MEDIS: A Methodology for the Formation of Highly Qualified Engineers at Masters Level in the Design and Development of Advanced Industrial Informatics Systems

WP2.1 Chapter 3.2: Project documentation

Co-funded by the
Tempus Programme
of the European Union

544490-TEMPUS-1-2013-1-ES-TEMPUS-JPCR

Authors: Busquets, J.V., Albaladejo, J.

MEDIS: A Methodology for the Formation of Highly Qualified Engineers at Masters Level in the Design and Development of Advanced Industrial Informatics Systems

WP2.1 Chapter 3.2: Project documentation

Contract Number: *544490-TEMPUS-1-2013-1-ES-TEMPUS-JPCR*

Starting date: 01/12/2013 **Ending date:** 30/11/2016

Deliverable Number: 2.1

Title of the Deliverable: AIISM teaching resources - Industrial Computers

Task/WP related to the Deliverable: Development of the AIISM teaching resources - Industrial Computers

Type (Internal or Restricted or Public): Public

Author(s): Busquets, J.V., Albaladejo, J.

Contractual Date of Delivery to the CEC: 30/09/2014

Actual Date of Delivery to the CEC: 30/09/2014

Project Co-ordinator

Company name :	Universitat Politecnica de Valencia (UPV)
Name of representative :	Houcine Hassan
Address :	Camino de Vera, s/n. 46022-Valencia (Spain)
Phone number :	+34 96 387 7578
Fax number :	+34 963877579
E-mail :	husein@upv.es
Project WEB site address :	https://www.medis-tempus.eu

Context

WP 2	Design of the AIISM-PBL methodology
WPLeader	Universitat Politècnica deValència (UPV)
Task 2.1	Development of the AIISM teaching resources - Industrial Computers
Task Leader	UPV
Dependencies	MDU, TUSofia, USTUTT, UP

Author(s)	Busquets, J.V., Albaladejo, J.
Reviewers	Hassan, H., Martínez, J.M., Domínguez, C.

History

Version	Date	Author	Comments
0.1	01/03/2014	UPV Team	Initial draft
1.0	19/09/2014	UPV Team	Final version

Table of Contents

1 Executive summary

WP 2.1 details the learning materials of the Advanced Industrial Informatics Specialization Modules (AIISM) related to the Industrial Computers Module.

The contents of this package follows the guidelines presented in the UPV's documentation of the WP 1 (Industrial Computers Module)

- The PBL methodology was presented in WP 1.1
- The list of the module's chapters and the temporal scheduling in WP 1.2
- The required human and material resources in WP 1.3
- The evaluation in WP 1.4

During the development of this WP a separate document has been created for each of the chapters of the Industrial Computers Module (list of chapters in WP1.2).

In each of these documents, section 2 introduces the chapter; sections 3, 4, 5 and 6 details the Lecture, Laboratory, Seminar and Mini-project of the chapter; section 7 lists the bibliography and the references.

2 Introduction

This chapter introduces the documentation of the project. The objective is to give the student the guidelines to write the documentation of the project.

The practice proposes the writing of a sample document

The miniproject proposes the student to complete the documentation of the project

3 Lecture

Any engineering project requires a number of previous steps prior to the implementation: requirements statement, analysis and design. A good analysis and design is necessary to complete a good project. A bad or no design, will deliver a poor project. Each step in this process must be adequately documented. All the design cycle (analysis, design, implementation and test) must have a related documentation, each change and improvement must be reflected in these reports.

3.1 Documentation

Each document has a determined notation depending on the author and the audience. The documents can be classified as:

- Administrative project management, contracts, costs, etc.
    - Economic data.
    - Delivery.

    - Resources that should be available to facilitate development.
    - Duties and compensation.
    - Mechanism for modification in case of unforeseen incidents.
    - Acquisition of information and knowledge from the client to the designer.
    - Features of the process:
        - normal operating modes.
        - exceptional modes of operation.
    - Characteristics of the environment.
    - User needs.
- Document requirements.
    - Functionality:
        - of the various prototypes.
        - of the complete system.
    - Required Features and admitted limitations.
    - Definition of tests for acceptance.
- Specification of software and hardware.
    - General derived from customer requirements.
    - In each of the modules, set by the designers.
- Test results.
    - Intermediate tests for each prototype.
    - Final tests of the entire project
- Planning of the monitoring and maintenance.
- User manuals.
- Installation of the system, both software and hardware.
- Operation.

The documents relating to management are in possession of the customer and the project coordinator, and its notation is in natural language. The acquisition of information and knowledge is documented in a notation that evolves from casual in the early stages to technical and structured at the end. The latter is usually facilitated by an expert in the process, and the users of the final system, and it is usually collected by the management of the designer’s team.

The specifications of software and hardware are usually handled by the designers, and serve to define the activities to be performed. Their notation, at least, it is structured.

The test results should be expressed in formal notation, to allow evaluation, and thus, accept or reject the prototype or system.

Finally, the documentation provided to the user is often informal structured. It will allow the operation and maintenance of the system: user manual, technical manual, etc.

3.2 Planning

Let's consider, for example, the concepts seen for the planning phase of an Industrial Informatics System (IIS), such as: the activities of system definition and requirements analysis, documentation that reflects the needs of users, customer requirements and specifications. Also the levels of notation used in each of the documentation units. For the rest of this chapter, we will explain the different documents related to the Water Tank control project.

3.3 Document for system description and user needs

Let’s assume a particular customer who wants to modernize his industrial process. It is the distribution of hot water to certain small business: residential heating, sports facilities, etc.

This industrial application has a tank with water, which you want to control your level (0-8000 liters) and temperature (ranging from 0 to 100).

There is currently a manual control in which a skilled operator periodically reviews the status of deposit and updates its parameters (water level and temperature). The water inlet is provided by a pump: The pump runs continuously while the tank is filled. Once the water tank is at the required level, it starts heating to a temperature reference. Once achieved these goals, volume and temperature, we proceed to pipe the water.

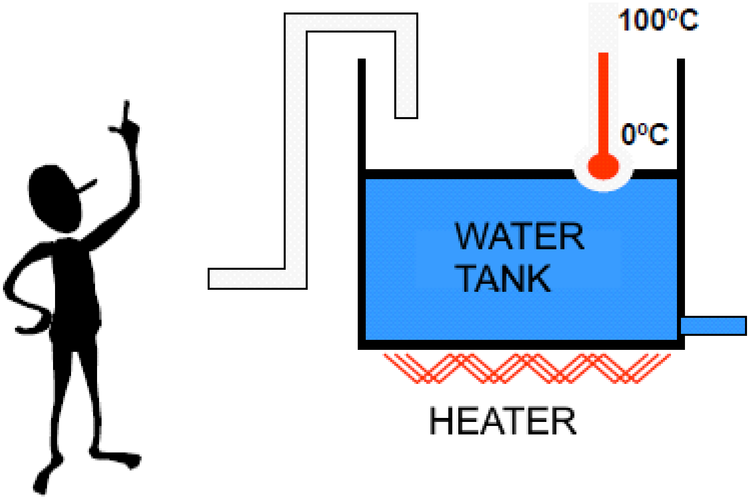


Figure 1: Industrial system with manual control

3.3.1 Document: from customer to designer

At first meeting, the customer and the designer analyse the needs of users of the industrial system. The shortcomings that must be solved, and the services that must be provided.

3.3.2 Reply Document: from designer to customer

In a subsequent meeting, the designer tells the client its readiness to implement the project, considering some aspects for the preparation thereof:

- The confirmation that the proposed project can be developed.

It also proposes to his client some improvements, such as:

- You can collect historical information on the operation of the process.

Calls permissions and guarantees, to enter the premises of the industrial process. If necessary, an insurance policy can be subscribed against human risk. This is because you must know the real boundary conditions of the system during project implementation.

- For example: You will need regular access to the process for measurement and testing. And for that, will require the help of specialized operator.

In the aim to improve and automate the manufacturing process, it will be needed to add new sensors and actuators so that in this way you can exercise a computer control.

In this previous document, the designer must specify the deadline for completion of the project to the client.

- For example: The project will be completed within 10 weeks.

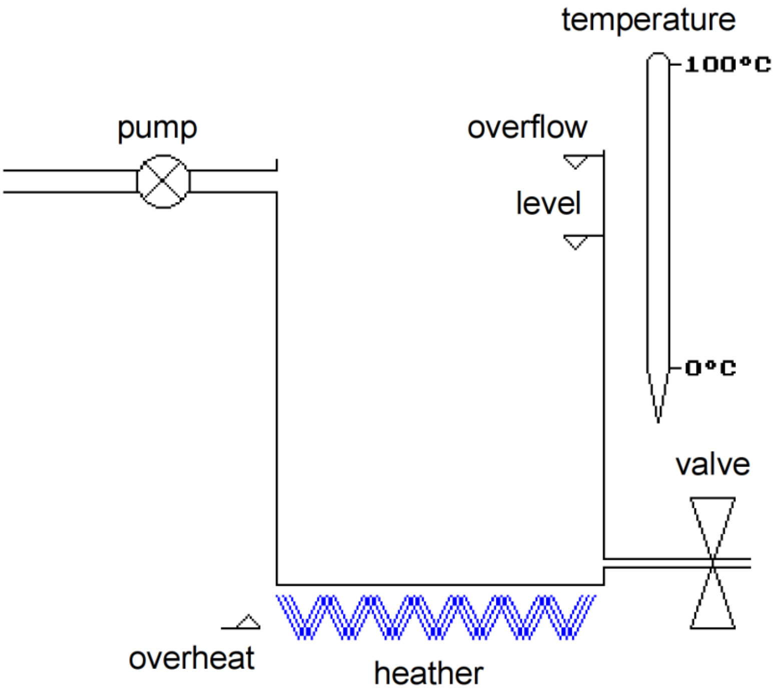


Figure 2: Outline of the proposed project

The designer should also make an outline, usually freehand, from the current system (Figure 1), outlines the system to be designed. See Figure 2. It consists of a tank with a liquid to be heated to a certain temperature in order to be used. The liquid is sent to the tank by a pump and is released by a valve.

A submerged heating element heats the water. To do this, you should be able to know the amount of liquid in the tank. The liquid in the tank can be measured by a volume sensor.

As a precaution, and for the control of energy consumption, it is proposed to add an overheating sensor to cover with both faulty states: either by heating when the tank is empty or above the desired temperature. The designer also proposes an overflowing sensor to control the maximum water volume in case the level sensor is broken.

3.4 Document of Requirements

Once agreed the project implementation, the designer must draft the document of requirements:

3.4.1 Technical requirements:

Construction of an automatic system for controlling the level between 0 and 8000 litres, and temperature between 0ºC to 100ºC of a water tank. The operator interface will show the system state (real process measures) and will allow the operator to change the goals: desired temperature and desired level. The controller must be able to automatically maintain the water temperature and level to the goal values.

The application must also produce printable periodic reports on performance. The deadline is set at ten weeks. The project will be accompanied by sufficient technical documentation.

3.4.2 Technical requirements to the customer

Access to industrial process will be facilitated, as well as support from the necessary operators.

The customer must provide the technical information of the currently installed sensor and actuators, if any, to allow the designed to study if they must me modified or adapted.

3.4.3 Analysis of requirements

Once agreed, client and designer, the designer proceed to the analysis of the requirements document and the preparation of the specification document.

Activity: Write Minimum Requirements Document

Activity: Expanding Requirements

It is proposed as an activity to improve every aspect of the project. As improvement ideas a series of guidelines are listed below:

- Design and build hardware to connect to the data acquisition card.
- Improve control loop considering more complex scenarios such as heating cycles / download, dosed, etc.
- Improve the operator interface by incorporating graphical representations of process status, historical data of temperatures, levels, relative humidity, etc.
- Record the time evolution, permanently storing values in files.
- Allow the selection of different operation modes: manual, automatic, etc.
- Add other sensors / actuators to improve the system.
- Use of tools for automatic generation of documentation. For example Doxygen.
- Use a software repository on the Internet (Sourceforge, Gitorius, Github, etc.).

3.5 Document of Specification

This document is performed by the designer and it contains among other, full details of the hardware and software components of the application. In the example we are developing, we have:

Digital inputs:

- Overflow: TTL compatible digital sensor; (ON = 5 V, OFF = 0V)

- Overheating: TTL compatible digital sensor; (ON = 5 V, OFF = 0V)

Analogue inputs:

- Temperature: 0º to 100º Celsius, sensor voltage range -5 to +5 volts.
- Level: 0 to 8000 liters, sensor voltage range -5 to +5 volts.

Digital outputs:

- Heater: Actuator TTL compatible; (ON = 5 V, OFF = 0V)
- Valve: TTL Compatible; (ON = 5 V, OFF = 0V)

Analog outputs:

- Pump: Analog actuator with voltage range of -10 to +10 volts

3.5.1 Specification of system signals

In the diagram of Figure 2 a name is given to each sensor and actuator. These names are used to describe these signals in the software to be implemented, thus each signal of the system will be identified with the same name in the software.

Therefore, the significance of signal values is specified in the various sensors and actuators:

The digital overflow sensor lets you know if there is danger of overflow of liquid. A value of "0" indicates a danger of overflow. A "1" indicates no overflow

The digital overheat sensor lets you know if there is danger of overheating. A value of "1" indicates such a danger. A value of "0" indicates no overheating.

The digital valve actuator allows sinking the water contained in the tank. The value "1" opens the valve. A value of "0" closes the valve.

The water temperature is measured by an inverse linear sensor which delivers a voltage of 5 volts at 100 ° C and 0 volts at 0 ° C.

The volume (or level) of the liquid is measured with a linear height sensor and delivers 5 volts for a height equal to 0 litres in the tank and from 0 volts to a height equivalent to 8600 litres.

The tank filling is performed by a pump, which is an analog actuator, with a power of 0% to 100%. The voltage range varies from 0 volts (0% power) to 10 volts (100% power).

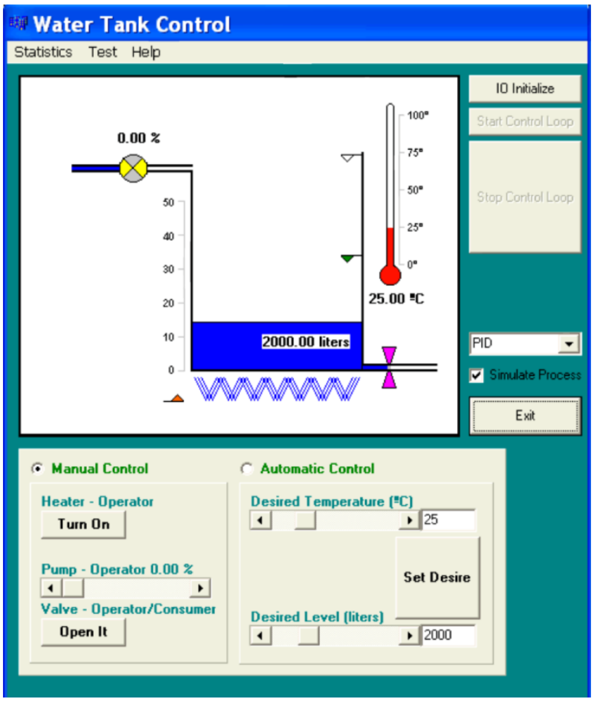
Activity:

The customer modifies the specifications on the heater. Now, the heater must be an analog actuator, and accept a DC control signal proportional to the power delivered. -5 Volt value corresponds to a 0% power and 5 volts corresponds to 100% power. Note that this signal can be set to any power between these limits, also manually via a potentiometer.

Regarding the pump in charge of tank filling, now it will have two control inputs, which are applied manually using a button, and a potentiometer. The button to start or stop the pump, and the potentiometer to change the pump power between the limits.

3.5.2 Operator Interface:

The following figure, shows the basic elements of the operator interface:



PID

☑ Simulate Process

● Manual Control ○ Automatic Control

Desired Temperature (ºC) 25

Desired Level (liters) 2000

Figure 3 shows a draft with the Basic elements operator interface.

Output: The computer monitor will show graphically the status of the process and the necessary buttons to access to the defined operations.

Input: The graphic user interface (GUI) will allow the user to enter the parameters by the user of the keyboard and mouse.

Control activities:

In a periodic rate, the various sensors are sampled and the outputs are adjusted accordingly (Period <2 seconds)

Hardware Modules:

Electronic card for input and output attached to the computer.

3.5.3 Estimated temporary planning for design and development:

- 1st Week: Preliminary design of software and hardware.
- 2nd and 3rd Weeks: Detailed design of the modules.
- 4th week: Software coding, hardware construction and prototype integration.
- 5th and 6th Weeks: Prototype testing.
- 7th Week: Analysis of the results of the prototype tests.
- 8th Week: Software coding, hardware construction and final system integration.
- 9th week: Test of final system.
- 10th week: Writing of user documentation and operator training.

3.6 Document: user guide

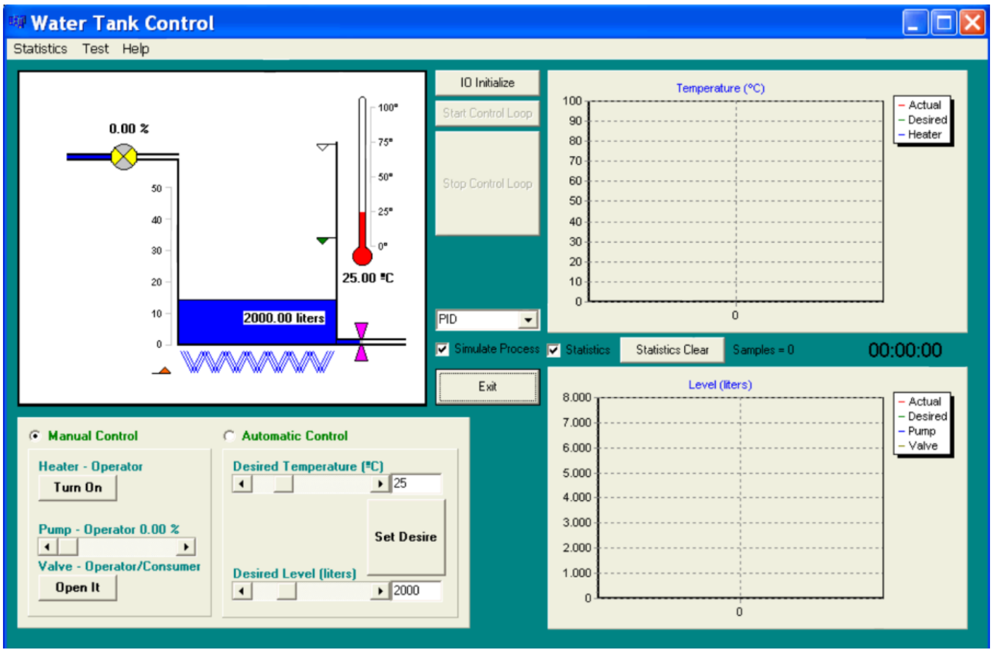
This document is performed by the application designer, consists of explaining each of the program options and should understand and control a normal user, but without explicit knowledge of the industrial computerization.

The user guide should explain simply and graphically, how the application works. That is, each of the inputs and outputs, which are offered to a user so you can control the process, such as menus and popup windows, inputs for edit boxes and labels.

It is quite advisable to design a help menu to enter goals and also the different actions that are permitted for the user of the application.

As an example of the schemes that can be included in the user guide, show a number of captures from the graphical user interface that has been done to control the tank.

In the figure 4, the main application window is displayed. Here, it is arranged various graphical components that can be used by any user of the process.



● Manual Control ○ Automatic Control

Pump - Operator 0.00 %

Desired Temperature (ºC) 25

Desired Level (liters) 2000

PID

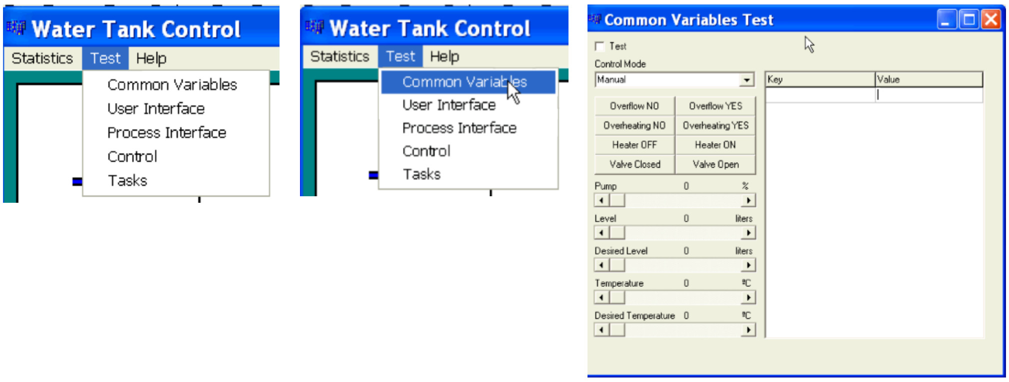
☑ Simulate Process ☑ Statistics Samples = 0 00:00:00

Figure 4: Graphical user interface of the application

In the central part it is observed buttons to start or quit the application. To the left it is shown a graph of the current status of deposit. On this chart the state of the variables are shown by labels such as pump power, current temperature, the amount of liquid in the tank, or graphic as the valve closed, or the heater off. Below the graph on the left, you can select one of two possible modes, manual or automatic.

Also, on the right side of the window, we can see the graphs that represent the evolution of the process variables (historical): temperature and level of liquid in the tank.

In the manual, you should also explain the different options offered by the application. In the figure 5, the dropdown menu of common variables of the application is shown, where the user can change some values of the goals variables.



□ Test

Control Mode: Manual

Pump 0 %

Level 0 liters

Desired Level 0 liters

Temperature 0 ºC

Desired Temperature 0 ºC

Figure 5: Dropdown menu options

4 Lab

For the documentation of the code of an application, in our case C ++, there are different tools. One of the most used is *Doxygen*. This tool allows to generate the documentation of the code that we have implemented in C ++ presented in any of the following formats: web page, html, Latex® or pdf.

From the *Doxygen* website (www.doxygen.org), you can download the program. Following the instructions and once installed the software, we can access some of the tutorials that exist to know how to use it in a practical way. For example for C ++ [1]. Among others, the fundamental steps to follow are the following:

1. Document the code. However, only prototypes of the functions we have implemented, which are in the "* .h" header files, should be documented.

The comments must be included at the beginning of each file. All comments are grouped within a generic C ++ comment, ("/ * ... * /"). The comments have a defined structure. That is, they all start with an asterisk "*" followed by an "@", with the word "key" that defines the comment that follows. For example, the file name, version, author, date, title, would come as follows:

```
/* *

* @file (This is the name of the header file, for example: functions.h)

* @version (refers to the version of the code, for example: 1.0)

* @author (You must include the Name & Surname of the code designer)

* @date (Indicates the date of completion of that code)

*  @title  (Indicates  what  kind  of  functions  it  performs,  for  example:
geometric functions)

* @brief (Indicates small feature comments)

*/
```

2. If you want to include a code example of how these functions are used, use the * @code statement with the code in the middle, ending with: * @endcode, for example:

```
* @Code

// Here are the statements of C
* @Endcode
```

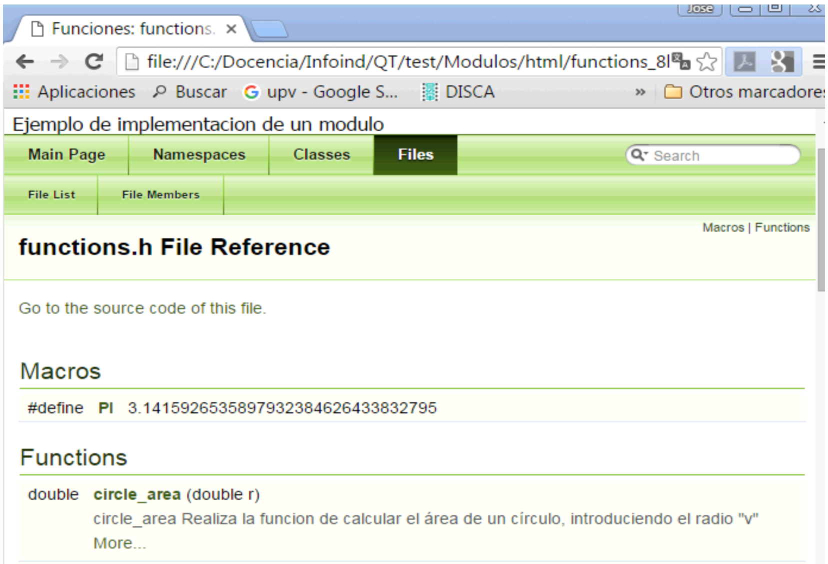
3. In the header file, which we have implemented in each module, the comments must be added to document our functions. Thus, in the example with the functions of geometric calculation of the areas of a circle and a rectangle, in the header file (functions.h) we add the following comments, for example, to document the function of the area of a circle:

```
/**

*@brief circle_area Performs the function of calculating the area of a
circle by entering the radius "v"
* @param v of real type is the variable that represents the radius
* @return The area of the circle = Pi * radius * radius (a real value)
*/

Double circle_area (double v);
```

4. After documenting all the code of our functions, "Doxygen" is executed, declaring the location of the code, with the path where the project code is, for example the Qt folder of our application project, and the destination where we want to store the final project documentation, which can be the same folder as the Qt application. Once you have all parameters set in Doxygen, run the application and display the output of documents, for example in html, pdf, or latex.
5. In Figure 6 we can see the result of the documentation that has been generated for the example of one of the functions of the module (functions.cpp + functions.h,). The doxygen program allows you to navigate through the different tabs of the website you have generated and view all the documentation.



Ejemplo de implementacion de un modulo

functions.h File Reference

Go to the source code of this file.

Macros

#define **PI** 3.14159265358979323846264338327950

Functions

double **circle_area** (double r)
circle_area Realiza la funcion de calcular el área de un círculo, introduciendo el radio "v" More...

Figure 6: Documentation generated by Doxygen

5 Seminar

In the seminar, it is proposed to search in the web the following resources:

Search for models of legal contracts for a factory installation, and propose changes to adapt to your Industrial Informatics System project.

Look for advanced doxygen functions. For example make dependency graphs.

Search for other doxigen-like tools for automatic doc generation

6 Mini-project

Document the code of the mini-project.

Complete all project documents: requirements, specifications and a user guide.

Also write the document of a programmer's reference including documentation of the code generated by Doxygen.

7 References

[1] www.**doxygen**.org/

[2] https://www.youtube.com/watch?v=9JGCVseIq8w