundation in process optimization. However, this cannot occur without considering the monitoring of energy consumption and the implementation of sustainable systems and processes.

Nowadays, energy efficiency in manufacturing is one of the most discussed and relevant subjects and the goal of this project is to develop a **G**reen **Man**ufacturing information system (GMAN) that addresses the issue of efficient energy in an Industry 4.0 scenario.

2. Application scenario

The proposed application scenario for the GMAN is the automated pizza production line of Zume Pizza[1] in Figure 2[2]. Zume Pizza uses 4 robots at 3 steps of its production chain (pouring tomato, spreading the sauce, and putting pizzas in the oven). The production line integrates different sensors (current, temperature, humidity and light) and actuators (controlling lights (on/off) and heating (on/off)) installed in the six work cells (1 through 6 in the figure). Sensors are also used to measure the energy consumption of certain equipment (like the oven, the robots or the conveyors). Additional sensors can be installed and configured in the **GMAN** to cover the remaining sections of the production line and other equipment. **GMAN**'s energy management is based on the principles of Industry 4.0 and the Internet of Things (IoT) and various components, both hardware and software. Includes edge devices which can be installed directly on the equipment to collect energy consumption data and to promote long-term efficiency. In addition, allowing to monitor the correct operation of the production line, identifying any consumption or waste peak.

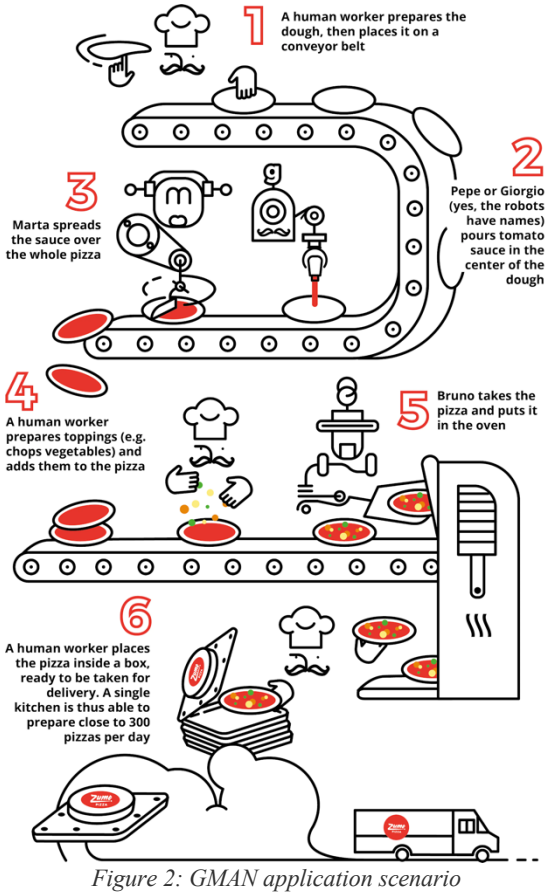

Figure 2: GMAN application scenario

[1] Zume, the *pizzas-made-by-robots-and-cooked-during-delivery startup*, was a normal pizza delivery service operating in Silicon Valley: you pick your order by phone, through their website or their apps, you pay and you get delivered.
[2] https://www.fabernovel.com/en/article/tech-en/when-pizzas-meet-ai

This information, along with other important data such as temperature or light, is then transmitted and historicized through a database. Then an intuitive software provides access to the data, for remote monitoring and analysis of the production lines.

The **GMAN should be scalable and configurable for production lines with different layouts and with different number of cells**. It could also be used to manage industrial units composed of multiple production lines. **GMAN** uses information collected by the sensors to apply rules that control the lighting and heating of the production line. There are general rules for the line (e.g. if total power consumption > 10 kW then turn of some equipment; if temperature < 23ºC then turn heating ON), rules for a specific cell (these override the line rules), rules for specific equipment (these override the cell rules) and the possibility for the user to manual control of the actuators (manual control overrides rules).

GMAN needs to be configured with information of the production line layout (number of cells) and location of sensors and actuators. **Sensors and actuators can be added and removed dynamically**. These operations are done by a user with specific privileges.

Web consoles available for the production line or cell are able to display energy consumption, temperature, light and humidity measurements, as well as the state of the actuators. These consoles also provide manual control of the actuators. Some examples of sensors and actuators can be found in Figure 3[3].

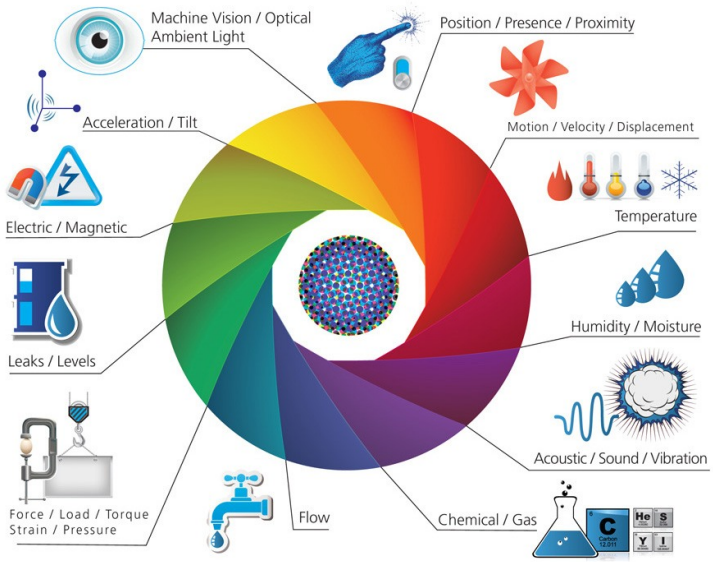

Figure 3: Examples of IOT sensors/actuators[2]

[3]https://www.electronicproducts.com/wp-content/uploads/internet-of-things-sensors-sensors-1-the-range-of-industrial-sensors.jpg

3. Main goals

The Main goal of the project is to implement an information system (the GMAN) based on a multitier architecture, represented in Figure 4:

1. real time data collection and control (edge layer);
2. data storage and management (cloud layer); and
3. data presentation and user interface (user interface layer).

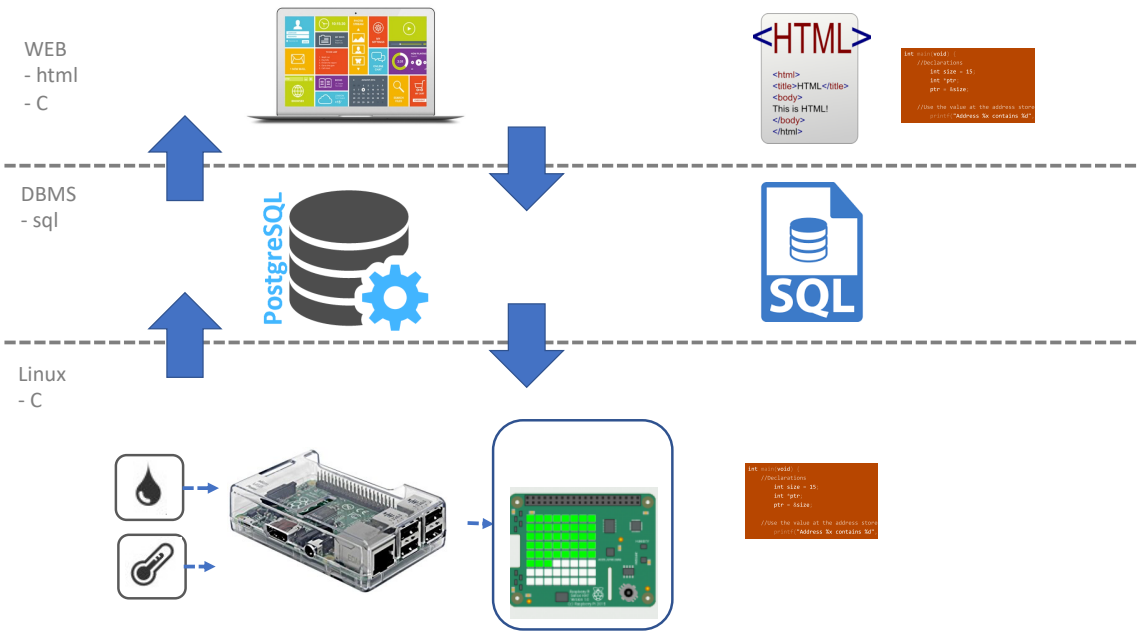

Figure 4: GMAN architecture

The main characteristics of each layer are described next.

Real time data collection and control (edge layer)

This layer is implemented on a Raspberry PI 3, which gathers the data collected by the sensors, processes the collected information and controls the lighting and heating, using actuators, based on a set of defined rules. The features for the edge layer are:

- Information gathering from the sensor networks.
- Control of the actuator networks.
- Implementation of control rules.
- State update of the actuators (on/off).

Data storage and management (cloud layer)

The data layer is implemented on a Relational Database Management Systems (PostgreSQL) running on a server. The application running on the Raspberry PI 3 connects to the database to store the data collected by the sensor networks and to get information on the control rules. The features of the cloud layer are:

- Store the production line layout information.
- Store the sensor and actuator specification (including location).
- Store information about the control rules.
- Storage the collected sensor data over time.

Data presentation and user interface

The presentation layer implements a set of web consoles (web pages) that display information and allow users to manually control the actuators. The presentation layer uses a web server and connects to the database to get data. The main features of the user interface layer are:

- Information display regarding the equipment, cell and production line.
- Allows manual control of the actuators.

4. Development process

The development of the **GMAN** will follow a Scrum based iterative process, as shown in Figure 5, organised in 3 Sprints (iterations) with well-defined objectives. The LP teacher will have the role of Product Owner (PO), while the team of students will take the role of Development team.

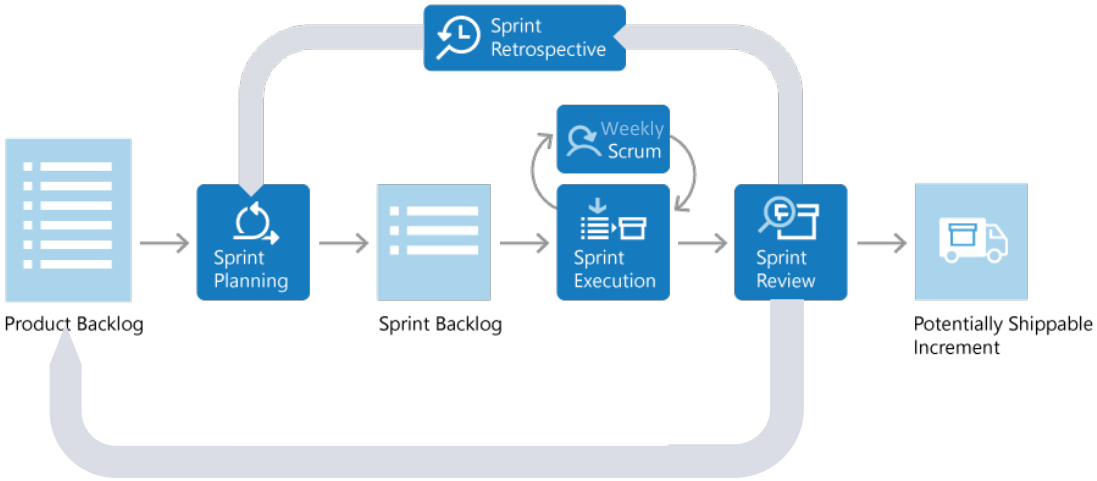

Figure 5: proposed GMAN development process

Sprint#1 will focus on the definition of the **GMAN** concept and its architecture, as well as the implementation of the software that will run in the edge device.

Sprint#2 will focus on the definition of the data model and the implementation of the **GMAN** database.

Sprint#3 will focus on the definition of the user interaction and implementation of the **GMAN** consoles.

The concrete goals will be provided, along with detailed specifications, at the beginning of each Sprint.

5. Development of the concept

The project starts with the development of the **product vision** and the creation of the **product backlog**. The product vision provides information regarding the product features, target group, product benefits, needs to be solved and goals for the product. The vision statement must articulate all this information in a concise matter, since it can be used to pitch the product to stakeholders and end users. The product owner is responsible for knowing about the product, its goals, and its requirements and, with the help of the development team, takes responsibility for creating the vision statement. The methodology for the development of the product vision is organised in 5 steps (see slide deck from T1):

1. Develop the product objective (team).
2. Create a draft vision statement (team).

3. Validate and revise the vision statement (team and PO).
4. Finalize your vision statement (team).
5. Write the vision statement in the README file of the GitHub project repository.

The product vision statement can be created using the following template. This statement is supported by the information identified in the categories of *Target Group*, *Needs*, *Product* and *Goals*.

Target Group	**Needs**	**Product**	**Goals**
Who are the target customers and users?	*What problem does the product solve?* *Why does the customer needs the product?*	*What product is it?* *What is the main features of the product?*	*How will the product benefit the company creating it?*
Vision Statement "**For** *(target customer)* **who** *(statement of need or opportunity)* **the** *(product name)* **is a** *(product category)* **that** *(key benefit, reason to buy/use)*."			

Once the product vision is done and approved by the PO, the team is ready to create the product backlog. The product backlog is an ordered list of everything that is known to be needed in the product and single source of features, functions and requirements for any changes to be made to the product. To help creating the product backlog, first the team develops the **UML Use Case diagram** of the product. This model will show the purpose/utility of the product, specifying the context and capturing the functional requirements. The methodology to develop the Use Case diagram is organised in 4 steps:

1. Identify actors in the system, or user profiles and other systems that interact with the system;
2. Identify, for each actor, their major use cases (there may be cases involving the participation of more than one actor);
3. Structuring use cases in order to promote reuse, extensibility and modularity;
4. Specify textually each use case according to a predefined template, which is shown in Figure 6.

Section	Content
Indetifier	
Name	
Author	
Version	
Change history	
Priority	
Criticality	
Source	
Short description	
Goal(s)	
Primary actor	
Other actors	
Precondition	
Postcondition	
Result	
Main scenario	
Alternative Scenario(s)	
Exception scenario(s)	

Figure 6: Use case documentation template

The use case diagram should be built using *draw.io* and saved in the team *GitHub* project repository, along with the use cases descriptions. The corresponding product backlog (i.e. the list of user stories) should be created in *GitHub*.

Use cases describe a sequence of interactions between a system and an external actor that results in the actor being able to achieve some outcome: use cases are written in the format "`<verb> <object>`" (e.g. "check in for a flight"). **User stories** (used in agile projects!) are short, simple description of a feature told from the perspective of the person who desires the new capability, usually a user or a customer of the system: user stories are written in the format "As a `<type of user>` I want `<some goal>` so that `<some reason>`" (e.g. "as a traveller, I want to check in for a flight so that I can fly to my destination").

6. Wrap up and goals for Sprint#1

The result of work described is the "starting" product backlog. This backlog will contain a set of high-level user stories that will be refined with the information provided by the PO in the following weeks.

In the first sprint we will focus on the general system concept and on the development of the real time data collection and control application. The goals for sprint#1 are the following:

1. Develop the product vision and the system concept (UML use case diagram).
2. Develop an application for real time data collection and control.

The second goal will be detailed further.