FINAL PUBLISHABLE SUMMARY REPORT

Grant Agreement number: 243484

Project acronym: PLC-PROG

Project title: *Improving PLC Programming Through a New Graphical, Object-Oriented and Brand-Independent Programming Framework*

Funding Scheme: Research for the benefit of SME Associations

Period covered: 01/09/2010 **to** 31/08/2013

Date of release: 22/11/2013

Name of the scientific representative of the project's co-ordinator

Mr. Valerio Grosso **LABOR S.r.l.**

Tel: +39 06 40040354

Fax: +39 06 40040357

E-mail: v.grosso@labor-roma.it

Project website address: www.plc-prog.eu

Disclaimer: This document is the property of the PLC-PROG Consortium. This document may not be copied, reproduced, or modified in the whole or in the part for any purpose without written permission from the PLC-PROG coordinator with acceptance of the Project Consortium.

This project was completed with the support of the Research Executive Agency (REA) of the European Commission under the 7th Framework Programme. The contents of this publication do not necessarily reflect the Commission's own position.

Summary

1. Executive summary 4

2. Project context and the main objectives....7

3. Main S&T results (Foreground) 10

3.1. PLC-PROG solution 10

3.2. PLC-PROG building components 13

3.3. PLC-PROG IDE 23

3.4. SCADA approach 33

3.5. Versioning (or revision control system) 37

3.6. Post-processors for three targeted major commercial brands 40

3.7. Application and validation of the PLC-PROG system in pilot plants 45

4. Potential impact, dissemination activities and the exploitation of results 51

4.1. Socio economic impact....51

4.2. Dissemination of results 56

4.2.1. Web site 66

4.2.2. Corporate Video, Brochure and poster....67

4.2.3. Pages on social networks for PLC-PROG project....68

4.3. Exploitation of results 71

4.3.1. PLC Application 72

4.3.2. PLC-PROG Code Generator....75

4.3.3. PLC-PROG Post Compilers....76

4.3.4. Exploitation Strategy....77

5. Public website address and relevant contact details. 87

1. Executive summary

PLC-PROG is much more than a software. It opens the way to the standardisation of program code in PLCs. Following the basis of the IEC 61131-3 Standard, PLC-Prog defines a visual language based on racks that permits the user to create complex PLC programs with simple drag and drop operations.

The PLC is still stuck in software development environments equivalent to late 80's PC software development. This delay in evolution has meant an increase in the complexity of programming PLC devices, increased program development time and extreme brand dependency.

In this context, where problems of brand-dependence and ease of use strongly affect PLCs programming, **PLC-PROG developed a truly automation PLC based Brand-Independent programming platform, which will provide a solution to SME-AG members and to the automation sector as a whole.**

At this purpose, **PLC-PROG incorporates the following innovations**:

- A high-level automation language with **full support for Object Oriented programming**
- The programming will be carried out **graphically** over an Integrated Development Environment IDE (that is, the visual and integrated development platform). New and/or pre-existing components will be "dragged" and connected over the IDE, which will mirror the physical elements used in the construction of electrical control cabinets.
- A unified programming environment **valid for different PLC brands.** PLC-PROG will fully integrate the IEC-61131 standard set by the EC, increasing programming features in PLCs. Programs generated within the PLC-PROG framework will **run potentially on any PLC brand** compliant with the standard.

PLC-PROG, by adding post-processors for the main PLC brands, will fully cover all capabilities of the devices. Nowadays, this is a main issue as in most of PLC devices the standard only cover a limited set of features and this fact forces user to use the specific software provided by the PLC vendor.

The easier way to reuse code is to program once and reuse it everywhere, and this is how PLC-Prog works. The user can create a single program, configure the physical inputs and outputs and export it to a desired brand. Easy, fast and debugged.

One of the bigger sources of errors and problems is to type the program code. In two main cases:

- to retype similar functionalities depending on parameters.
- to retype the code for a different PLC brand.

Using PLC-Prog this part is minimized drastically:

- The first problem is solved by using racks and functions. (about 20% time saving)
- The second problem is also solved because PLC-Prog includes a postprocessor for any brand that supports IEC 61131-3 Standard (at the moment Siemens, ABB and Schneider are fully supported). (about 80% time saving)

And extended tools have been created to debug the commissioning phase and to create a quick and effective SCADA screen for values visualization.

All these tools are based on the extended automation standard for communication: OPC DA and UA.

The post-processors are responsible for translating a PLC-PROG PLC program, which is fully compatible with the IEC-61131-3 standard, into a specific, non-fully compatible code, as needed by any of the target PLC brands in the market.

The post processor developed in the project is able to write automatically the final code for the control program to be downloaded into 5 different PLCs (ABB, PLCOpen XML, Telemecanique, Siemens or B&R), as it can be seen in the following figure. At this regard, it is worth to underline as two additional brands have been added in respect to the preliminary chosen ones, (namely: B&R and PLCOpen XML) since they are actually in use among some of the partners.

The PLC-PROG system has been tested with industrial processes (two water plants in Romania and one injection moulding machine in UK), in order to assess both the viability and the performance of the PLC-PROG programming tool.

The validation performed in the last part of the project on the industrial processes has showed that the project technical objectives have been fully achieved, and in particular that:

- It is possible to create the code for an industrial application using PLC-PROG, with a **programming time** which, for an user already experienced in PLC-PROG use, can be estimated at least equal and in some cases **lower** than with a conventional language.
- It is possible **to reuse** the same program code with 2 or more different PLC brands with minimal effort, leading to **an average reduction of the implementation time of 85%**, which is an impressive result.
- It is possible **to reuse** existing code from old application and encapsulate it in PLC-PROG functions for further use.
- It is possible **to create SCADA visualization** of the PLC variables' values with minimal configuration to embed in a SCADA software or Web Browser.

The combination of the features have been positively evaluated by the users from both an application and market point of view, and allow to state that:

- **PLC-PROG saves program development time** when using different PLC and SCADA brands because the same program can be easily adapted to supported brands.
- **PLC-PROG saves program maintenance time** when making modifications in the functions because the versioning system helps to restore/update functions easily.
- **The learning curve is fairly short** as the software is very intuitive. During the training sessions the attendants could create simple programs.
- **PLC-PROG permits non expert users to create or maintain PLC programs**.

2. Project context and the main objectives

During the last 20 years the automation sector has experienced an incredible expansion, which was hardly imaginable just 10 years ago; today, the monitoring and even the control of a factory located thousands of miles away has become possible, as well as the reception of production data, values of efficiency, consumption, etc. of a given real-time production is now possible directly on a mobile phone. Nowadays, the business competitiveness is incredibly increased and the costs must be reduced significantly in order to keep a valuable market position.

In this context, the **PLCs have made great progress as for performance and complexity**. They are designed to replace the hard-wired from relays, timers, counters, etc. substantially saving wiring and maintenance operations, and at the same time allowing the installations to have higher flexibility at the sole programming cost. Developments in the field of programming started from programs that could be fully understood only by the programmers themselves, these programs proving very expensive to maintain and implying huge dependence on the programmer.

In this situation, *any necessary modification of the PLC program to be performed by a person other than the original programmer normally resulted in a complete re-programming*. It is known that it is extremely easier to create a new control program than trying to understand and interpret a program created by another author.

The problem has been now minimized in large companies - such as the ones operating in the automotive sector - by structuring the control program using **templates**. This allows a maintenance activity which is independent of the programmer. The responsible engineering staff for process automation and industrial installations must be composed of several different experts capable of programming PLCs brand by brand, if PLCs programming is to be optimized and made efficient.

There is **no automatic conversion**, up to now, that allows a control program to be converted into different PLCs brands. The programmer is the only person who can manage the conversion, and perform it manually.

Actually, every PLC brands supporting IEC-61131-3 work with different standard programming languages, from graphical to text ones. The most popular for a generic end-user, and the most diffused, is the so-called "*Ladder*" or "*contact diagrams*" due to its similarity with the electrical line diagrams. On the other hand, some text languages are also widely used by skilled programmers for their potential, even if these languages are barely known by the end-users.

At this scope, **PLC-PROG Project aims to provide a single programming tool for PLCs which are compliant with the standard IEC 61131-3 to finally solve all the problems mentioned above related to PLCs programming.**

This programming tool for PLCs was targeted to have the following features:

- **to be valid for all the PLC brands which are compliant with the IEC 611131-3 standard**, among which we can find the most diffused in the automation market: Siemens, Telemecanique, ABB, B&R, Phoenix Contact, Beckhoff, etc. Using PLC-PROG tool *you won't need to be an expert in programming each single proprietary software of each specific brand*, since the same project can be exported to any of the aforementioned brands.
- **to be based on a graphical programming methodology** showing evident analogies with the physical elements generally present in electrical installations and electrical panels in the plant, so that the maintenance staff in the facility is able to interpret, modify and even schedule the different operations, being the program based on the assembly of objects like in a switchboard.
- **No need of specific programming skills** while using PLC-PROG, as the tool only needs the assembly of different objects corresponding to the physical elements displaced in an electrical box; advanced users will perform the programming of a new functionality - which is not developed yet - in case this functionality appears to be necessary.
- **to be based on the object-oriented programming** that has shown such outstanding results in the computers world. The key is to create objects that represent existing electrical objects such as a relay, a PID controller, filters, etc. Thanks to the properties of the object oriented programming (*polymorphism, inheritance, encapsulation, use of templates*, etc.) it allows objects reusability.
- **to perform both the monitoring and the visualization from the tool** or even from SCADA commercial packs, through an insertion of an object ActiveX or Applet which provide an insuperable functionality and versatility of the tool itself.

The unique selling proposition of PLC-PROG is to be the first tool applied to PLC programming to take advantage of a completely brand independent and user-friendly OO approach, **granting a reduction of the development times** and an increase in the **reliability** of the developed programs.

This was meant to be achieved thorugh the following objectives:

- the development of a graphical approach to developing and documenting PLC programs in an intuitive manner;
- the development of post-processors for translating a PLC-PROG project into native PLC code for 3 major PLC brands compliant with the IEC-61131 standard;
- the development of post-processors for integrating the PLC-PROG project into 3 major SCADA commercial programs;
- the development of the basic virtual modules for proper PLC software development;
- the development of Supervisory Control and Data Acquisition modules (OO-SCADA) for granting communication and data integrity between SCADA and PLCs;
- the validation of the quality of the system.

3. Main S&T results (Foreground)

3.1. PLC-PROG solution

Programmable Logic Controller (PLC) or programmable controller is a digital computer used for automation of electromechanical processes, such as control of machinery on factory assembly lines, amusement rides, or light fixtures. PLCs are used in many industries and machines.

Early PLCs were designed to replace relay logic systems. These PLCs were programmed in "ladder logic", which strongly resembles a schematic diagram of relay logic. This program notation was chosen to reduce training demands for the existing technicians. Other early PLCs used a form of instruction list programming, based on a stack-based logic solver.

PLCs are **essential components of modern life**, running everything from simple soft drink or candy machines to industrial equipment and highly sophisticated medical diagnosis and treatment equipment. However **the PLC is still stuck in software development environments equivalent to late 80's PC software development.** This delay in evolution has meant an increase in the complexity of programming PLC devices, increased program development time and extreme brand dependency.

In this context, where problems of brand-dependence and ease of use strongly affect PLCs programming, **PLC-PROG project intends to develop a truly automation PLC based Brand-Independent programming platform, which will provide a solution to SME-AG members and to the automation sector as a whole.**

Specifically, all the associations involved in the project (SEPE, BPF and GAIA) have agreed that innovation in PLC software development, in terms of modernization of the development itself and maintenance of PLC programs, and the ability to handle the large number of brands on the market, is a **common need to automation sectors and big brands won't drive the evolution of PLC programming environments** due to their own particular interests. Considering all of these issues, the Consortium believe a Collective project has great significance for the development of the PLC-

PROG system, providing critical mass for initial dissemination and exploitation from the beginning for the success of the product.

At this purpose, **PLC-PROG incorporates the following innovations**:

- A high-level automation language with **full support for Object Oriented programming**
- The programming will be carried out **graphically** over an Integrated Development Environment IDE (that is, the visual and integrated development platform). New and/or pre-existing components will be "dragged" and connected over the IDE, which will mirror the physical elements used in the construction of electrical control cabinets.
- A unified programming environment **valid for different PLC brands.** PLC-PROG will fully integrate the IEC-61131 standard set by the EC, increasing programming features in PLCs. Programs generated within the PLC-PROG framework will **run potentially on any PLC brand** compliant with the standard.

Moreover, PLC-PROG innovative character and technological challenge has several advantages, the main of which being: PLC-PROG should **improve** various **efficiency levels in development and maintenance,** with respect to present PLC programming systems.

Currently, a number of different products are available for PLC programming, and most of them are brand-dependent. The products generally include the PLC programming, simulation and elements for the connection of the computer and other devices, for example:

- ✓ **CodeSys: Con**troller **De**velopment **Sys**tem is a comprehensive software tool for industrial automation technology. The runtime system turns any device into an IEC 61131-3 controller programmable with CoDeSys. It is intended to be brand independent, although the interface is not graphical and is based on line code programming.
- ✓ **TwinCat**: Software for PLC programming and monitoring. TwinCAT PLC offers all the languages in the IEC 61131-3 standard and has a powerful development environment for programs with code size and data regions that far exceed the capacities of conventional PLC systems.

- ✓ **Siemens STEP 7 Lite**: STEP 7 Lite is designed for traditional PLC applications with centralised I/O. STEP 7 Lite is designed for non-networked automation applications using the SIMATIC S7-300 PLC, the C7 all-in-one PLC and HMI, and the intelligent CPUs of the ET200 distributed I/O family.
- ✓ **OMRON**: The CX-One Integrated Tool Package provides an environment to operate the Support Software required for all processes, from FA system design to operation and maintenance.
- ✓ **Hitachi Pro-H**: Pro-H is universally usable 32 bit programming software for all Hitachi H-Series PLC that provides for a flexible choice of editors.

Based on the information and technology analysis carried out on the above listed solution, together with the ones listed in the previous section of this document, it is possible to state that:

the most similar ones to PLC-PROG are **CodeSys** and **TwinCat**. These products **are limited only to the IEC 61131-3 standard** being this the main disadvantage in front of PLC-PROG as the standard IEC 61131-3 only covers a part of the capabilities of a PLC.

PLC-PROG, by adding post-processors for the main PLC brands, will fully cover all capabilities of the devices. Nowadays, this is a main issue as in most of PLC devices the standard only cover a limited set of features and this fact forces user to use the specific software provided by the PLC vendor.

What is PLC-PROG?

PLC-PROG is much more than a software. It opens the way to the standardisation of program code in PLCs. Following the basis of the IEC 61131-3 Standard, PLC-Prog defines a visual language based on racks that permits the user to create complex PLC programs with simple drag and drop operations.

The main purpose is to reuse code making PLC program's design, development and commissioning easier and faster. All the functionalities are encapsulated in functions and mounted in racks, so the user just need to configure the program as desired.

The easier way to reuse code is to program once and reuse it everywhere, and this is how PLC-Prog works. The user can create a single program, configure the physical inputs and outputs and export it to a desired brand. Easy, fast and debugged.

One of the bigger sources of errors and problems is to type the program code. In two main cases:

- to retype similar functionalities depending on parameters.
- to retype the code for a different PLC brand.

Using PLC-PROG this part is minimized drastically:

- The first problem is solved by using racks and functions (about 20% time saving).
- The second problem is also solved because PLC-PROG includes a postprocessor for any brand that supports IEC 61131-3 Standard (at the moment Siemens, ABB and Schneider are fully supported). (about 80% time saving)

And extended tools have been created to debug the commissioning phase and to create a quick and effective SCADA screen for values visualization.

All these tools are based on the extended automation standard for communication: OPC DA and UA.

3.2. PLC-PROG building components

The preliminary RTD activities implemented has been devoted to building the foundations of PLC-PROG:

I. The root components that every PLC-PROG component inherits.

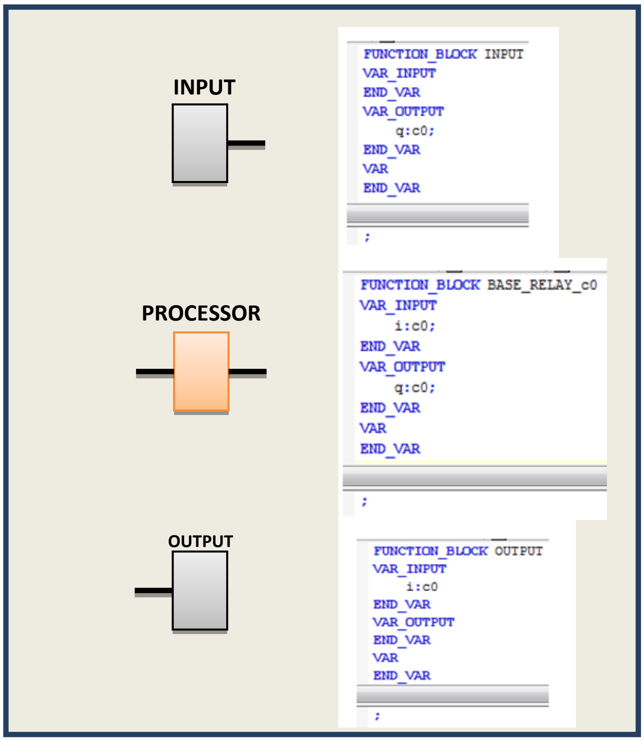
II. The polymorphic connectors that allow the inter-communication between PLC-PROG components (c0) and their corresponding scaled aggregations (c1, c2, etc.)

III. The generic structures that operate on PLC-PROG components: cards, racks, etc.

They are the components that the GUI presents to the programmer for building a PLC-PROG program, the components that the end user monitors in the SCADA components, and the elements that are post processed to assembly a Brand specific code.

ROOT PLC-PROG elements

The root PLC-PROG elements, from which any other PLC-PROG component is derived, are the input, relay and output component (Fig. 1):



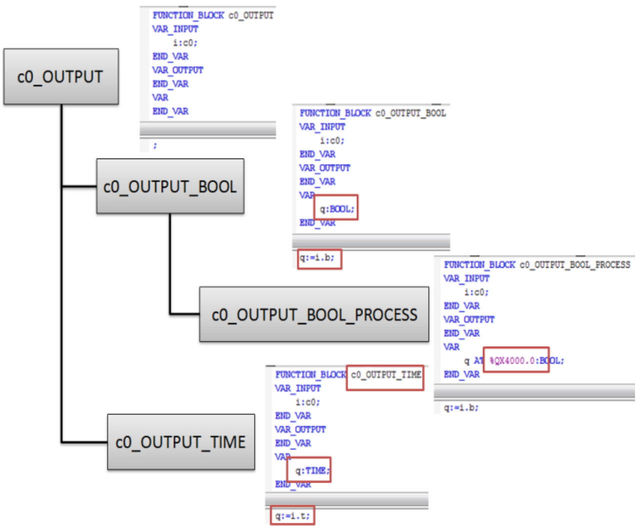
```
FUNCTION_BLOCK INPUT
VAR_INPUT
END_VAR
VAR_OUTPUT
    q:c0;
END_VAR
VAR
END_VAR

;
```

```
FUNCTION_BLOCK BASE_RELAY_c0
VAR_INPUT
    i:c0;
END_VAR
VAR_OUTPUT
    q:c0;
END_VAR
VAR
END_VAR

;
```

```
FUNCTION_BLOCK OUTPUT
VAR_INPUT
    i:c0
END_VAR
VAR_OUTPUT
END_VAR
VAR
END_VAR

;
```

Fig. 1 - Root elements of PLC-PROG

To generate a descendant from any of these root components, an inheritance mechanism has been implemented. As the present version of the IEC 61131-3 standard does not include this inheritance feature, PLC-PROG has implemented its own inheritance mechanism, as shown in Fig. 2 for the construction of output components derived from the root one.



```
FUNCTION_BLOCK c0_OUTPUT
VAR_INPUT
    i:c0;
END_VAR
VAR_OUTPUT
END_VAR
VAR
END_VAR
```

```
;
```

```
FUNCTION_BLOCK c0_OUTPUT_BOOL
VAR_INPUT
    i:c0;
END_VAR
VAR_OUTPUT
END_VAR
VAR
    q:BOOL;
END_VAR
```

```
q:=i.b;
```

```
FUNCTION_BLOCK c0_OUTPUT_BOOL_PROCESS
VAR_INPUT
    i:c0;
END_VAR
VAR_OUTPUT
END_VAR
VAR
    q AT %QX4000.0:BOOL;
END_VAR
```

```
q:=i.b;
```

```
FUNCTION_BLOCK c0_OUTPUT_TIME
VAR_INPUT
    i:c0;
END_VAR
VAR_OUTPUT
END_VAR
VAR
    q:TIME;
END_VAR
```

```
q:=i.t;
```

Fig. 2 - Inheritance mechanism for deriving descendant components

In the case of a relay component, a special program has been developed for automatically encapsulating existing code into a PLC-PROG relay element (Fig. 3). Also, an integrated editor has been developed for producing new relays (Fig. 4).

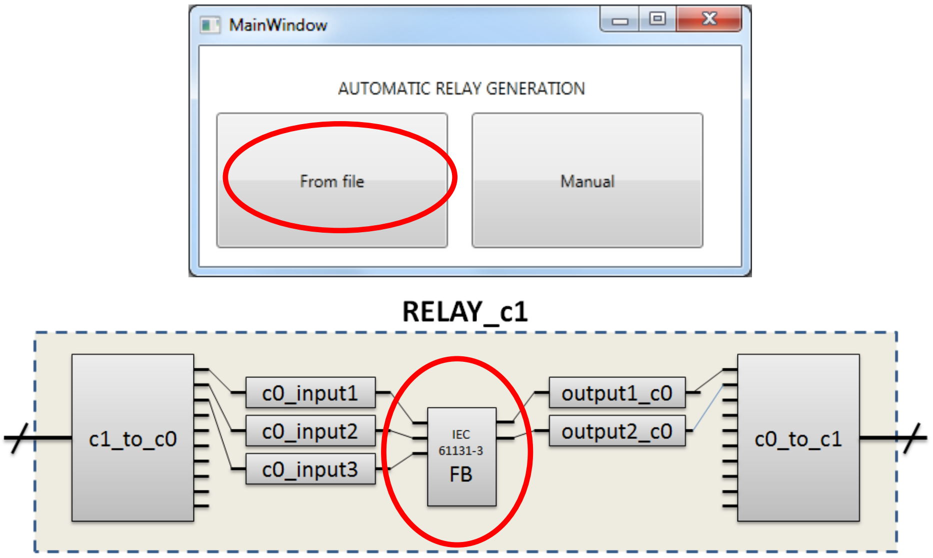
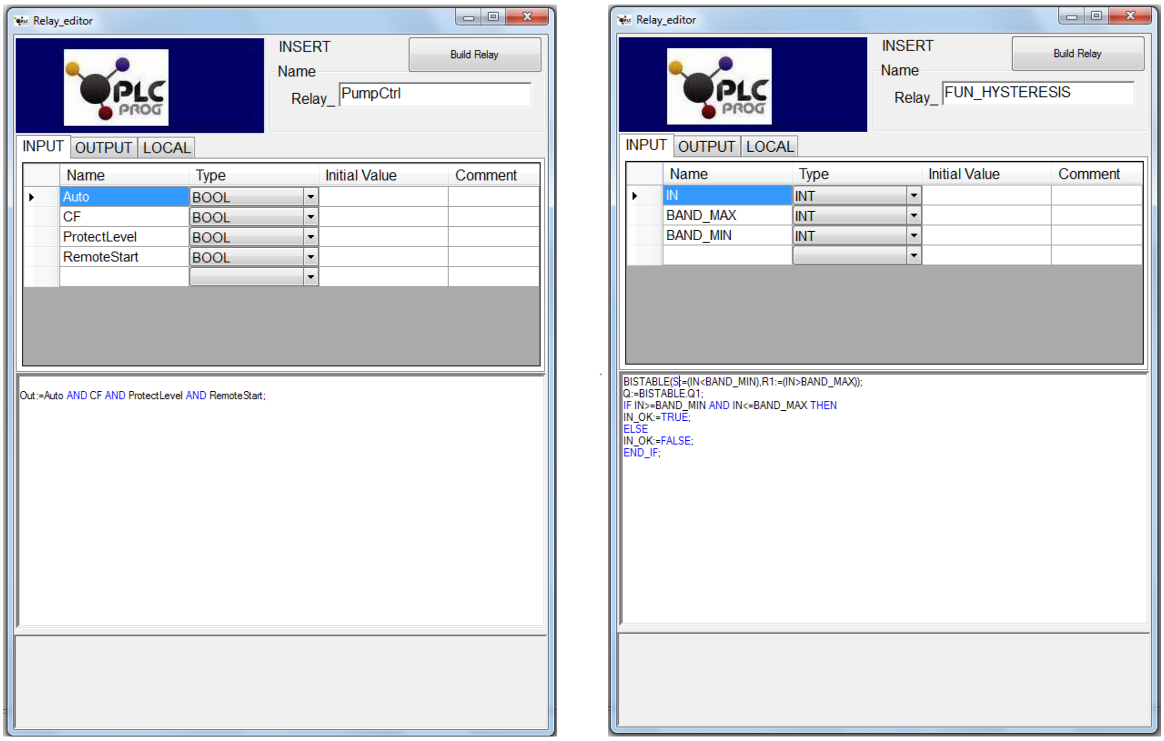


Fig. 3 - Automatic program for encapsulating existing FBs into PLC-PROG Relay components



Relay_ PumpCtrl

Name	Type	Initial Value	Comment
Auto	BOOL		
CF	BOOL		
ProtectLevel	BOOL		
RemoteStart	BOOL		

```
Out:=Auto AND CF AND ProtectLevel AND RemoteStart;
```

Relay_ FUN_HYSTERESIS

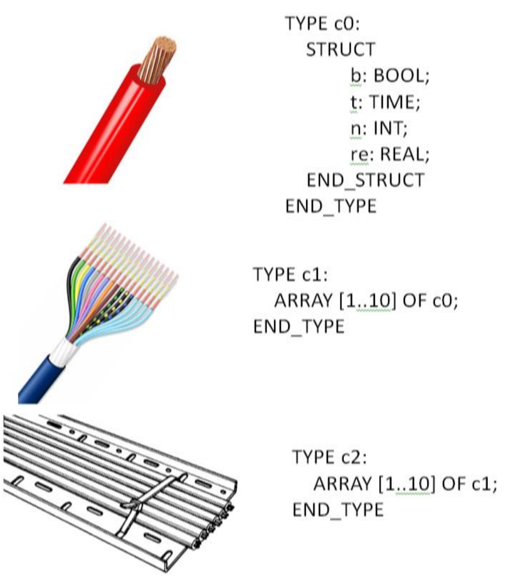
Name	Type	Initial Value	Comment
IN	INT		
BAND_MAX	INT		
BAND_MIN	INT		

```
BISTABLE(S:=(IN<BAND_MIN),R1:=(IN>BAND_MAX));
Q:=BISTABLE.Q1;
IF IN>=BAND_MIN AND IN<=BAND_MAX THEN
IN_OK:=TRUE;
ELSE
IN_OK:=FALSE;
END_IF;
```

Fig 4 - Integrated editor for building PLC-PROG relays

Connectors

Connectors in PLC-PROG are polymorphic: a single conductor can carry any type of data, in close analogy to real life, where electrical conductors can carry any type of information: analogical, digital, etc. Besides, as in real life, single conductor (C0 in PLC-PROG terminology) can be grouped to facilitate the task of connecting elements: 10 C0 conductor form one C1 connector, 10 C1 conductors are grouped into a single C2 ccable, and so on.



```
TYPE c0:
   STRUCT
         b: BOOL;
         t: TIME;
         n: INT;
         re: REAL;
   END_STRUCT
END_TYPE
```

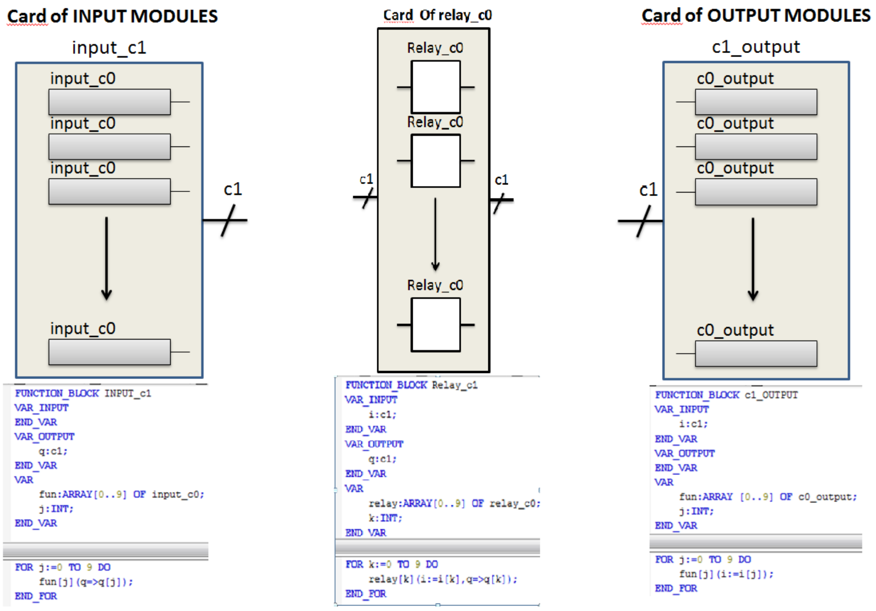
```
TYPE c1:
   ARRAY [1..10] OF c0;
END_TYPE
```

```
TYPE c2:
   ARRAY [1..10] OF c1;
END_TYPE
```

Fig. 5 - Polymorphic connectors in PLC-PROG (c0), and groups of connectors (c1, c2, etc.)

Generic Components

As all PLC-PROG components share a common interface, they can be assembled and interconnected in generic components that operate on them, irrespectively of their content. For example, a card component is created by packing 10 components into a single one, as in Fig. 6.



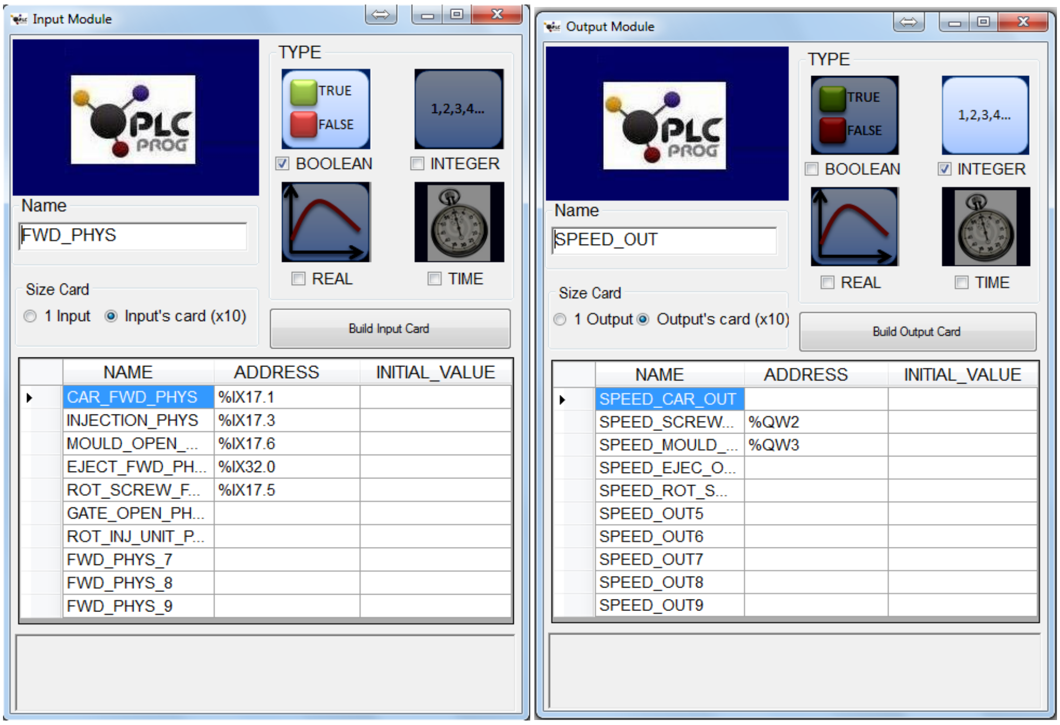
```
FUNCTION_BLOCK INPUT_c1
VAR_INPUT
END_VAR
VAR_OUTPUT
    q:c1;
END_VAR
VAR
    fun:ARRAY[0..9] OF input_c0;
    j:INT;
END_VAR

FOR j:=0 TO 9 DO
    fun[j](q=>q[j]);
END_FOR
```

```
FUNCTION_BLOCK Relay_c1
VAR_INPUT
    i:c1;
END_VAR
VAR_OUTPUT
    q:c1;
END_VAR
VAR
    relay:ARRAY[0..9] OF relay_c0;
    k:INT;
END VAR

FOR k:=0 TO 9 DO
    relay[k](i:=i[k],q=>q[k]);
END_FOR
```

```
FUNCTION_BLOCK c1_OUTPUT
VAR_INPUT
    i:c1;
END_VAR
VAR_OUTPUT
END_VAR
VAR
    fun:ARRAY [0..9] OF c0_output;
    j:INT;
END_VAR

FOR j:=0 TO 9 DO
    fun[j](i:=i[j]);
END_FOR
```

Fig. 6 - Card components of each of the PLC-PROG root elements (top) and interface

An integrated editor has been included for producing card components (Fig. 7).



Input Module

TYPE

TRUE FALSE ☑ BOOLEAN

1,2,3,4... ☐ INTEGER

☐ REAL

☐ TIME

Name: FWD_PHYS

Size Card: ○ 1 Input ● Input's card (x10)

Build Input Card

NAME	ADDRESS	INITIAL_VALUE
CAR_FWD_PHYS	%IX17.1	
INJECTION_PHYS	%IX17.3	
MOULD_OPEN_...	%IX17.6	
EJECT_FWD_PH...	%IX32.0	
ROT_SCREW_F...	%IX17.5	
GATE_OPEN_PH...		
ROT_INJ_UNIT_P...		
FWD_PHYS_7		
FWD_PHYS_8		
FWD_PHYS_9		

Output Module

TYPE

TRUE FALSE ☐ BOOLEAN

1,2,3,4... ☑ INTEGER

☐ REAL

☐ TIME

Name: SPEED_OUT

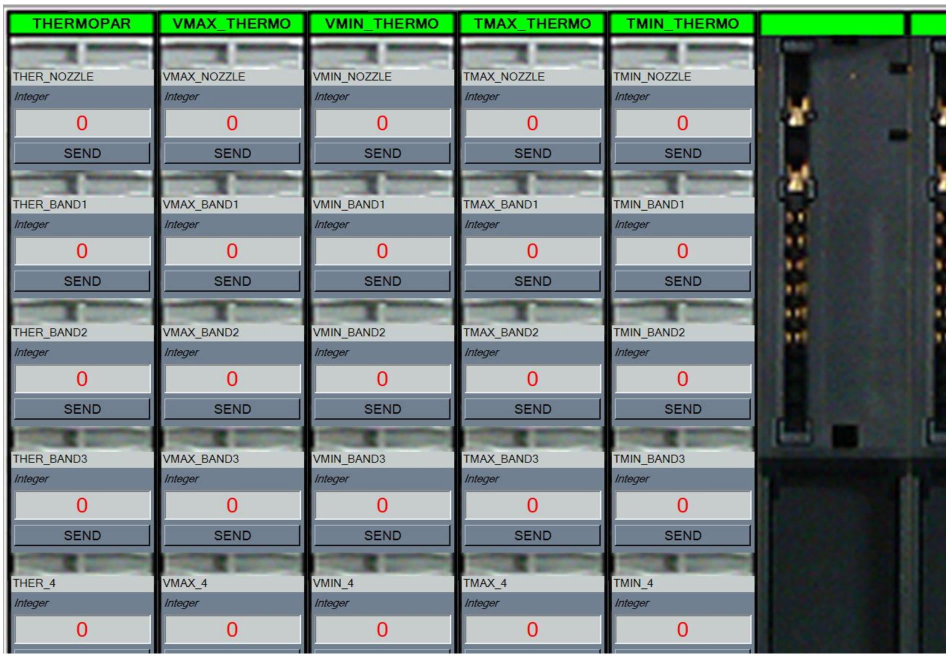
Size Card: ○ 1 Output ● Output's card (x10)

Build Output Card

NAME	ADDRESS	INITIAL_VALUE
SPEED_CAR_OUT		
SPEED_SCREW...	%QW2	
SPEED_MOULD_...	%QW3	
SPEED_EJEC_O...		
SPEED_ROT_S...		
SPEED_OUT5		
SPEED_OUT6		
SPEED_OUT7		
SPEED_OUT8		
SPEED_OUT9		

Fig. 7 - Editor for assembling input and output cards

Besides, card components (of inputs, outputs and relays) can be inserted into a rack components, which is the main processing unit of the PLC-PROG system (Fig. 8).



THERMOPAR	VMAX_THERMO	VMIN_THERMO	TMAX_THERMO	TMIN_THERMO
THER_NOZZLE Integer 0 SEND	VMAX_NOZZLE Integer 0 SEND	VMIN_NOZZLE Integer 0 SEND	TMAX_NOZZLE Integer 0 SEND	TMIN_NOZZLE Integer 0 SEND
THER_BAND1 Integer 0 SEND	VMAX_BAND1 Integer 0 SEND	VMIN_BAND1 Integer 0 SEND	TMAX_BAND1 Integer 0 SEND	TMIN_BAND1 Integer 0 SEND
THER_BAND2 Integer 0 SEND	VMAX_BAND2 Integer 0 SEND	VMIN_BAND2 Integer 0 SEND	TMAX_BAND2 Integer 0 SEND	TMIN_BAND2 Integer 0 SEND
THER_BAND3 Integer 0 SEND	VMAX_BAND3 Integer 0 SEND	VMIN_BAND3 Integer 0 SEND	TMAX_BAND3 Integer 0 SEND	TMIN_BAND3 Integer 0 SEND
THER_4 Integer 0	VMAX_4 Integer 0	VMIN_4 Integer 0	TMAX_4 Integer 0	TMIN_4 Integer 0

Fig. 8 - Detail of the input cards of a rack component

Racks can be connected to perform complex operations, as in Fig. 9.

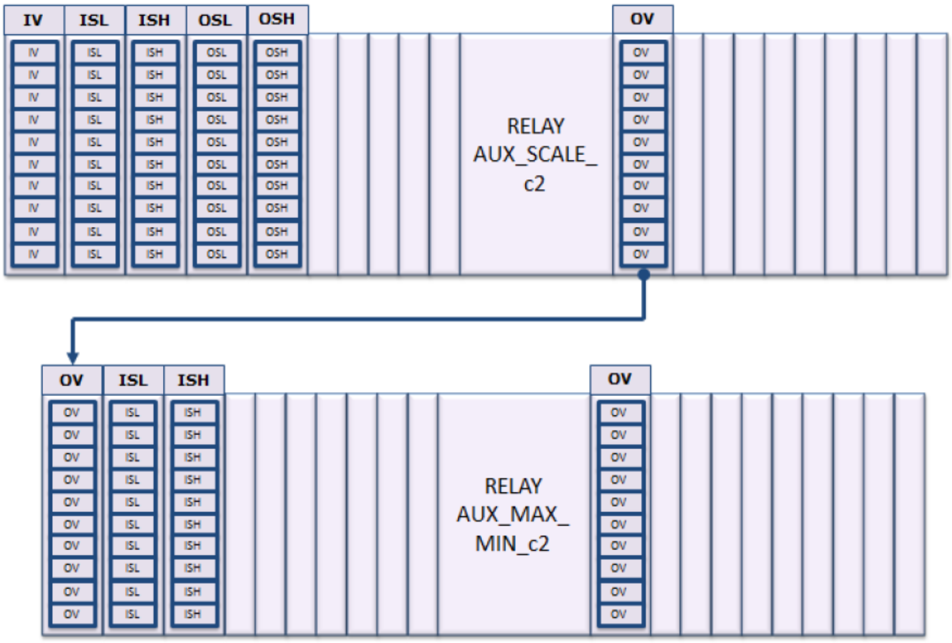


Fig. 9 - Interconnection of racks

Finally, every PLC-PROG program is expressed as a hierarchy of interconnected racks:

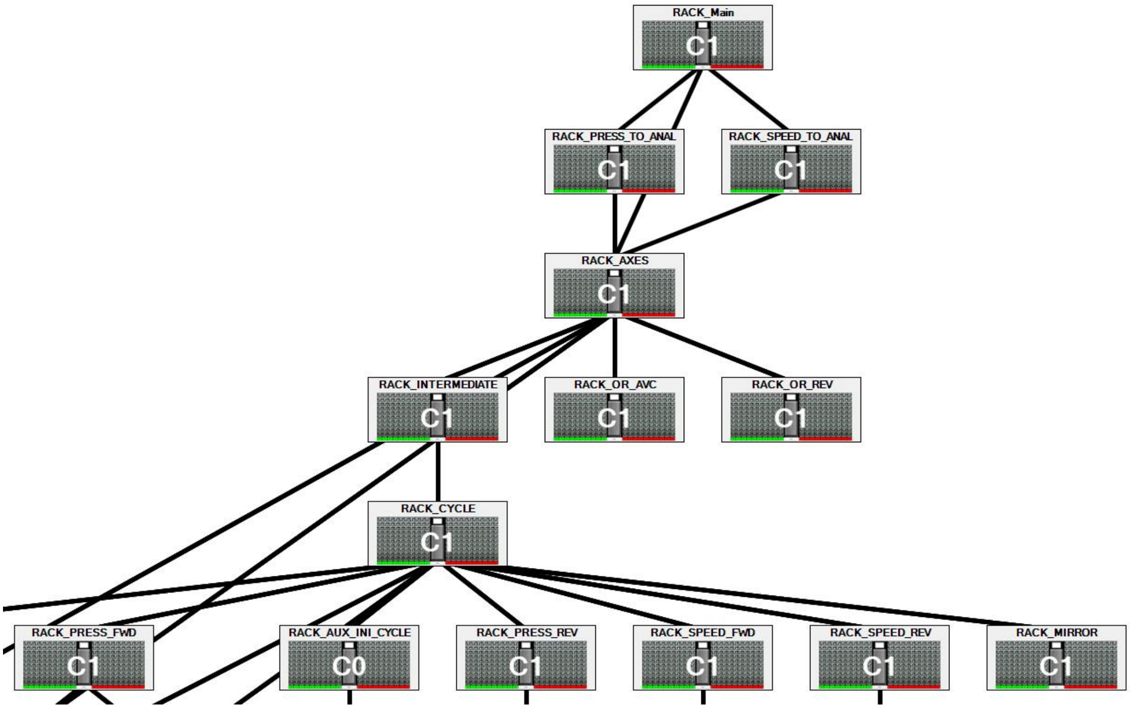
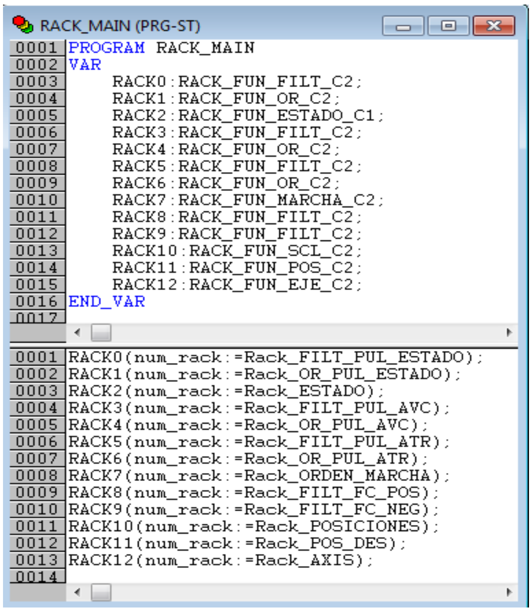


Fig. 10 - Structure of a PLC-PROG program as a hierarchy of interconnected rack components

The hierarchy of racks is converted into a linear sequence of racks before being downloaded into the target PLC (Fig. 11).



```
PROGRAM RACK_MAIN
VAR
    RACK0:RACK_FUN_FILT_C2;
    RACK1:RACK_FUN_OR_C2;
    RACK2:RACK_FUN_ESTADO_C1;
    RACK3:RACK_FUN_FILT_C2;
    RACK4:RACK_FUN_OR_C2;
    RACK5:RACK_FUN_FILT_C2;
    RACK6:RACK_FUN_OR_C2;
    RACK7:RACK_FUN_MARCHA_C2;
    RACK8:RACK_FUN_FILT_C2;
    RACK9:RACK_FUN_FILT_C2;
    RACK10:RACK_FUN_SCL_C2;
    RACK11:RACK_FUN_POS_C2;
    RACK12:RACK_FUN_EJE_C2;
END_VAR
```

```
RACK0(num_rack:=Rack_FILT_PUL_ESTADO);
RACK1(num_rack:=Rack_OR_PUL_ESTADO);
RACK2(num_rack:=Rack_ESTADO);
RACK3(num_rack:=Rack_FILT_PUL_AVC);
RACK4(num_rack:=Rack_OR_PUL_AVC);
RACK5(num_rack:=Rack_FILT_PUL_ATR);
RACK6(num_rack:=Rack_OR_PUL_ATR);
RACK7(num_rack:=Rack_ORDEN_MARCHA);
RACK8(num_rack:=Rack_FILT_FC_POS);
RACK9(num_rack:=Rack_FILT_FC_NEG);
RACK10(num_rack:=Rack_POSICIONES);
RACK11(num_rack:=Rack_POS_DES);
RACK12(num_rack:=Rack_AXIS);
```

Fig. 11 - Linear list of executable racks produce by the postprocessor module

Finally, a library of PLC-PROG components has been produced, including the FBs that appear in the IEC-61131-3 standard (see Fig. 12), as well as a library of technological functions (Fig. 13).

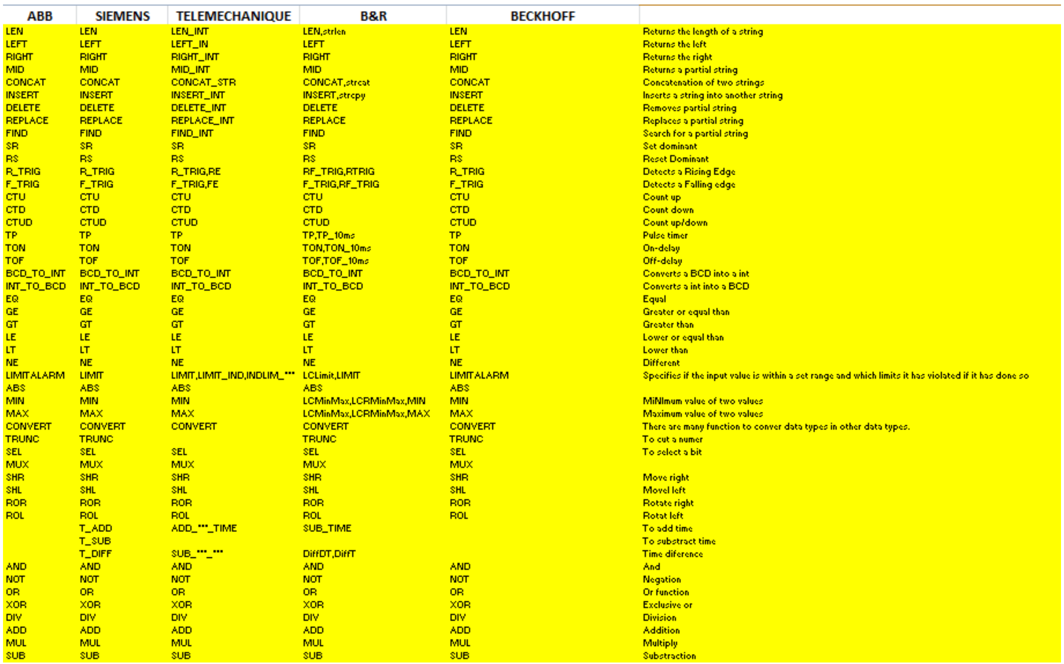


ABB	SIEMENS	TELEMECHANIQUE	B&R	BECKHOFF	
LEN	LEN	LEN_INT	LEN,strlen	LEN	Returns the length of a string
LEFT	LEFT	LEFT_IN	LEFT	LEFT	Returns the left
RIGHT	RIGHT	RIGHT_INT	RIGHT	RIGHT	Returns the right
MID	MID	MID_INT	MID	MID	Returns a partial string
CONCAT	CONCAT	CONCAT_STR	CONCAT,strcat	CONCAT	Concatenation of two strings
INSERT	INSERT	INSERT_INT	INSERT,strcpy	INSERT	Inserts a string into another string
DELETE	DELETE	DELETE_INT	DELETE	DELETE	Removes partial string
REPLACE	REPLACE	REPLACE_INT	REPLACE	REPLACE	Replaces a partial string
FIND	FIND	FIND_INT	FIND	FIND	Search for a partial string
SR	SR	SR	SR	SR	Set dominant
RS	RS	RS	RS	RS	Reset Dominant
R_TRIG	R_TRIG	R_TRIG,RE	RF_TRIG,RTRIG	R_TRIG	Detects a Rising Edge
F_TRIG	F_TRIG	F_TRIG,FE	F_TRIG,RF_TRIG	F_TRIG	Detects a Falling edge
CTU	CTU	CTU	CTU	CTU	Count up
CTD	CTD	CTD	CTD	CTD	Count down
CTUD	CTUD	CTUD	CTUD	CTUD	Count up/down
TP	TP	TP	TP,TP_10ms	TP	Pulse timer
TON	TON	TON	TON,TON_10ms	TON	On-delay
TOF	TOF	TOF	TOF,TOF_10ms	TOF	Off-delay
BCD_TO_INT	BCD_TO_INT	BCD_TO_INT	BCD_TO_INT	BCD_TO_INT	Converts a BCD into a int
INT_TO_BCD	INT_TO_BCD	INT_TO_BCD	INT_TO_BCD	INT_TO_BCD	Converts a int into a BCD
EQ	EQ	EQ	EQ	EQ	Equal
GE	GE	GE	GE	GE	Greater or equal than
GT	GT	GT	GT	GT	Greater than
LE	LE	LE	LE	LE	Lower or equal than
LT	LT	LT	LT	LT	Lower than
NE	NE	NE	NE	NE	Different
LIMITALARM	LIMIT	LIMIT,LIMIT_IND,INDLIM_***	LCLimit,LIMIT	LIMITALARM	Specifies if the input value is within a set range and which limits it has violated if it has done so
ABS	ABS	ABS	ABS	ABS	
MIN	MIN	MIN	LCMinMax,LCRMinMax,MIN	MIN	MiNImum value of two values
MAX	MAX	MAX	LCMinMax,LCRMinMax,MAX	MAX	Maximum value of two values
CONVERT	CONVERT	CONVERT	CONVERT	CONVERT	There are many function to conver data types in other data types.
TRUNC	TRUNC		TRUNC	TRUNC	To cut a numer
SEL	SEL	SEL	SEL	SEL	To select a bit
MUX	MUX	MUX	MUX	MUX	
SHR	SHR	SHR	SHR	SHR	Move right
SHL	SHL	SHL	SHL	SHL	Movel left
ROR	ROR	ROR	ROR	ROR	Rotate right
ROL	ROL	ROL	ROL	ROL	Rotat left
	T_ADD	ADD_***_TIME	SUB_TIME		To add time
	T_SUB				To substract time
	T_DIFF	SUB_***_***	DiffDT,DiffT		Time diference
AND	AND	AND	AND	AND	And
NOT	NOT	NOT	NOT	NOT	Negation
OR	OR	OR	OR	OR	Or function
XOR	XOR	XOR	XOR	XOR	Exclusive or
DIV	DIV	DIV	DIV	DIV	Division
ADD	ADD	ADD	ADD	ADD	Addition
MUL	MUL	MUL	MUL	MUL	Multiply
SUB	SUB	SUB	SUB	SUB	Substraction

Fig. 12 - IEC-61131-3 functions that have been ported to the PLC-PROG system

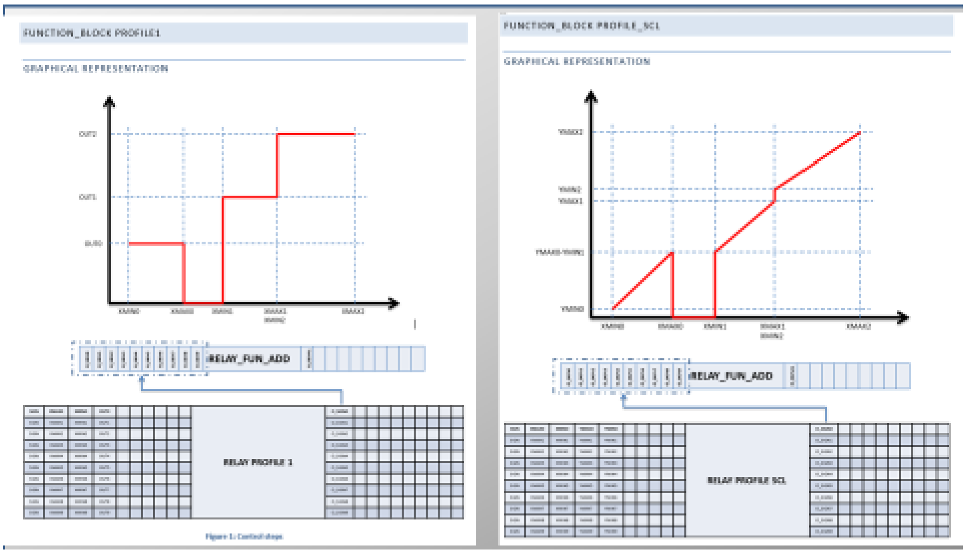
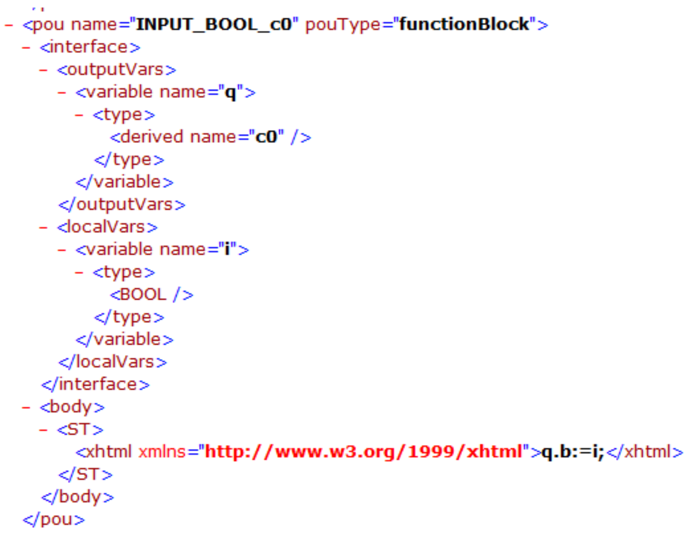


Fig. 13 - Example of technological functions that are part of the PLC-PROG library

It is worth to mention that all the elements of PLC-PROG are persisted using standard XML language (Fig. 14).



```
- <pou name="INPUT_BOOL_c0" pouType="functionBlock">
  - <interface>
    - <outputVars>
      - <variable name="q">
        - <type>
            <derived name="c0" />
          </type>
        </variable>
      </outputVars>
    - <localVars>
      - <variable name="i">
        - <type>
            <BOOL />
          </type>
        </variable>
      </localVars>
    </interface>
  - <body>
    - <ST>
        <xhtml xmlns="http://www.w3.org/1999/xhtml">q.b:=i;</xhtml>
      </ST>
    </body>
  </pou>
```

Fig. 14 - File containing a PLC-PROG element, using standard XML language

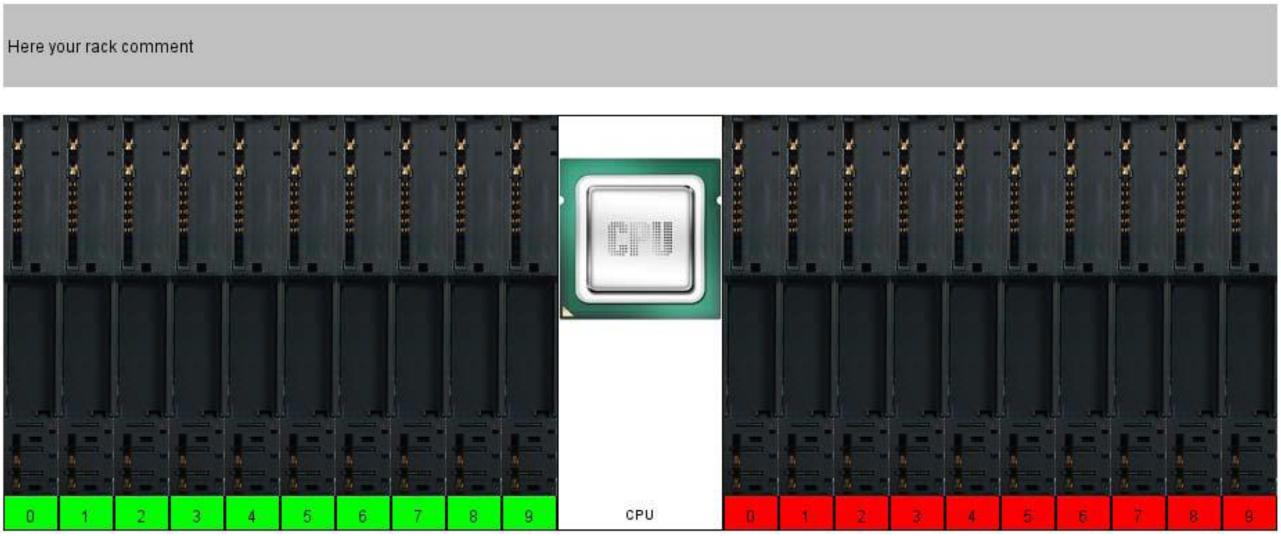
What is a Rack?

A rack is a PLC-PROG object that encapsules a functionality and its configuration. The functionality itself is encapsulated in a function, and the configuration is set using inputs and outputs modules.

A rack always maintains the same structure and the user just need to worry about the configuration. The configuration is done by dragging and dropping modules. The rest is done automatically by the software.

The rack is composed of:

- 10 input slots (0 to 9 in green)
- 10 output slots (0 to 9 in red)
- 1 Function slot (cpu area)



Here your rack comment

Fig. 15 - PLC-PROG rack structure

3.3. PLC-PROG IDE

The objectives of the project are mainly the development of a visual integrated development environment (IDE) integrating object oriented (OO) features. It must contains tools to facilitate all kind of users (experts and non-experts) to program easy and complex PLC applications. The result is PLC-PROG: a software to program PLCs. It contains a visual editor based on the concept of rack and a set of tools to help the user when creating the program.

The Graphical User Interface integrates a Debugger Tool that allows to monitor and to modify all of the PLC values during the PLC cycle. This tool is extremely useful for debugging the programs and commissioning the installation; its environment is graphical and presents the same aspects of the GUI.

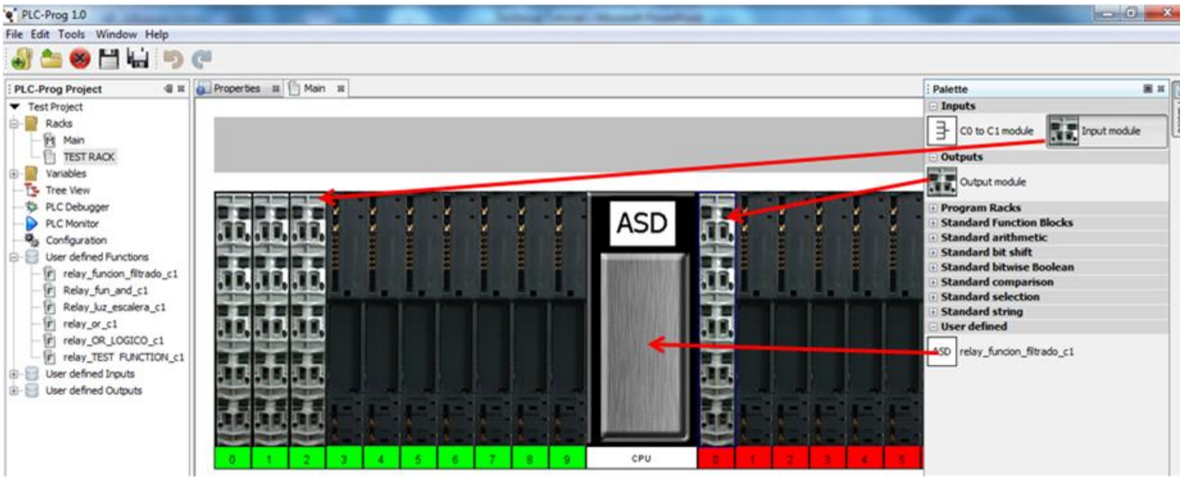


Fig. 16 - PLC-PROG IDE overview

PLC Debugger

It is a separate executable tool that connects directly to the OPC server. It contains all the variables of the project and permits the user to debug the program by reading and writing values online. To make it easier, it maintains the same aspect as in the IDE. The Debugger is a ".net Object" as an “ActiveX Object”. This object is able to operate alone or embedded in commercial SCADA System. It offers a consolidated, identical view in all of them.

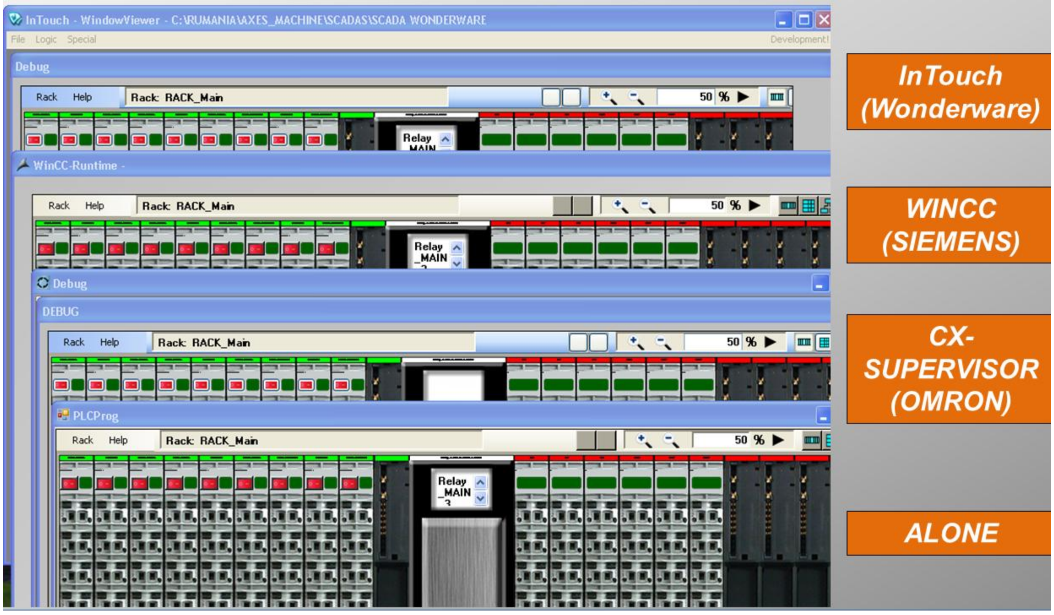


Fig. 17 - Debugger like an Object embedded in Commercial SCADA System or running alone

It is connected with the PLC using OPC technology. When the post-processing is successful a file with the extension ".debug" is created. This file contains the tags with the addresses of the variables that are being used for communicating between the debugger and the PLC.

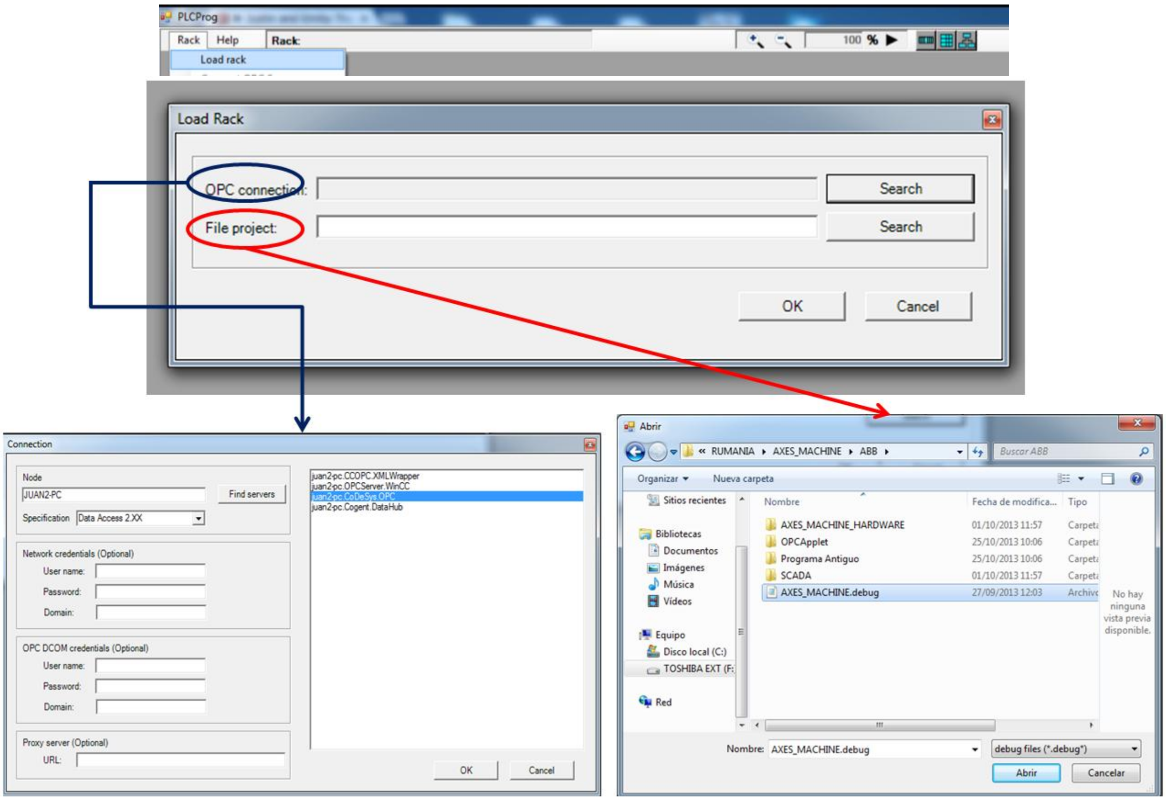


Fig. 18 - Debugger: Load a Project and OPC Server

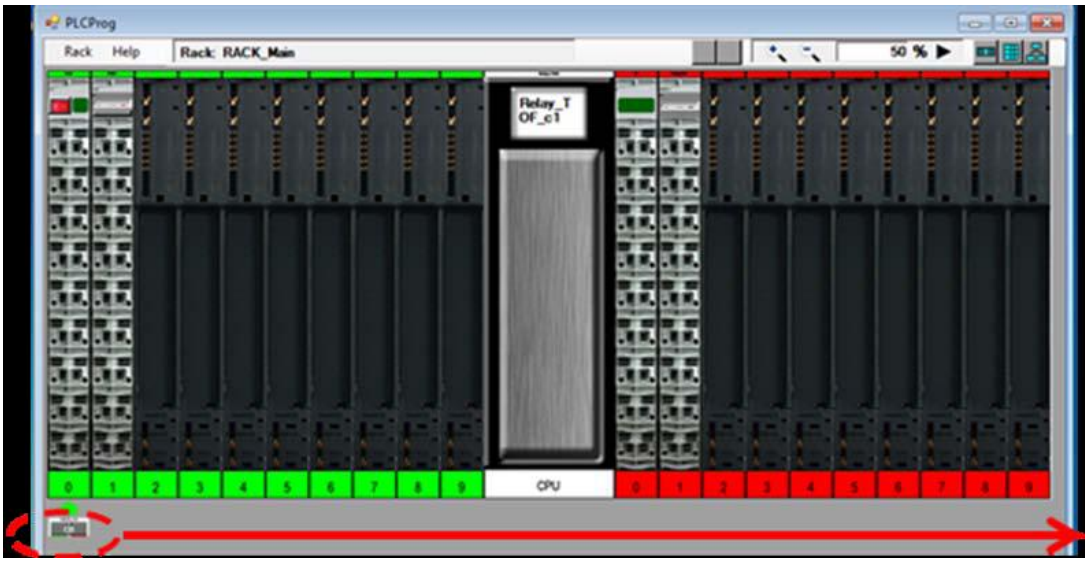
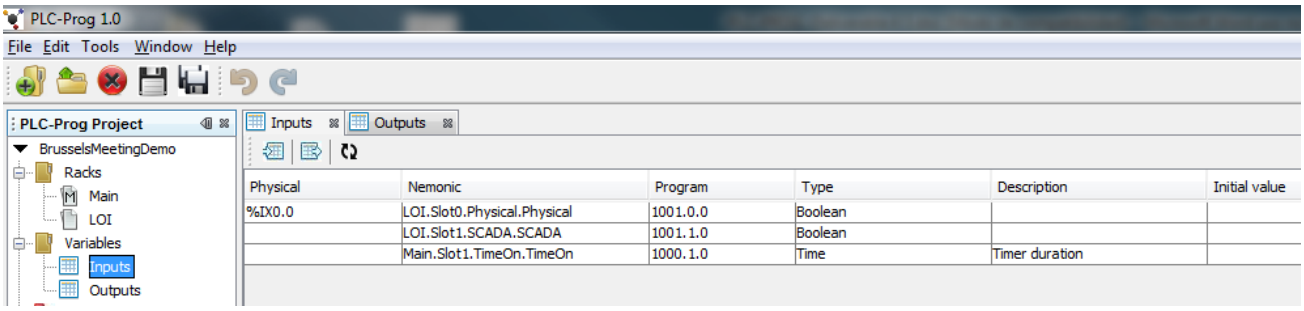


Fig. 19 - PLC Debugger

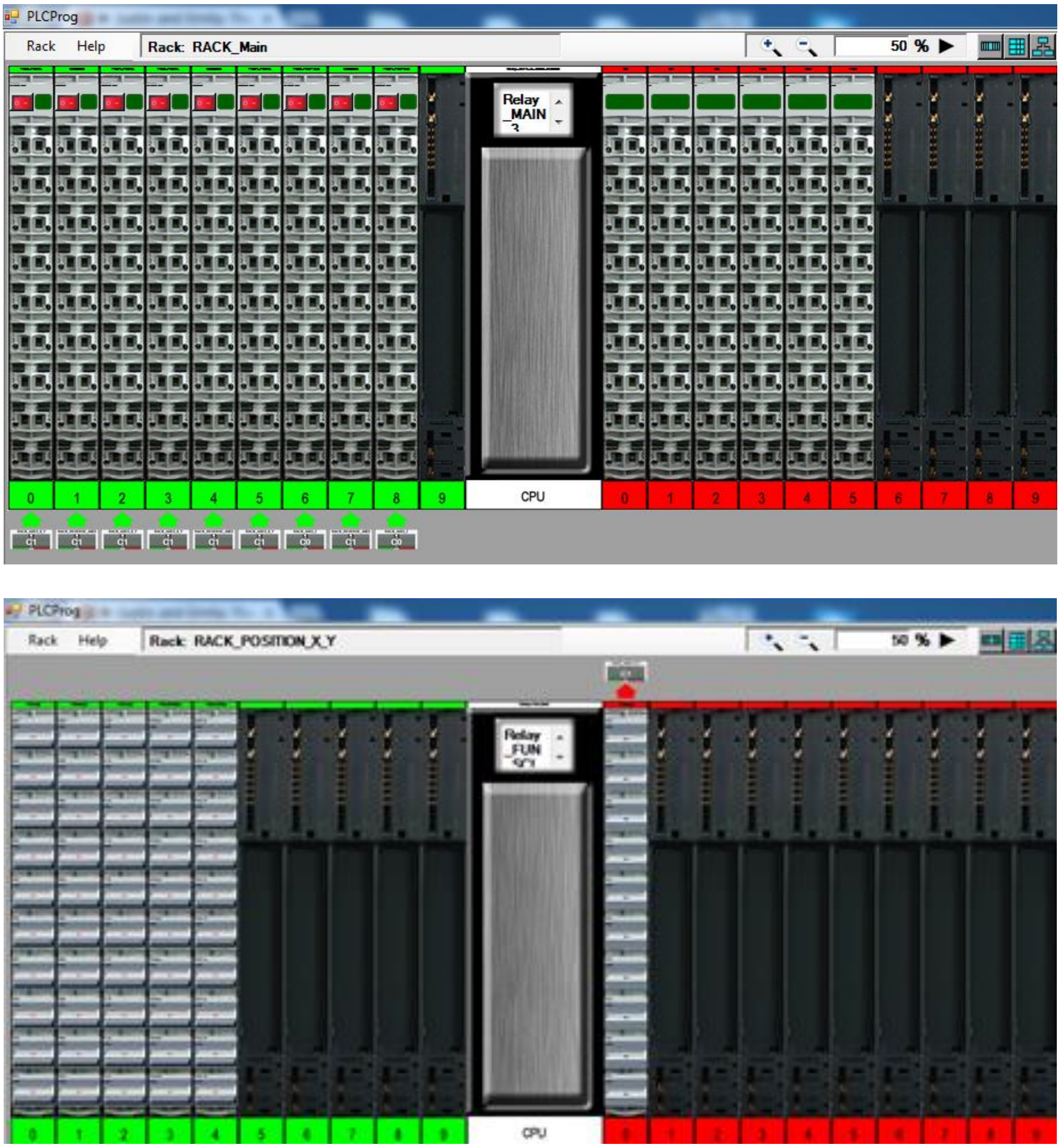
The Debugger has three different **Views** of the project:

- **Rack View**: this view shows the complete rack, in the same way that in the IDE. On the central part of the rack it is displayed the function that performs the rack's logic, called CPU. In the left, there are the inputs of the rack, sort by slots (maximum 10). And in the right part of the CPU there are the outputs of the rack, also sorted in slots (maximum 10). These slots are function of the type of the rack, C0 is one variable for Slot, C1 are ten variables for slots, etc. These variables represent the input and output that PLC-PROG creates automatically for the user who can configure physical addresses for inputs and outputs for an specific PLC brand. The variables can be of any type, boolean, integer, real or time, because they are encapsulate in c0 polymorphic conductors.



Physical	Nemonic	Program	Type	Description	Initial value
%IX0.0	LOI.Slot0.Physical.Physical	1001.0.0	Boolean		
	LOI.Slot1.SCADA.SCADA	1001.1.0	Boolean		
	Main.Slot1.TimeOn.TimeOn	1000.1.0	Time	Timer duration	

Fig. 20 - Variables table



Rack: RACK_Main

Rack: RACK_POSITION_X_Y

Fig. 21 - Debugger Racks View

On the top and at the bottom borders of the Rack, there are arrows in the case that these cards are connected to other racks' inputs, so indicating the flow of the program. For examples, a small arrow at the bottom of an input card mean the values of this card are fed by the outputs of another rack, to whom we can navigate.

Besides, an extra view that is included in the Debugger is the possibility of working with a zoom of the slots.

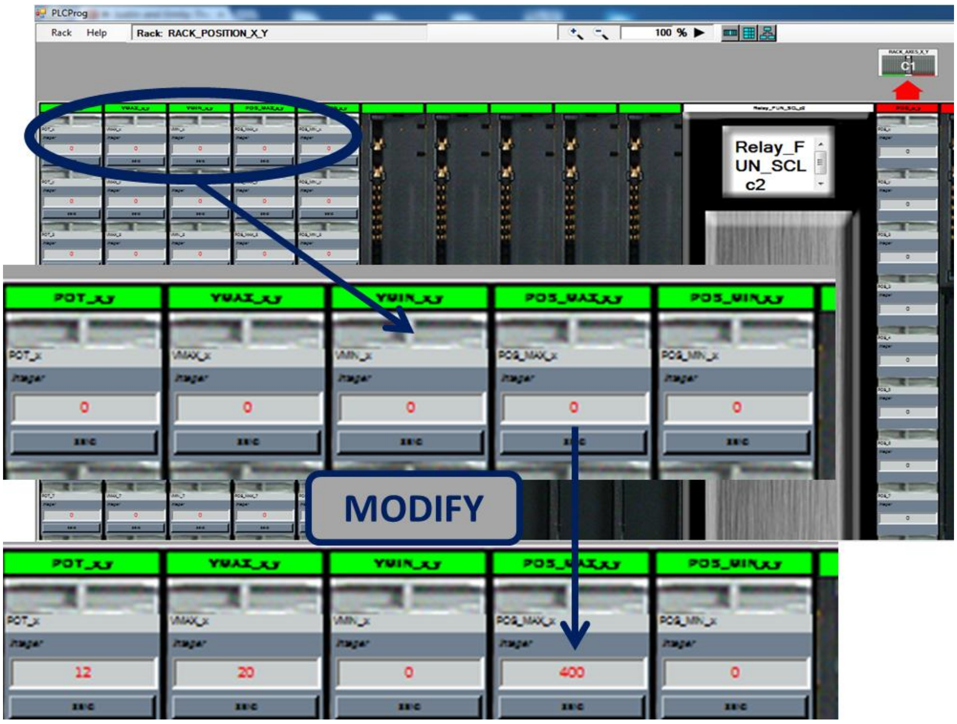
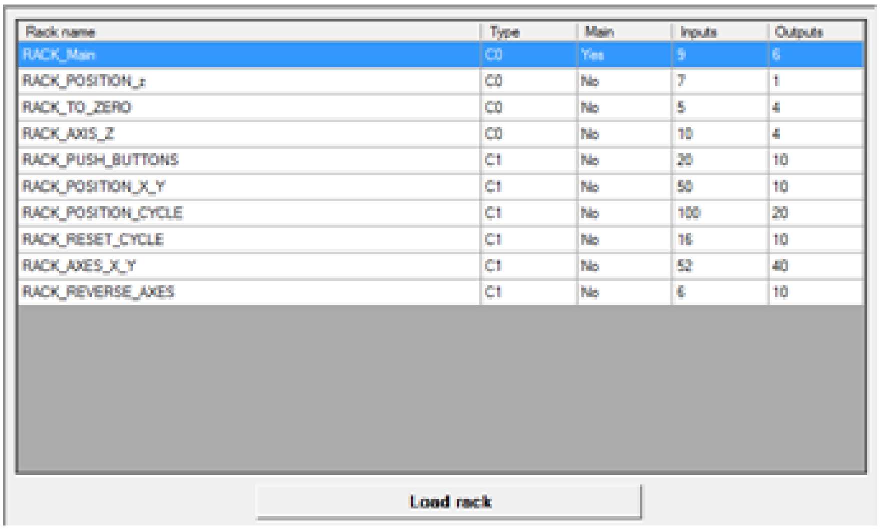


Fig. 22 - Debugger Zoom Tool

- **List View:** this view shows a table with every rack contained in the Project. The main characteristics are: the type of the connection (C0, C1,..) the number of inputs, the number of outputs and which is the main rack in this project.



Rack name	Type	Main	Inputs	Outputs
RACK_Main	C0	Yes	9	6
RACK_POSITION_z	C0	No	7	1
RACK_TO_ZERO	C0	No	5	4
RACK_AXIS_Z	C0	No	10	4
RACK_PUSH_BUTTONS	C1	No	20	10
RACK_POSITION_X_Y	C1	No	50	10
RACK_POSITION_CYCLE	C1	No	100	20
RACK_RESET_CYCLE	C1	No	16	10
RACK_AXES_X_Y	C1	No	52	40
RACK_REVERSE_AXES	C1	No	6	10

Fig. 23 - Debugger List View

- **Tree View:** this view shows the hierarchical structure of a PLC-PROG program in tree form, displaying the connection between racks, functions and variables in the project. Any of the rack can be accessed and opened from this view. This view is very comfortable because it allows navigating inside of the project.

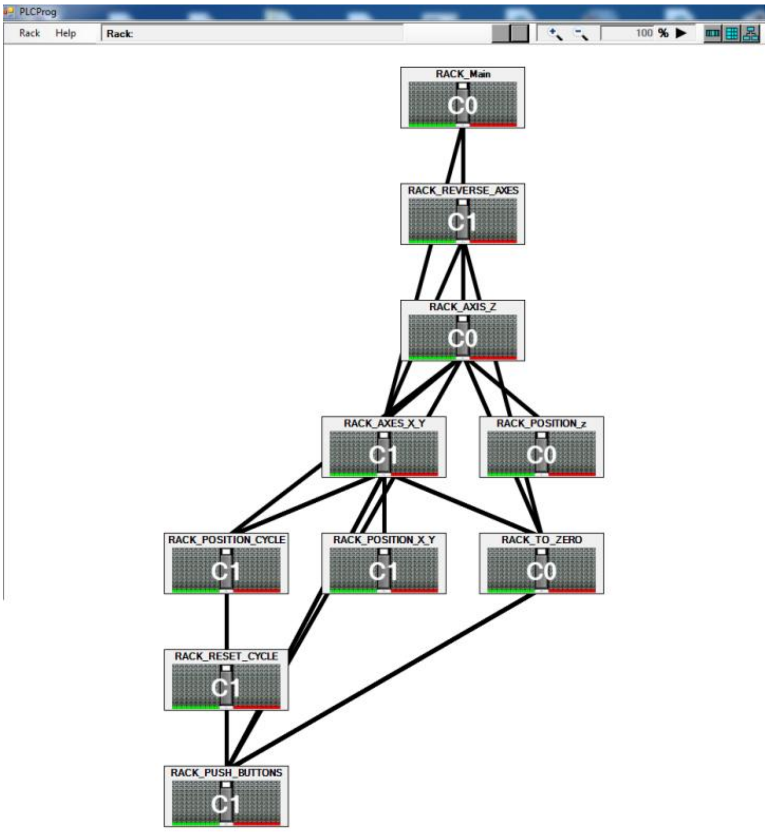


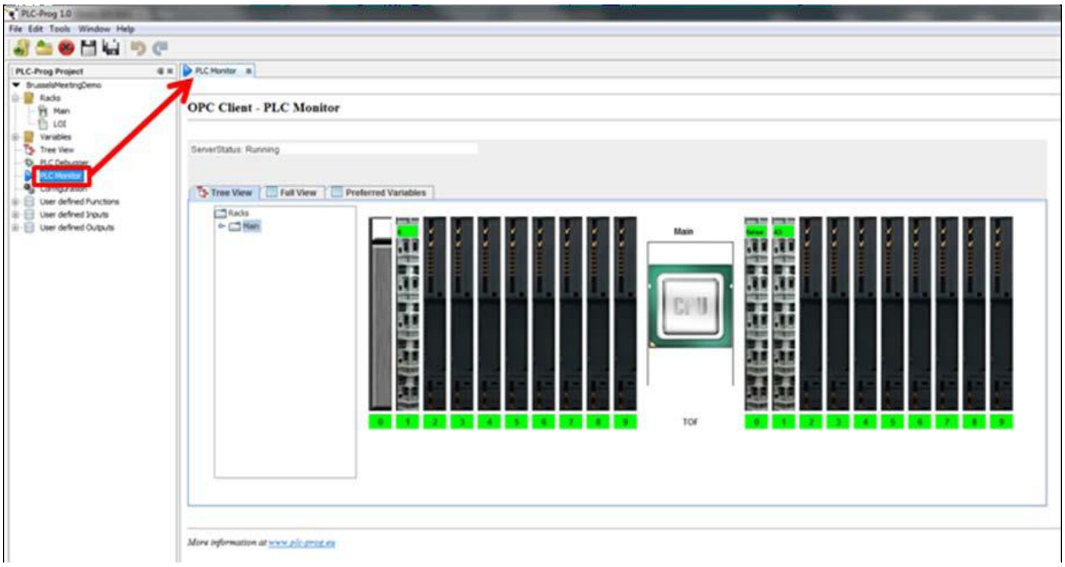
Fig. 24 - Debugger Tree View

Finally, the Debugger is a fundamental tool in PLC-PROG, allowing:

- to debug the control program;
- to monitor and to change all of the variables like a SCADA system.

PLC Monitor

This tool connects directly to the OPC server that contains all the variables of the project using OPC UA or DA protocol. It permits the user to monitor the program online by reading and writing values from any web browser. To make it easier, it maintains the same aspect as in the IDE.

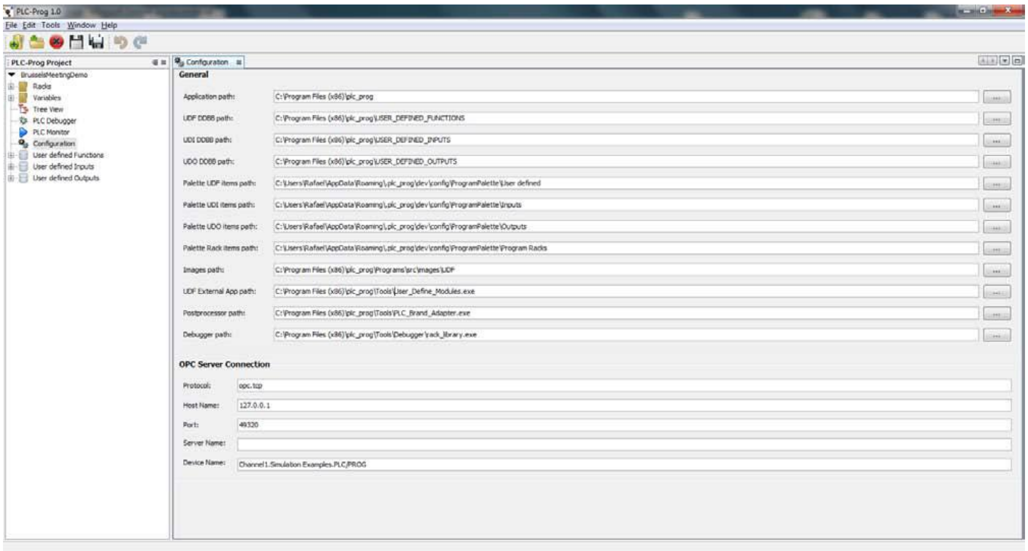


ServerStatus: Running

Fig. 25 - PLC Monitor

Configuration

This section permits the user to change the configuration settings related to OPC connection and internal paths of the software.



General

Application path: C:\Program Files (x86)\plc_prog
UDF DDBB path: C:\Program Files (x86)\plc_prog\USER_DEFINED_FUNCTIONS
UDI DDBB path: C:\Program Files (x86)\plc_prog\USER_DEFINED_INPUTS
UDO DDBB path: C:\Program Files (x86)\plc_prog\USER_DEFINED_OUTPUTS
Palette UDF items path: C:\Users\Rafael\AppData\Roaming\.plc_prog\dev\config\ProgramPalette\User defined
Palette UDI items path: C:\Users\Rafael\AppData\Roaming\.plc_prog\dev\config\ProgramPalette\Inputs
Palette UDO items path: C:\Users\Rafael\AppData\Roaming\.plc_prog\dev\config\ProgramPalette\Outputs
Palette Rack items path: C:\Users\Rafael\AppData\Roaming\.plc_prog\dev\config\ProgramPalette\Program Racks
Images path: C:\Program Files (x86)\plc_prog\Programs\src\images\UDF
UDF External App path: C:\Program Files (x86)\plc_prog\Tools\User_Define_Modules.exe
Postprocessor path: C:\Program Files (x86)\plc_prog\Tools\PLC_Brand_Adapter.exe
Debugger path: C:\Program Files (x86)\plc_prog\Tools\Debugger\rack_library.exe

OPC Server Connection

Protocol: opc.tcp
Host Name: 127.0.0.1
Port: 49320
Server Name:
Device Name: Channel1.Simulation Examples.PLC/PROG

Fig. 26 - Configuration tab

User defined Functions/Inputs/Outputs

These sections have been designed to maintain the flexibility of the PLC programmers' needs. These sections permit to create closed, debugged functions to be used in the Racks program just by configuring them. These tools permit the migration of existing projects to PLC-Prog. These are complex sections designed to be used by PLC programmers only.

User defined Functions Editor

The main rack is the starting point where the users create the program that need to run in an automated process. This program is full of smaller pieces of code grouped into other racks containing functions. Every function is written to execute an action or to have a specific behaviour depending on some parameters.

PLC-Programmers can use the main program and create other racks as well, but it is important for them to define and create functions and to facilitate to parameterize functions that can later be used in the main program and racks by end-users.

This section focuses on the creation of User Defined Functions (UDF). These functions are complex functions created based on IEC Standard functions to accomplish a specific task. Some functions can even include other UDF's to create even more complex behaviours.

Most of the automation processes will need different functions with very similar behaviour. This is where the importance of parameters emerges. A single function can slightly change its response if it has been programmed to react to different input values. The PLC-Programmers decide how many parameters a function needs, the type of function needed and the way the function can be parameterized.

When a UDF is created and debugged, it can be included in the palette with the rest of the functions and can be used in the program racks or even in the definition of a more complex UDF.

This tool also permits the creation of input and output modules. The behaviour of these modules is exactly the same as of the Standard modules just that its name and type are predefined.

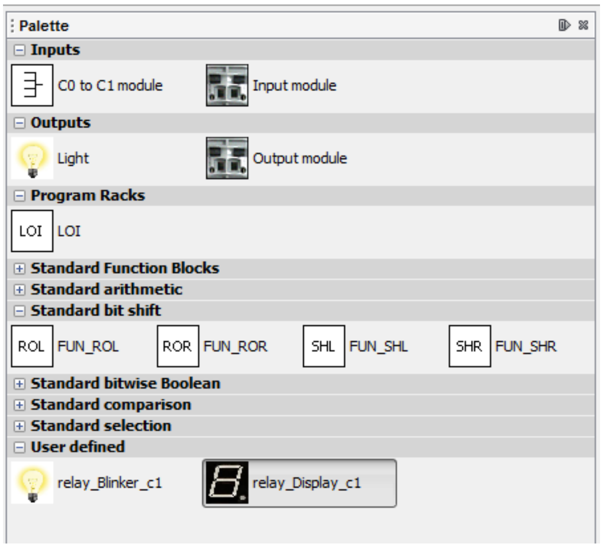
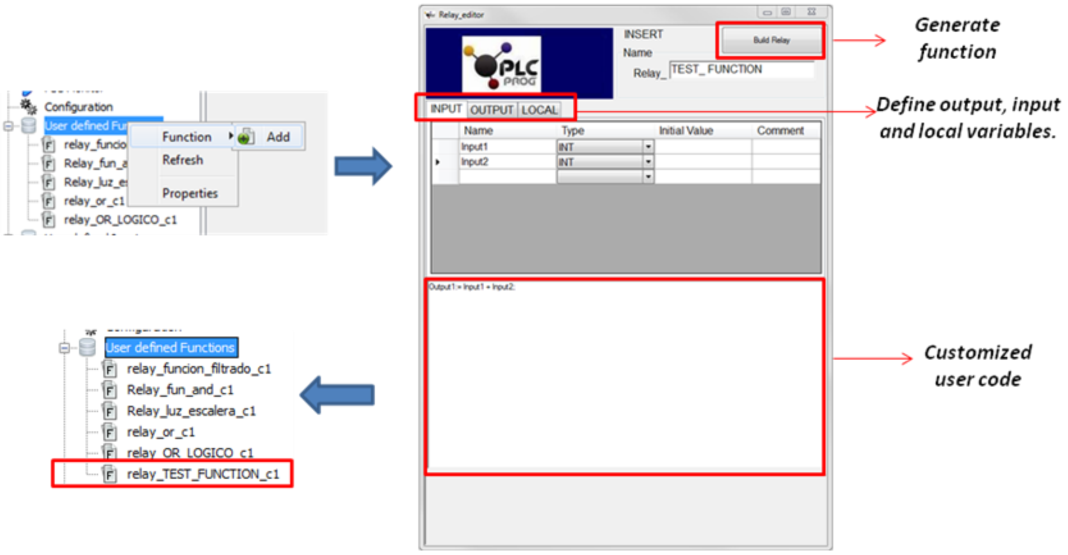


Fig. 27 - Palette filled with Standard and User Defined Functions

The following picture shows the User Defined Functions editor:



Relay_ TEST_ FUNCTION

Name	Type	Initial Value	Comment
Input1	INT		
Input2	INT		

```
Output1:= Input1 + Input2;
```

Fig. 28 - Create a new User Defined Function

The Function Editor tool contains three tabs to define Input, Output and Local variables, respectively. The user can define up to ten variables per tab. The lower part of the editor is the part where the user can insert the programming code. The editor accepts ST language code (ST language is defined in the IEC-61131-3).

Moreover, any piece of code written in ST language can be included in a function. It has the following pros because the user is able to:

- Create new functions from scratch.
- Reuse code from other User Defined Functions.
- Import any external code or function written in ST language and encapsulate it in a PLC-PROG object.

The editor creates an xml file with the definition of the function and its encapsulation in a PLC-PROG object.

The Function editor tool also permits to create Input and Output predefined modules. The purpose of the creation of these modules is to be reused. If there is a type of variable definition that is repetitive, a User Defined Input or Output reduces the definition time considerably (about 50%).

Inside PLC-PROG

PLC-PROG IDE has been developed having in mind the expansion, further development, and with the possibility of being multiplatform. This is why Java as programming language has been selected.

Moreover, a methodology for programming have been used to ease the further work. Netbeans RCP (Rich Client Platform) core and libraries have been used for development, and a modular structure have been used. Any extra capability for PLC-PROG can be easily introduced by using modules or plugins.

3.4. SCADA approach

The development of any industrial application cannot exempts to provide the user with an interface for monitoring the activity and control it; these operations are usually done by a SCADA application. PLC-PROG's aim it not that of offering a complete SCADA system, for at least two reasons:

1. SCADA systems are very complex and the development of one complete and usable application is far from our possibility and our scope in PLC-PROG.
2. Each user already has a SCADA system, and our objective is absolutely not to replace this system but to integrate our work with the existing. In fact, the industrial environment is less flexible respect to some other sectors, so the only opportunity we identified for PLC-PROG system to gain a portion of this market is to act for a soft introduction, without revolutions.

In order to offer the "integration with the already on the market SCADA systems", we must control our portion of code (running into a PLC) with something that could be integrated.

At the beginning of the project the only approach that we had identified for its novelty was that of working with ActiveX, whist during the project implementation, other possible solutions appeared to be more suitable due the introduction of the PLC UA architecture in the market and the new functionalities that appeared to be present in the new generations of PLCs.

During the proceedings of the project a new OPC-UA protocol appeared pm the market. This new protocol tries to switch the intercommunication carrier (between PCs and PLCs) from DCOM (Microsoft's technology, proprietary and so old to be deprecated by Microsoft itself since the appearance of Window Vista in the 2007) to sockets, the standard carrier for all internet communications (open, standard and cross platform).

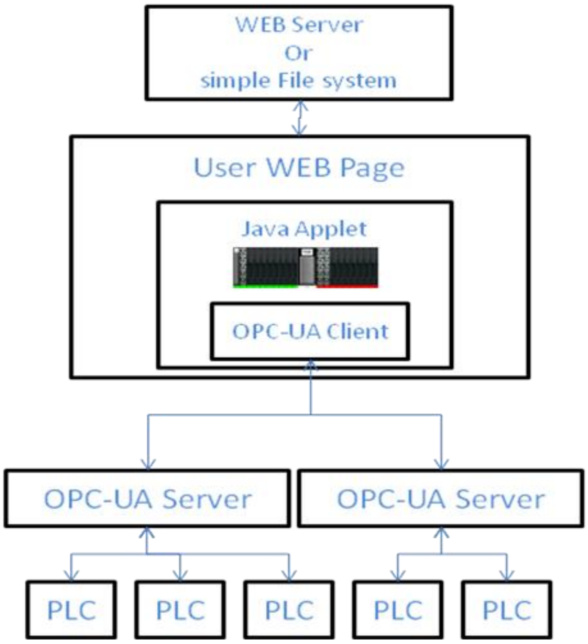
The initial approach to develop a set of ActiveX (one for each base function on the IEC 61131) and then build in an automatic way the SCADA page that contains the needed components.

Each of those ActiveX components should use the SCADA OPC-DA communication engine to get/set process information from/to PLC's server.

This approach demonstrated great limits, in particular for what concerns the automatic page generation: the only available way to realize that task was use the ATL interfaces that some producer "offer" in its owns SCADA packet.

Those ATL interfaces are handles for controlling SCADA graphical designer but are developed for internal use only (for this "offer" was between quotes), so they are:

- ✓ Mainly undocumented.
- ✓ Different for each producer
- ✓ Not even available
- ✓ Continuity and compatibility of the ATL are not guarantee



So the approach resulted hard to be applied to a reasonable set of SCADA products and with unacceptable uncertainties for future availability.

For these reasons we modified the approach upside-down: instead of developing a set of ActiveX and use the communication engine of the SCADA system, we created a single Java Applet (with its own communication engine, but this time OPC-UA) and used a simple HTML to display all in a standard browser (or an Internet Explorer ActiveX) included in a SCADA page.

This way the resulting product allowed to achieve a number of advantages:

- It's "much simpler" to realize and maintain
- The Applet can be run either locally or remotely.
- It doesn't need a web server but it can be introduced into a Web site.
- It's simple to be configured.
- The Applet can be used directly and immediately as a "inline debugger".
- It doesn't introduce communication delays.

- It’s usable in (virtually) any device and any OS.
- Requirements on the end devices are acceptable (only a Sun-Java engine is required).
- A Web page can be introduced in practically any existing SCADA system.
- To use the applet into a SCADA system is very simple.

But also some disadvantages:

- It offer a limited grade of graphical customizations: the graphic is inside the Applet.
- Commercially available SCADA applications will not be able to read UA data directly from the Applet.

The new prospective permits to collapse this enormous set of micro-functions (that are not really useful for the SCADA user) in Racks like the IDE view. Each Rack represents a Function Block in a high level language where Inputs and Outputs lose dimension and type to became “entities”, “events” and “messages”.

In this way, SCADA and IDE views overlap and the use of both instruments becomes easy.

Web applets work like China’s boxes:

- The web browser loads the HTML file that is substantially a programming code
- The HTML commands the browser to reserve a portion of its window to the Java engine
- The HTML informs the Java engine where the Applet to load is located

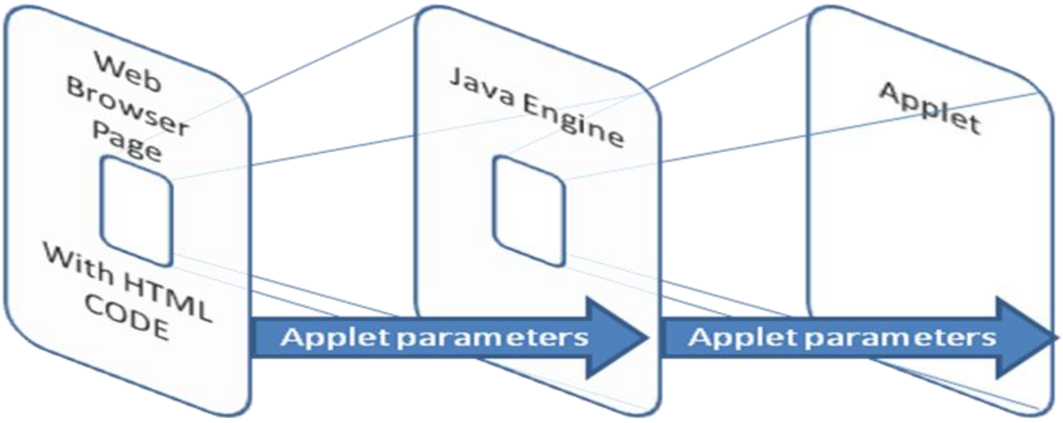


Fig. 29 – WEB Applet workflow

Our system works with some additional complications:

1. On the left side (at highest level) we introduce an ActiveX that contains an Internet Explorer Web Browser object like a transparent frame. The reason will be clearer in the following sections, but fundamentally was for simplifying the beginners on the IDE usage.
2. On the bottom side our Applet communicates with the PLC, reads and writes data and displays the SCADA page.

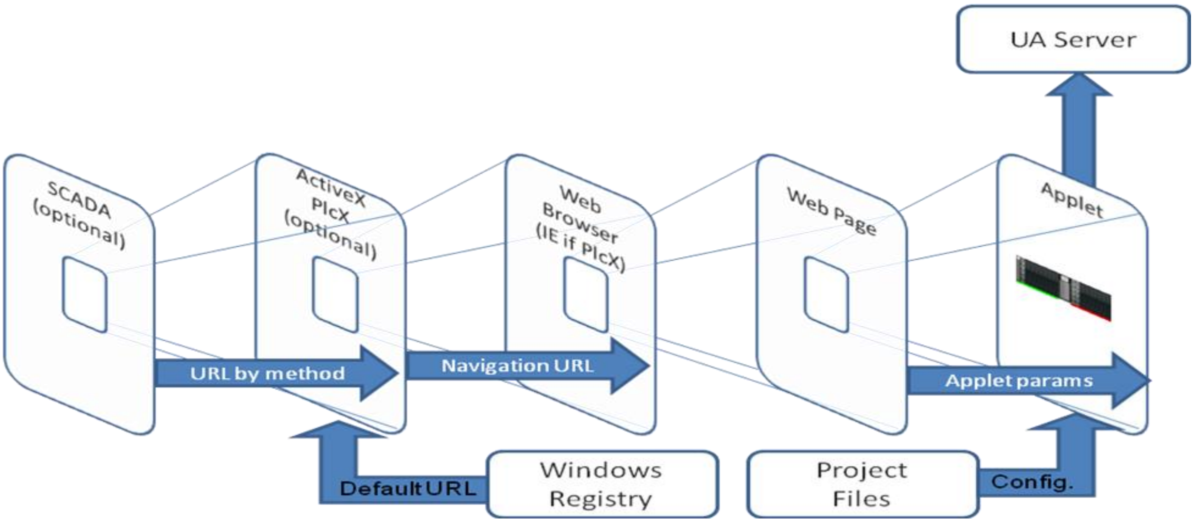


Fig. 30 – Applet and SCADA communication flow

Despite to this apparent complication, the use of the ActiveX and the Applet is simple like clicking an hyperlink. To show it into a Browser's page, simple double-click the HTML that you will find into the "exported SCADA folder" and this will be the result:

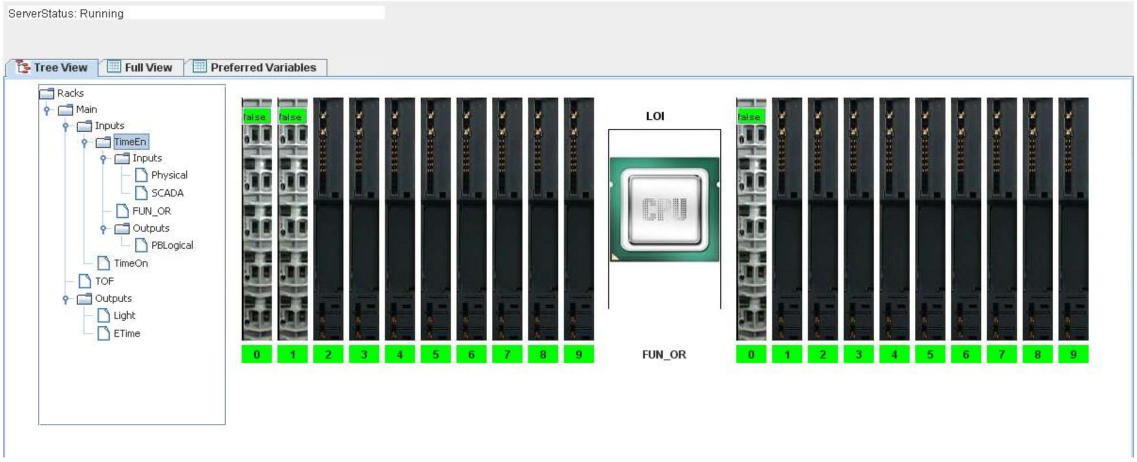

ServerStatus: Running

Fig. 31 – Browser's view

At the same way, to show it into any kind of SCADA system in projecting mode, it is needed to simply introduce the desired page into the PlcX component and, then, resize it as required.

3.5. Versioning (or revision control system)

"Software versioning is the process of assigning either unique version names or unique version numbers to unique states of computer software" (Wikipedia).

In other words *"… it is the management of changes to documents, computer programs, large web sites, and other collections of information. Changes are usually identified by a number or letter code, termed the "revision number", "revision level", or simply "revision". For example, an initial set of files is "revision 1". When the first change is made, the resulting set is "revision 2", and so on. Each revision is associated with a timestamp and the person making the change. Revisions can be compared, restored, and with some types of files, merged"* (Wikipedia).

We need a versioning system for administer the multiply version of each User Defined Function (UDF) that the user can create/modify.

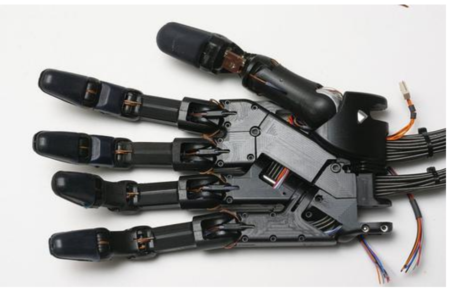
Those UDF are functions created by the user for solve a complex problem starting from a more simple set of objects and are great resources for future projects and for this reason they must be exportable, should be roll-able back.

To explain what UDF can be, we can think to an axes control made by:

- One engine control;
- One position measure;
- Two proximity sensors.

With those basic component you can create the Axis Control UDF.

Joining 5 Axis Control UDF component the user can create an Arm Control UDF.



…or modify and complicate it for other scopes.

A versioning system is usually designed to work in one of these two ways:

1. distributed, with a central manager owning the Database and a set of client… (Subversion-like)
2. locally, with all the information needed into the data packet… (Word revision-like)

The PLC-PROG system developed in the project applies a different and absolutely unusual approach: it works locally, has single/multiply export capabilities but has no full data inside the data packet.

Moreover, the versioning system is window-less (the interface was the IDE program) and a communication mechanism was implemented.

We solved the problem dividing the problem in two pieces, the main folder that contains all UDF with all their versions and a working directory set, loaded by the UDF versions used in the opened project. When the user opens a project, the IDE puts it into a specified folder (surveyed by the versioning system); inside the project file there are all UDF's names and relative version so the versioning is able to fill the working folders with the right UDFs and permits the IDE to use it.

The versioning module surveys the working set directory continuously and responds to commands (specific files into a specific folder), modification of the existing files (for modified UDFs) and appearance of new files for import operations and new UDF.

These are the principal functions implemented and how they work:

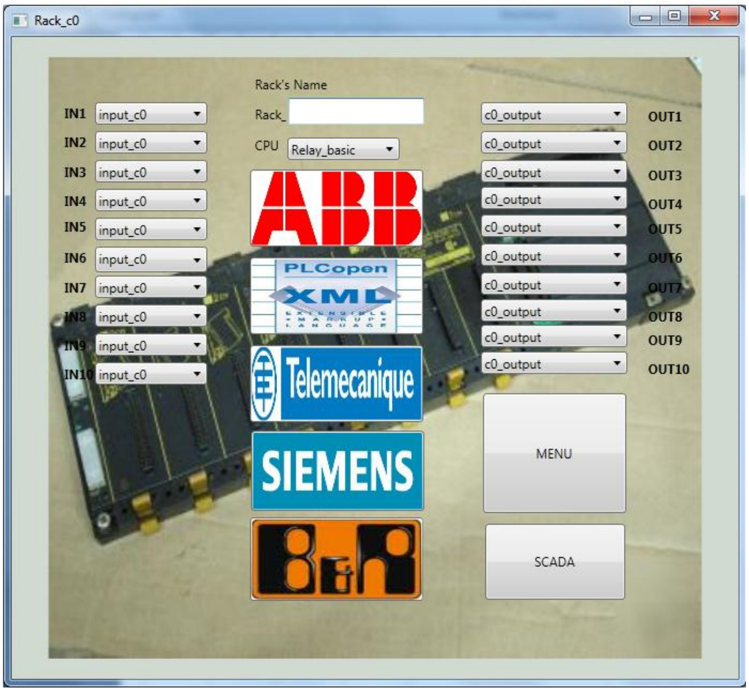
- roll-back some UDF (IDE asks the versioning to do that with a command)
- modify an existing one (the versioning notice the change and creates a new version for the modified UDF into the main DB)
- add new one (the versioning notice the new UDF and insert it into the main DB)
- export part or all the main DB (IDE asks the versioning to do that with a command)
- import a set of UDF (exactly like can add new one).

3.6. Post-processors for three targeted major commercial brands

The post-processors are the coworkers in the dark. This subsystem is integrated in the Graphical User Interface and works in the background to facilitate the work of the end-user.

The post-processors are responsible for translating a PLC-PROG PLC program, which is fully compatible with the IEC-61131-3 standard, into a specific, non-fully compatible code, as needed by any of the target PLC brands in the market. It has been developed in order to convert the standard code produce by PLC-PROG tool into the native code for three specific brands of PLC: 'Step 7' of Siemens, 'SoMachine' of Telemecanique, and 'Codesys V2.3' of ABB, as well as two additional brands have been added in respect to the preliminary chosen ones, (namely: B&R and PLCOpen XML) since they are actually in use among some of the partners.

This application has been checked satisfactory with the 3 pilot's plants.



Rack_c0

Rack's Name

Rack_

CPU Relay_basic

IN1 input_c0
IN2 input_c0
IN3 input_c0
IN4 input_c0
IN5 input_c0
IN6 input_c0
IN7 input_c0
IN8 input_c0
IN9 input_c0
IN10 input_c0

c0_output OUT1
c0_output OUT2
c0_output OUT3
c0_output OUT4
c0_output OUT5
c0_output OUT6
c0_output OUT7
c0_output OUT8
c0_output OUT9
c0_output OUT10

Fig. 32 – PLC brands compliant with the post-processor

Moreover, other application has been developed in order to convert the code produced by PLC-PROG tool into native code for the Commercial SCADA Systems

In the Post-Processors there is a common part, and a specific part that depends of the PLC brand. In the PLC-PROG methodology, the PLC brand does not affect to the control program, which is the same for any PLC brand before post-processing.

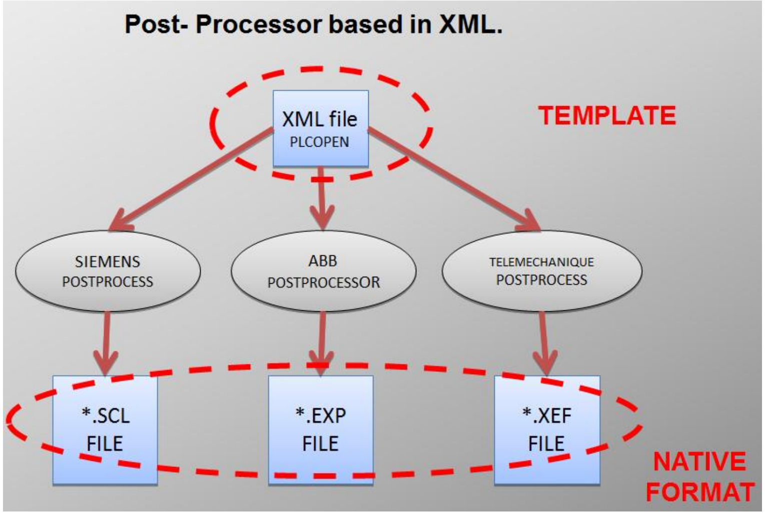
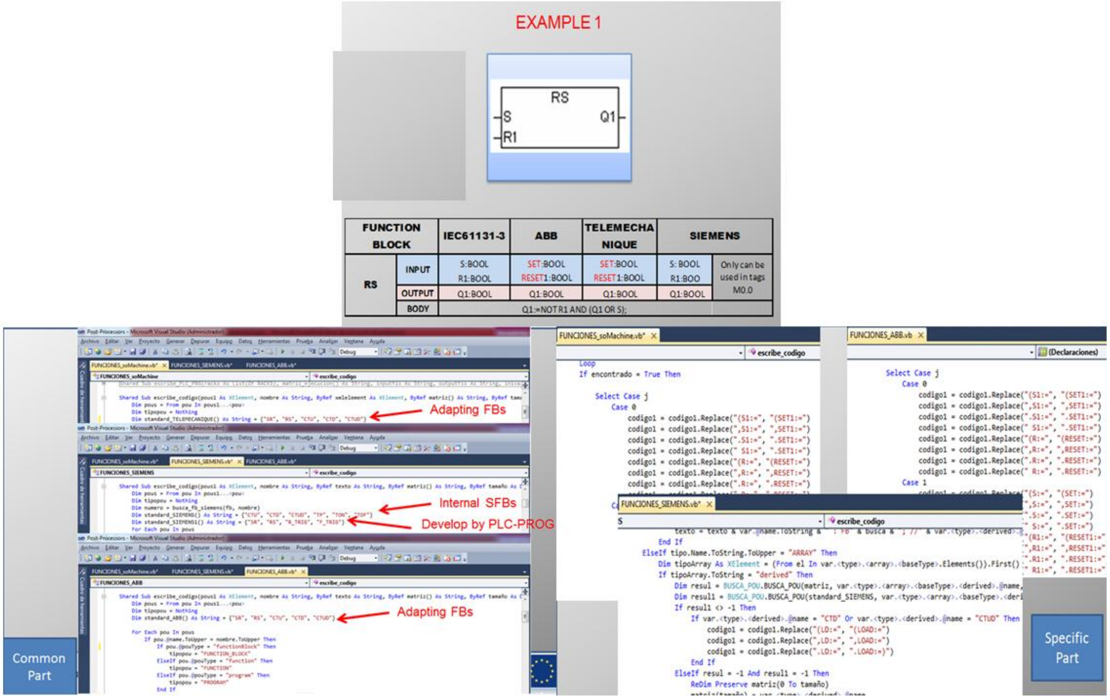


Fig. 33 - Post-Processor Structure

In the following figure, an example is showed by the specifically part of the post-processors and the common part of these.



FUNCTION BLOCK		IEC61131-3	ABB	TELEMECHANIQUE	SIEMENS	
RS	INPUT	S:BOOL R1:BOOL	SET:BOOL RESET1:BOOL	SET:BOOL RESET1:BOOL	S: BOOL R1:BOO	Only can be used in tags M0.0
	OUTPUT	Q1:BOOL	Q1:BOOL	Q1:BOOL	Q1:BOOL	
	BODY	Q1:=NOT R1 AND (Q1 OR S);				

```
FUNCIONES_soMachine.vb*   FUNCIONES_SIEMENS.vb*   FUNCIONES_ABB.vb*
FUNCIONES_soMachine   escribe_codigo
    Shared Sub escribe_codigo(pous1 As XElement, nombre As String, ByRef xmlelement As XElement, ByRef matriz() As String, ByRef tam
        Dim pous = From pou In pous1...<pou>
        Dim tipopou = Nothing
        Dim standard_TELEMECANIQUE() As String = {"SR", "RS", "CTU", "CTD", "CTUD"}
```

```
FUNCIONES_SIEMENS   escribe_codigo
    Shared Sub escribe_codigo(pous1 As XElement, nombre As String, ByRef texto As String, ByRef matriz() As String, ByRef tamaño As I
        Dim pous = From pou In pous1...<pou>
        Dim tipopou = Nothing
        Dim numero = busca_fb_siemens(fb, nombre)
        Dim standard_SIEMENS() As String = {"CTU", "CTD", "CTUD", "TP", "TON", "TOF"}
        Dim standard_SIEMENS1() As String = {"SR", "RS", "R_TRIG", "F_TRIG"}
        For Each pou In pous
```

```
FUNCIONES_ABB   escribe_codigo
    Shared Sub escribe_codigo(pous1 As XElement, nombre As String, ByRef texto As String, ByRef matriz() As String, ByRef tamaño As I
        Dim pous = From pou In pous1...<pou>
        Dim tipopou = Nothing
        Dim standard_ABB() As String = {"SR", "RS", "CTU", "CTD", "CTUD"}

        For Each pou In pous
            If pou.@name.ToUpper = nombre.ToUpper Then
                If pou.@pouType = "functionBlock" Then
                    tipopou = "FUNCTION_BLOCK"
                ElseIf pou.@pouType = "function" Then
                    tipopou = "FUNCTION"
                ElseIf pou.@pouType = "program" Then
                    tipopou = "PROGRAM"
                End If
```

```
FUNCIONES_soMachine.vb*   escribe_codigo
    Loop
    If encontrado = True Then

        Select Case j
            Case 0
                codigo1 = codigo1.Replace("(S1:=", "(SET1:=")
                codigo1 = codigo1.Replace(",S1:=", ",SET1:=")
                codigo1 = codigo1.Replace(".S1:=", ".SET1:=")
                codigo1 = codigo1.Replace(" S1:=", ".SET1:=")
                codigo1 = codigo1.Replace("(R:=", "(RESET:=")
                codigo1 = codigo1.Replace(",R:=", ",RESET:=")
                codigo1 = codigo1.Replace(".R:=", ".RESET:=")
```

```
FUNCIONES_ABB.vb   (Declaraciones)
        Select Case j
            Case 0
                codigo1 = codigo1.Replace("(S1:=", "(SET1:=")
                codigo1 = codigo1.Replace(",S1:=", ",SET1:=")
                codigo1 = codigo1.Replace(".S1:=", ".SET1:=")
                codigo1 = codigo1.Replace(" S1:=", ".SET1:=")
                codigo1 = codigo1.Replace("(R:=", "(RESET:=")
                codigo1 = codigo1.Replace(",R:=", ",RESET:=")
                codigo1 = codigo1.Replace(".R:=", ".RESET:=")
                codigo1 = codigo1.Replace(" R:=", ".RESET:=")
            Case 1
                "(S:=", "(SET:=")
                ",S:=", ",SET:=")
                ".S:=", ".SET:=")
                " S:=", ".SET:=")
                "(R1:=", "(RESET1:="
                ",R1:=", ",RESET1:="
                ".R1:=", ".RESET1:="
                " R1:=", ".RESET1:="
```

```
FUNCIONES_SIEMENS.vb*   escribe_codigo
            End If
        ElseIf tipo.Name.ToString.ToUpper = "ARRAY" Then
            Dim tipoArray As XElement = (From el In var.<type>.<array>.<baseType>.Elements()).First()
            If tipoArray.ToString = "derived" Then
                Dim resul = BUSCA_POU.BUSCA_POU(matriz, var.<type>.<array>.<baseType>.<derived>.@name,
                Dim resul1 = BUSCA_POU.BUSCA_POU(standard_SIEMENS, var.<type>.<array>.<baseType>.<deri
                If resul1 <> -1 Then
                    If var.<type>.<derived>.@name = "CTD" Or var.<type>.<derived>.@name = "CTUD" Then
                        codigo1 = codigo1.Replace("(LD:=", "(LOAD:=")
                        codigo1 = codigo1.Replace(",LD:=", ",LOAD:=")
                        codigo1 = codigo1.Replace(".LD:=", ".LOAD:=)")
                    End If
                ElseIf resul = -1 And resul1 = -1 Then
                    ReDim Preserve matriz(0 To tamaño)
```

Fig. 34 - Example of the post-processors

All of these modules are prepared for different levels, for example the same trigger is ready for a C0, C1 and C2. In fact, the inputs/outputs and the functionality of this module have been ready for post-processing and checking into the corresponding PLCs independent of its brand.

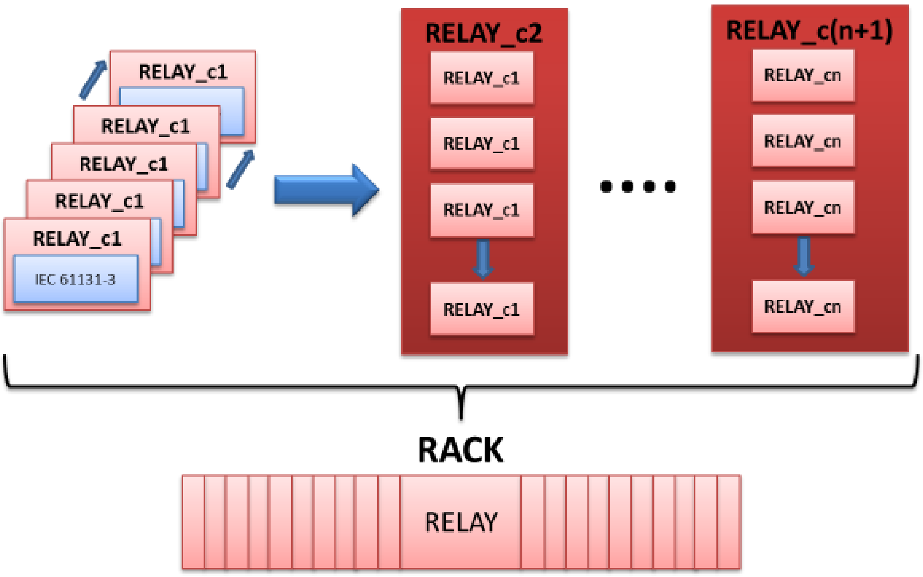


Fig. 35 - PLC-PROG modules

The results are generated for different scales, and ready to use racks have been produced, which can be integrated in any PLC-PROG user library. The code is presented both in ST language and in XML format.

In particular, we have shown how to apply the post-processors developed in the first year of PLC-PROG to the basic functions defined in the IEC standard. These functions can be summarised in set of functions, listed below:

- *Bi-stable function blocks*
- *Trigger function blocks*
- *Counter function blocks*
- *Timer function blocks*

In term of main benefits, it is possible to mention the following:

- **Reduced Memory Program:** the post-processors manage the organization and the execution order of the racks. During the development of the control program the logical organization is like in the figure 34, but the structure inside of the PLC is linear, as in Fig. 36.

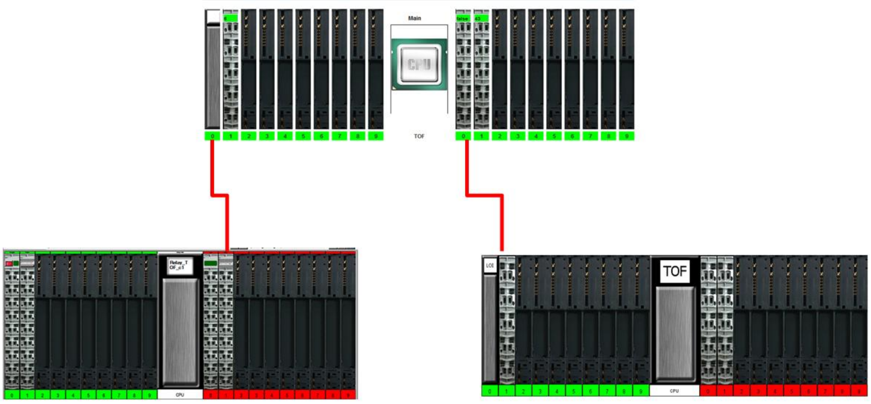


Fig. 36 - Logical Control Program

During this process the post-processors organize and group the signals, optimize their memory footprint, achieving a great reduction in the size of the program.

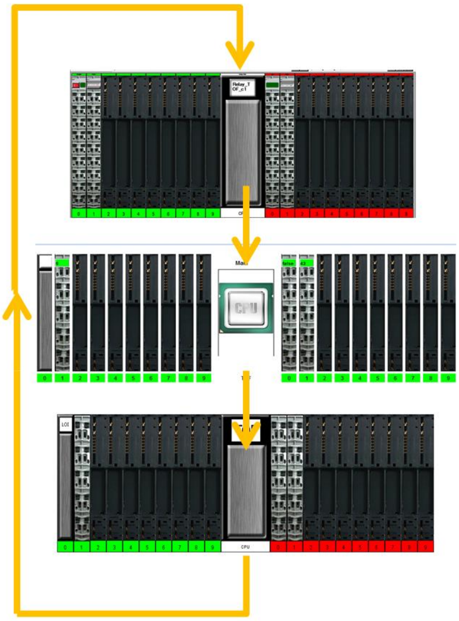
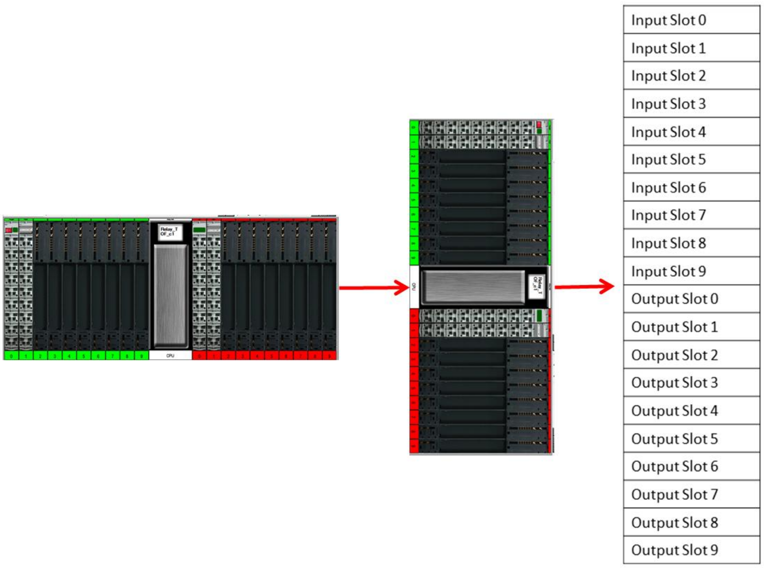


Fig. 37 - Control Program Execution inside the PLC

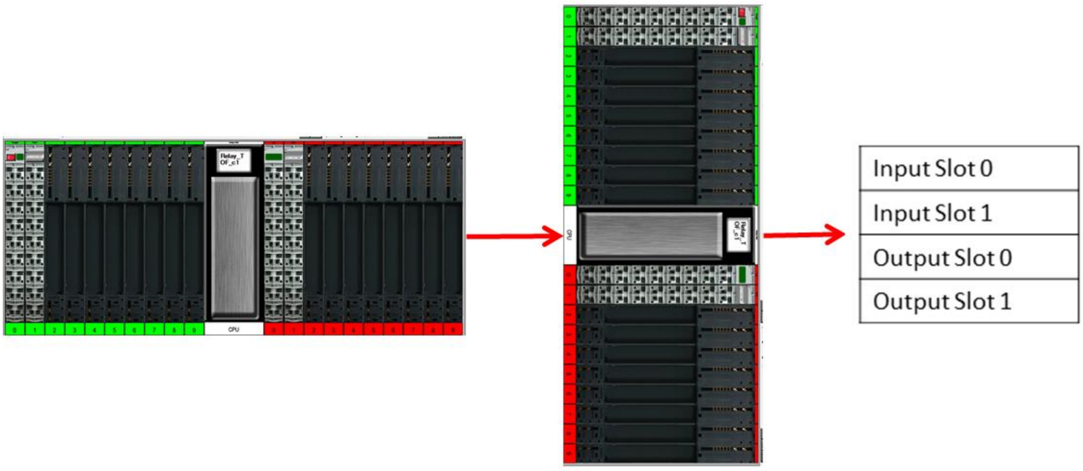
- **Data Memory Assignment INDEPENDENT of the programmer:** the data memory layout is done by the post-processors, independently of the programmer. So it is not necessary to share a table of variables (memory map) between the SCADA System and Control PLC.
- **Reduced Memory Variables:** the post-processors manage the organization of data in the PLCmemory. PLC-PROG assigns automatically the memory to each rack, as in Fig. 38.



Input Slot 0
Input Slot 1
Input Slot 2
Input Slot 3
Input Slot 4
Input Slot 5
Input Slot 6
Input Slot 7
Input Slot 8
Input Slot 9
Output Slot 0
Output Slot 1
Output Slot 2
Output Slot 3
Output Slot 4
Output Slot 5
Output Slot 6
Output Slot 7
Output Slot 8
Output Slot 9

Fig. 38 - Memory Variables Organization

After arranging the data memory, the free space is remove and the memory is compacted.

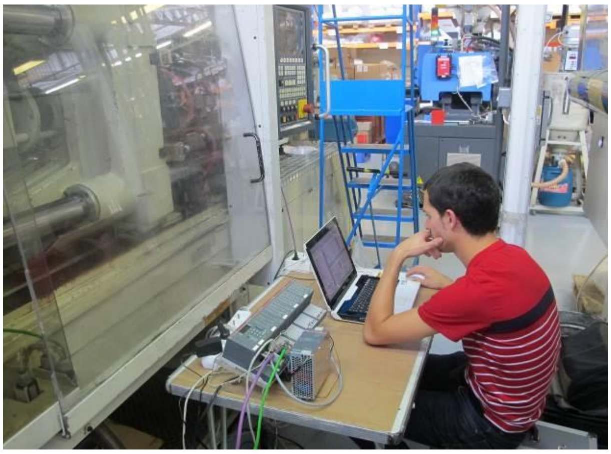


Input Slot 0
Input Slot 1
Output Slot 0
Output Slot 1

Fig. 39 - Reduced Memory Variables Organization

3.7. Application and validation of the PLC-PROG system in pilot plants

The PLC-PROG system has been tested with industrial processes (two water plants in Romania and one injection moulding machine in UK), in order to assess both the viability and the performance of the PLC-PROG programming tool. Indeed, in this specific framework of activities, different plc brands have been used in each selected installation, but all the programs have been generated within the same PLC-PROG system, and the finally, the brand-dependant code has been automatically generated by using the post-processor tool, integrated in the PLC-PROG environment. The IEC-61131-3 functions and most of the technological functions developed within the WP3 have been successfully reused in the three installations in order to assess the openness of PLC-PROG.



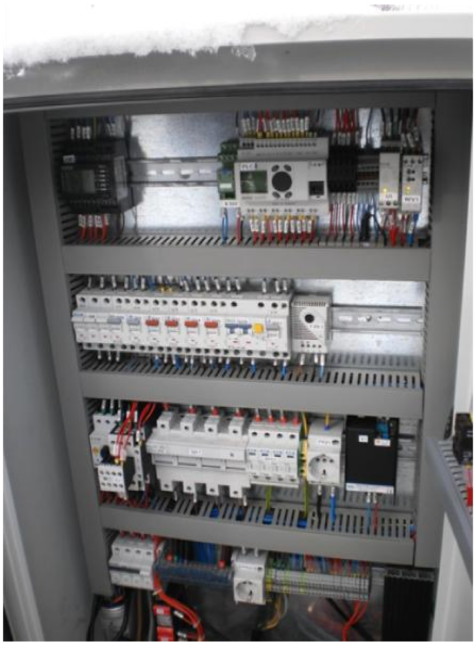


Fig. 40 - Pilot installations in UK (left) and Romania (right)

1. 3-axis industrial machine

The 3 axis-machine was made available by UPV partner and it integrates 3 individually controlled position axis, one with encoder feedback and the other two with analog position sensor feedback.

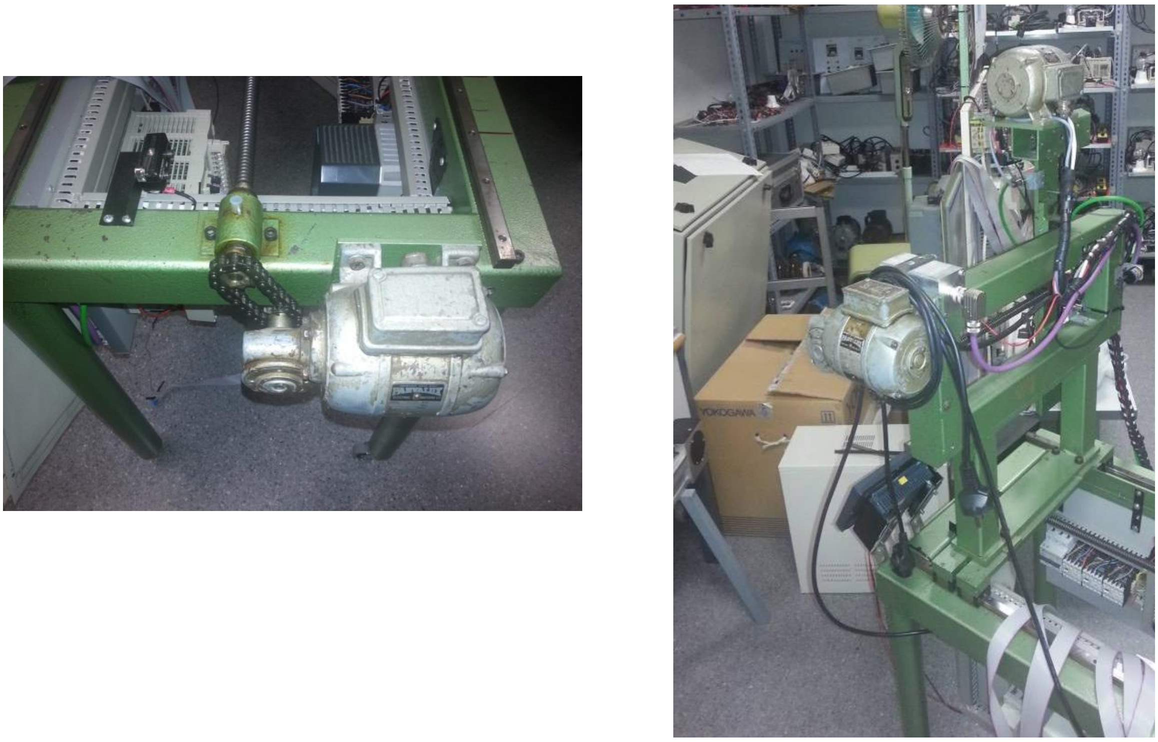


Fig. 41 - 3-axis demonstrator machine

The three motors are fed by a single variable speed drive, which is switched by three contactors in function of the axe that it is in movement.

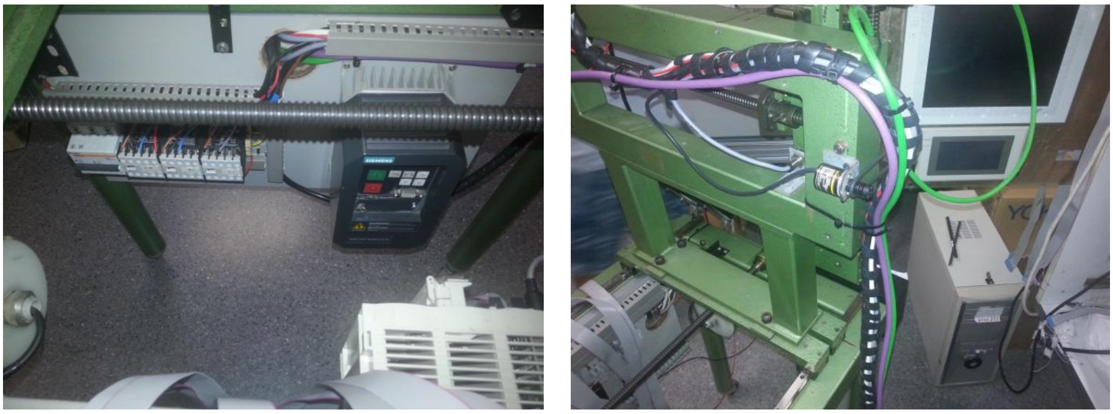


Fig. 42 - 3-axis demonstrator machine

Each axe has a different position sensors such as an encoder for measuring the speed or an analog potentiometer. Edge and zero-position digital sensors are mounted on each axe.

The machine has also a panel button and a touch panel to operate it locally in manual or automatic mode; both elements work simultaneously.

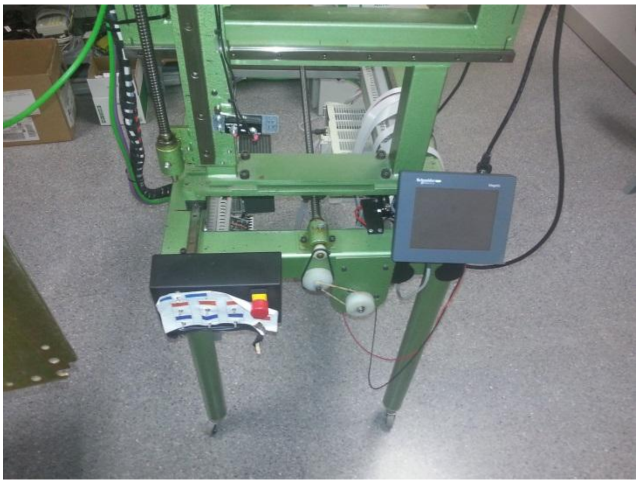


Fig. 43 - 3-axis demonstrator machine

The machine is provided with a local PLC, not compliant with the IEC-61131-3 standard, that controls the machine, and has been programmed with traditional tools.

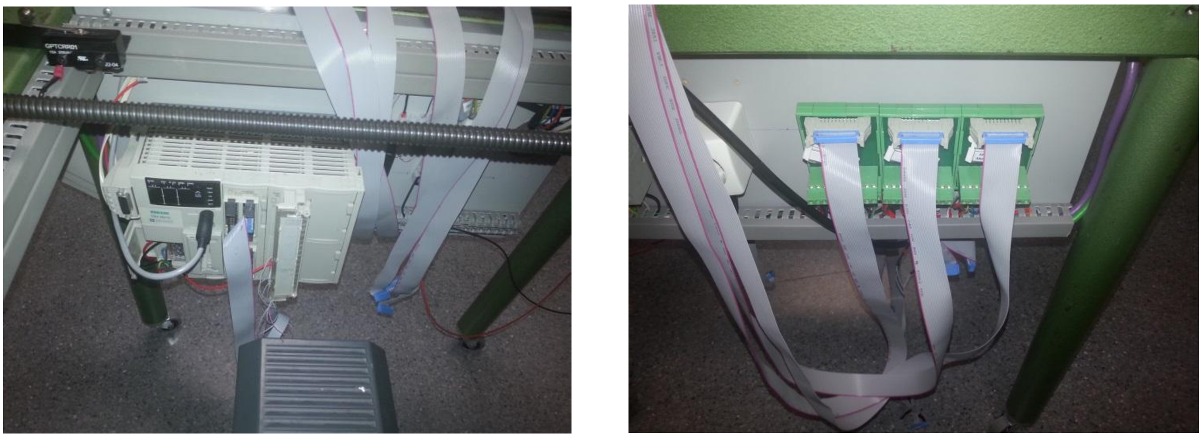


Fig. 44 - 3-axis demonstrator machine

To make the PLC-PROG tests using three different PLC brands, the machine has been equipped with three fast connectors for easy exchange between the selected PLC brand (local control in traditional way, ABB, Siemens or Telemecanique, the last three programmed using PLC-PROG tool).

PLCs: we have the three brands of PLCs that are supported by PLC-PROG: Siemens, Telemecanique, and ABB.

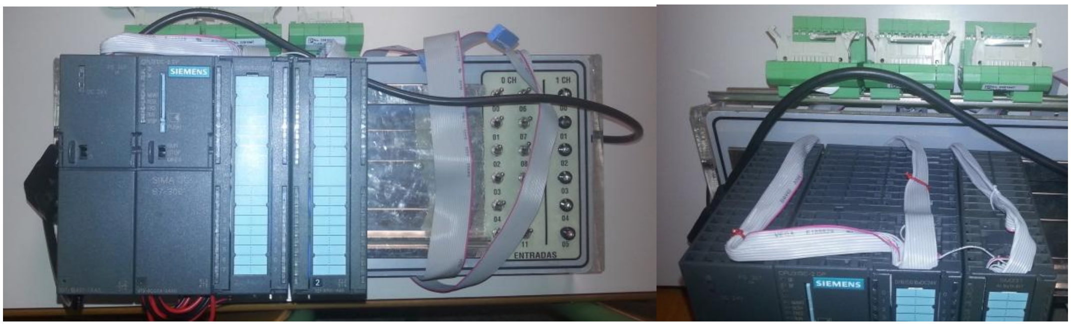


Fig. 45- Siemens 313-2DP

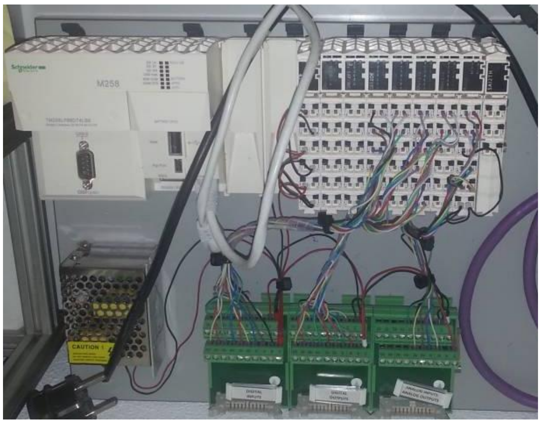


Fig. 46 - Telemecanique M258

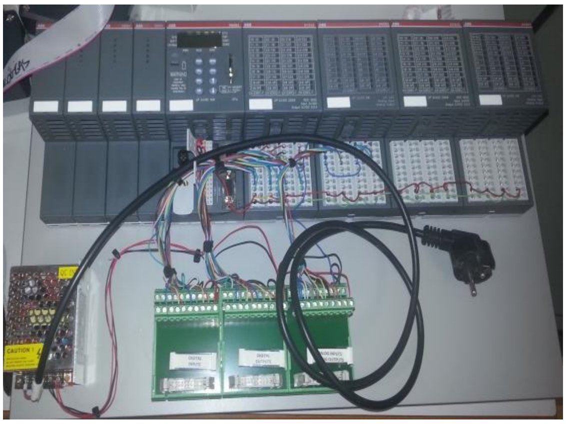


Fig. 47 - ABB PM583

The same PLC-PROG control project of the three axis machine can be downloaded to any of the 3 PLCs installed in the machine, and their performance compared with the old program. Any changes to the PLC-PROG program can be so inmediately downloaded to any of the desired PLC.

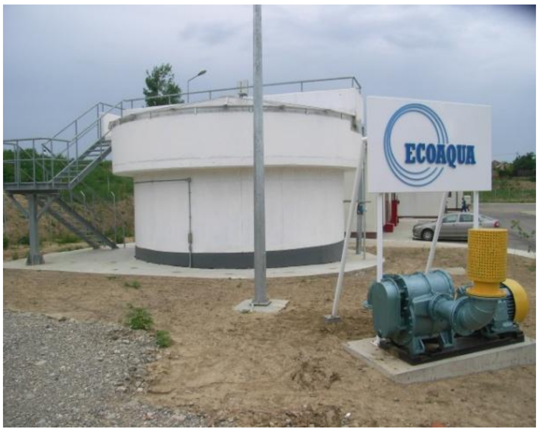
2. Water extraction plant or waste water treatment plant

The second demo was provided through one of the plants made available by one the participants (Mikon) in Romania. The application was controlled by using a PLC of the brand ABB: the model PM573 with a module of 8 digital inputs/8 digital outputs and a modules of 4 analog inputs/4 analog output. It can be configured and the i/o can be set as digital or analog.

The application was used for:

- ✓ showing the electrical cabinet where the ABB PLC has been installed for demo purposes
- ✓ showing the program created in PLC-PROG to control the application
- ✓ showing the post-processed program in ABB software
- ✓ showing the SCADA in which the operation can run the application
- ✓ running the application and explain the data visualized

There is a central building where the water filters are installed as well as the Control room where the operators will control the whole installation. The pumping station is situated about two meters under and accessible via a ladder through the blue trapdoor.



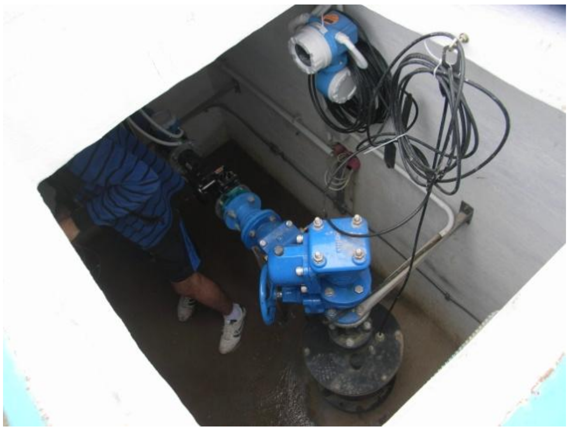


Fig. 48- MIKON's plant: water pump installation

Inside the trapdoor there is a pipe installation with a water pump, a flow meter, a level meter and the electrical cabinet where the PLC is mounted. This pump (among others) is used to bring the water inside the building to be treated. The PLC is used to control the pump automatically regarding some conditions such as level or remote starting command.

Summarising, the performed validation has showed that the project technical objectives have been fully achieved, and in particular that:

- It is possible to create the code for an industrial application using PLC-PROG, with a **programming time** which, for an user already experienced in PLC-PROG use, can be estimated at least equal and in some cases **lower** than with a conventional language.
- It is possible **to reuse** the same program code with 2 or more different PLC brands with minimal effort, leading to **an average reduction of the implementation time of 85%**, which is an impressive result.
- It is possible **to reuse** existing code from old application and encapsulate it in PLC-PROG functions for further use.
- It is possible **to create SCADA visualization** of the PLC variables' values with minimal configuration to embed in a SCADA software or Web Browser.

The combination of the features have been positively evaluated by the users from both an application and market point of view, and allow to state that:

- **PLC-PROG saves program development time** when using different PLC and SCADA brands because the same program can be easily adapted to supported brands.
- **PLC-PROG saves program maintenance time** when making modifications in the functions because the versioning system helps to restore/update functions easily.
- **The learning curve is fairly short** as the software is very intuitive. During the training sessions the attendants could create simple programs.
- **PLC-PROG permits non expert users to create or maintain PLC programs**.

4. Potential impact, dissemination activities and the exploitation of results

4.1. Socio economic impact

According to the main market analyst organizations at worldwide level, such as: IMS Research, ARC Advisory Group and TechNavio; last years have been extremely difficult for most PLC suppliers.

As main consequence of the world economic crisis started in 2008, it was forecast that the market will not recover until the second half of 2012. Moreover, it was underlined as no region was insulated from the effects of the economic downturn: Europe, the Middle East and Africa (EMEA), the largest market for PLCs, were the most regions badly affected. The main factor behind that was that Germany, the most important market for PLCs in EMEA, suffered the largest contraction during the global recession. In fact, after a sharp fall in the German export business, improvement in activity was forecast to be slower than elsewhere, especially until the middle of 2010. Conversely, the 2009 market contraction was least marked in Asia; this market was forecasted to grow the fastest from 2008 to 2013, with a revenue CAGR of 6%. That was due to improved prospects in China and India, in part reflecting substantial macroeconomic stimulus and a faster-than-expected turnaround in capital flows.

However, **last trends estimated for the next future**: *"Western Europe is the most important market for EMEA, as Germany led the way and was the first country coming out of the recession. Germany will continue to be the market leader in the PLC market and be the main driving force for future demand. The emerging markets in Eastern Europe, where automation level is relatively low but growth is rapid, will make up for some of the negative growth regions. As the EMEA market is mature, the PLC growth in EMEA will be relatively slow at 10.4% growth in 2011. This region is expected to retain a single digit growth rate over the next few years. China and India will lead growth next years with an future annual growth of around 20%. In Americas the overall market is projected to enjoy a growth rate of more than 11% the following years".*

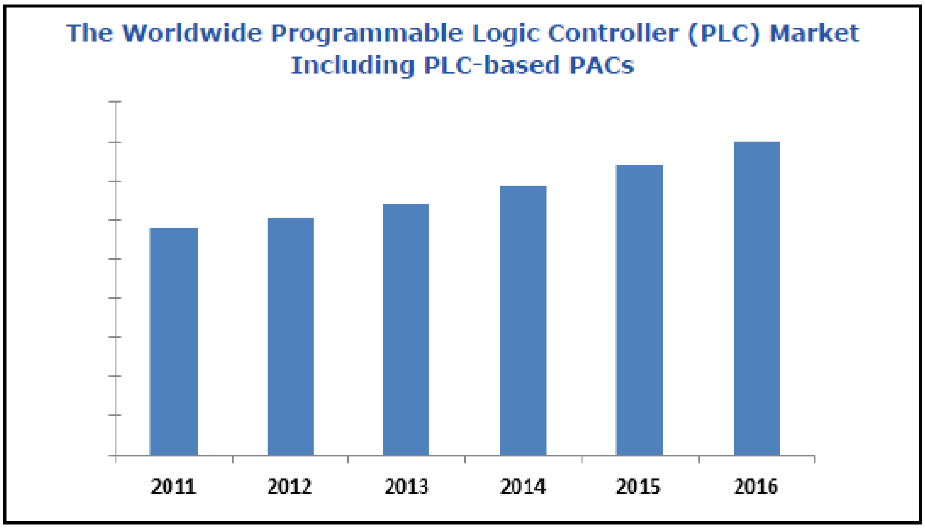


Fig. 49 - PLC's market trend (source ARC Advisory Group 2012)

As showed in the graphic above, the same market trend' forecast is also confirmed in the report provided by another of the main market analyst organization (ARC Advisory Group), which has stated in one of the most recent publication the following[1]: "***the economic growth that had started in 2010 continued in 2011****. While demand for automation remained strong for PLCs and PLC-based PACs during the first half of 2011, the pace of growth slowed as the year progressed, caused by the escalating sovereign debt crisis in a number of industrialized countries as well as specific concerns about some of the economies in southern Europe. Despite the looming financial crisis in Europe and instability in the Middle East,* ***there are crucial factors that will continue to drive the use of automation, fueling PLC and PLC-based PAC market growth****"*. Moreover, it continues stating that: *"the industrial objectives, such as: improving plant or machinery utilization, yield, product quality, availability, safety, and delivery performance, strongly influence capital investments in automation.* ***Automation suppliers, especially PLC*** *and PLC-based PAC suppliers,* ***are well positioned in this environment as this equipment is widely employed across all industrial segments*** *as companies face challenges to raise productivity, lower product costs, reduce plant operating expenses, and increase return on investment".*

In this context, and in order to validate this positive trend, it is also worth to mention briefly the market forecast indicated by the TechNavio's analysts[2]: "***the Global Programmable Logic Controllers market to grow at a CAGR of 9.1 percent over the period 2011-2015****. One of the key factors*

[1] according to 2012 ARC Advisory Group's report: "PLC and PLC-based PACs Worldwide Outlook"

contributing to this market growth is governmental regulations and policies. The Global Programmable Logic Controllers market has also been witnessing the increasing demand for integrated programmable logic controllers. However, the high cost of customer switching could pose a challenge to the growth of this market".

The presented data allow us to agree on the final conclusion stated by these main market analysts claiming that: *"althought the PLC suppliers and their industrial customers are currently very uncertain whether growth will continue, considering the risks to the world economy,* ***the PLC market will still grow in the next three-year period, mainly because many large and important PLC markets,*** *such as Germany, France, China and the US, are still performing well at the turn of the year. In addition,* ***emerging markets,*** *such as Brazil, and India, which already account for half the entire PLC market,* ***will be the main driving force for future growth"****.*

The programmable logic controller (PLC) was originally designed to replace hardwired systems composed of hundreds or thousands of relays, cam timers, drum sequencers, and dedicated closed-loop controllers. Updating these relay-based systems to accommodate changes and additions in the controlled machines and processes was a lengthy and expensive process, requiring electricians to add components and perform extensive rewiring. Troubleshooting was a nightmare, as problems could occur among thousands of connections, switch contacts, and hundreds or thousands of components.

New architectures are expanding capabilities, blurring the line between programmable automation controllers and PLCs. And at this scope, all the selected article have provided clear explanation how advances in processing power have enabled PLCs to execute more functions in less time at lower cost, such that PLC functionality now often approaches programmable automation controller (PAC) power.

Leading advances of modern PLCs
1. Additional features incorporated in PLC CPU, eliminating external components
2. High-speed serial or Ethernet communications to remote wired or wireless I/O

[2] according to the TechNavio's report "the Global Programmable Logic Controllers Market 2011-2015".

3. Wide choice of I/O form factors
4. Custom functionality and faster execution, allowing PLC to take on more tasks
5. Advanced instruction reduce PLC programming time
6. Superior memory mapping improves data handling
7. Data structures simplify programming and maintenance

However in this context, it's worth to note that, although OOP has demonstrated its capability for handling complex software development problems in an elegant way and for producing flexible, reusable software components, no single PLC incorporates all of the features listed. Rather, different products from various suppliers have various combinations of strengths and weaknesses, and no one product is the best fit for all applications.

Some practical examples have been developed through the years for the application of an object-oriented methodology to PLC programs. An example is reported in a publication[3] showing how the standard IEC 61131 can be applied to the development of the control software of a medium complexity manufacturing machinery.

At this scope, it is worth to report what is also stated by Codesys[4] in relation to this last technological aspect; specifically that the **industrial controllers (PLC) are mostly programmed in the languages of the IEC 61131-3 standard**. While some developers cannot wait to use OOP for PLC-programming, others may be skeptical about the adequacy of OOP for their PLC-projects.

In order to address both parties, an object-oriented programming tool like the PLC-PROG should satisfy the following requirements:

[3] *PLC Object-oriented programming using IEC 61131 norm languages: an application to manufacture machinery* - Marcello Bonfe, Cesare Fantuzzi and Luca Poretti – University of Ferrara (2001).

[4] Source : CodeSys website http://www.3s-software.com/

- ✓ **Integration in a PLC development environment**: Integrated with the object-oriented programming itself, one should have integrated configuration of I/Os, direct access to I/O signals, and online debugging functionality with variable forcing and online change.

Advantage: PLC-application development has many specifics, which exiting C++ development tools available for many target-CPUs do not address.

- ✓ **Multi-paradigm programming:** Object-orientation should be optional. Code programmed in the classical, procedural way should be mixable with object-oriented code.

Advantage: this offers the sceptics a stepwise and reversible transition.

- ✓ **OOP by extension of the IEC:** Object-oriented PLC-programming should be supported by extending IEC programming with a small set of standard object-oriented features.

Advantage: not having to learn a completely new language avoids a steep learning curve for PLC-programmers.

- ✓ **Multi-lingual:** Object-oriented programming should be supported in all languages of the IEC 61131-3, not just in the textual IEC-language “ST” most similar to C++ and other known object-oriented languages.

Advantage: textual languages cannot represent clearly certain important aspects of PLC applications, like state machines and complex Boolean connection networks.

4.2. Dissemination of results

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES								
No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
1	INTERVIEW	GAIA	Electronic mailing to companies	M12 - M36 August 2011 - August 2013	Spain, Greece, UK	SME	436 companies	Spain, Greece, UK
2	CONFERENCE	BPF	PLAST 2012 Conference	from 8th- 12th May 2012	Milan, Italy	PLASTICS SECTOR	-	Europe
3	EXHIBITION	BPF	Plastics Design and Moulding Exhibition	29th- 30th May 2012	Telford (UK)	PLASTICS SECTOR	-	UK
4	EXHIBITION	BPF	PlastEurasia 2012	27th November- 2nd December 2012	Istanbul, Turkey	PLASTICS SECTOR	1085 companies from 43 countries	Europe

[5] A drop down list allows choosing the dissemination activity: publications, conferences, workshops, web, press releases, flyers, articles published in the popular press, videos, media briefings, presentations, exhibitions, thesis, interviews, films, TV clips, posters, Other.

[6] A drop down list allows choosing the type of public: Scientific Community (higher education, Research), Industry, Civil Society, Policy makers, Medias, Other ('multiple choices' is possible).

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES								
No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
5	EXHIBITION	BPF	InterPlastica 2013	January – 1st February 2013.	Moscow, Russia	PLASTICS SECTOR	**712 exhibitors** from 27 countries	Russia, Europe
6	EXHIBITION	BPF	Total Processing and Packaging Exhibition	-	Birmingham, UK	PLASTICS SECTOR	-	30 different countries
7	CONFERENCE	GAIA	Electronics Goes Green 2012	9th to 12th September 2012	Berlin, Germany	IT companies	450 conference delegates	Europe, EEUU, Taiwan, etc
8	CONFERENCE	GAIA	TCI Annual Global Conference 2012	16th to 19th October 2012	Basque Country, Spain	IT companies, universities, clusters, etc	-	Europe
9	CONFERENCE	GAIA	Artemis Summer Event	13th and 14th March 2013.	Brussels, Belgium	SMEs, Universities, Associations	200 members of the	Europe

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
							organization	
11	EXHIBITION	SEPE	CEBIT 2013	5th- 9th March	Hannover, Germany	IT companies, Universities, associations, etc	-	Europe, Japan, China, EEUU, etc
12	EXHIBITION	SEPE	E-volution Awards Greece 2013	19th December 2012	Athens, Greece	SME and Universities	-	Greece
13	CONFERENCE	SEPE	6th Pan-Hellenic Electrical and Computer Engineering Students Conference	17th March 2013	Athens, Greece	Electrical and Computer Engineering student	-	Greece
14	CONFERENCE	SEPE	Job Fair Athens 2013	3rd- 4th April 2013	Athens, Greece	Students, Universities, industry	-	Greece
15	CONFERENCE	MIKON	Matching 2012	26th and 28th Nov. 2012	Milan, Italy	Automation sector	-	Europe

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
16	CONFERENCE	MIKON	AAIR 2013 (Romanian Automation Companies Association)	10th to 12th May 2013	Bucharest, Romania	Automation sector	-	Romania
17	EXHIBITION	DOTSOFT	Thessaloniki International Fair	7th to 15th September 2013	Thessaloniki, Greece	ICT	-	Greece
18	EXHIBITION	DOTSOFT	TIF Helexpo Trade show,	08/10/2013	Thessaloniki, Greece	Automation sector		Greece
19	CONFERENCE	LOGITEK	Wonderworld 2013	9th - 10th May 2013	Toledo, Spain	Large industries	175 attendees	Europe
20	CONFERENCE	LOGITEK	PROFIBUS Day, 1st Day Official Technical PROFIBUS /PROFINET	June 27th, 2013	Madrid, Spain	Automation sector	-	Europe

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
21	CONFERENCE	LOGITEK	Congress of Industrial Cybersecurity	October 10th, 2012	Madrid, Spain	cybersecurity industry	123 attendees	Europe
22	CONFERENCE	LOGITEK	Technology Seminar: Profibus & Profinet	27 of February 2013	Barcelona, Spain	Automation industry	25 attendees	Spain
23	WORKSHOP	GAIA	Business meeting	M24 - M36 1 August 2012 – 1 August 2013	San Sebastian, Madrid (Spain)	ICT Companies	30 companies	Spain
24	WORKSHOP	GAIA	Territorial meeting	1 April 2013	San Sebastian, Bilbao, Vitoria (Spain)	ICt Companies	60 companies	Spain
25	WORKSHOP	GAIA	General Assembly	1 June 2013	San Sebastian (Spain)	ICT Companies	200 companies	Spain

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
26	WORKSHOP	SEPE	General Assembly	3 June 2013	Athens, Greece	ICT Companies	100 companies	Greece
27	WORKSHOP	BPF	General Assembly	4 June 2013	London, UK	ICT Companies	300 companies	UK
28	CONFERENCE	BPF	1st training session of PLC prog	10 April 2013	London, UK	Consortium companies	5 companies	Consortium companies
29	CONFERENCE	GAIA	2nd training session of PLC prog	24th May 2013	San Sebastian (Spain)	ICT companies	12 companies	Spain
30	CONFERENCE	SEPE	3th training session of PLC prog	22nd July 2013	Athens, Greece	ICT Companies	15 companies	Spain
31	PRESS RELEASE	GAIA	Publication of an article about PLC PROG in electronic magazines	1 October 2012	Spain	All	22 publications	Spain

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES								
No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
32	ARTICLES PUBLISHED IN THE POPULAR PRESS	GAIA	Publication of an article about the training of PLC PROG in electronic magazines	2 May 2013	Spain	All	11 publication	Spain
33	ARTICLES PUBLISHED IN THE POPULAR PRESS	GAIA	Publication of an article about PLC PROG in GAIA SAREAN	5 November 2012	Spain	ICT companies	260 stakeholders	Spain
34	ARTICLES PUBLISHED IN THE POPULAR PRESS	GAIA	Publication of an article about the training of PLC PROG in GAIA	10 May 2013	Spain	ICT Companies	260 stakeholders	Spain

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
			SAREAN					
35	ARTICLES PUBLISHED IN THE POPULAR PRESS	ITT	Publication in specialized media	20 June 2013	Blog blog of Dr. Geertjan Wielenga, a product manager in the Oracle Developer Tools group in Amsterdam.	Automation sector	-	All Europe
36	WORKSHOP	ITT	Presentation of PLC PROG to OPC Foundation	10th October 2012.	Paris, France	UPC Foundation	100 attendees	All Europe
37	WORKSHOP	GAIA	Article sent to interested companies	M24 - M32 1 August 2012 - August 2013	Mailing	ICT and Auromation sector	200 stakeholders	Europe
38	WORKSHOP	UPV	Dissemination	M12 - M36	During the	ICT, automation	-	Europe

TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
			of PLC PROG through a presentation	(August 2011 – 10 August 2013)	international and national events	and plastics sectors		
39	VIDEO	ITT	Project video available	M12 - M36 (1 August 2011 - August 2013)	Websites	ICT, automation and plastics sectors	-	Europe
40	BROCHURES	LABOR	Dissemination of brochures in English, Romanian, Greek, Spanish, Italian , German, Russsian	M12 - M36 (1 August 2011 - August 2013)	During the international and national events	ICT, automation and plastics sectors	1000 stakeholders	Europe

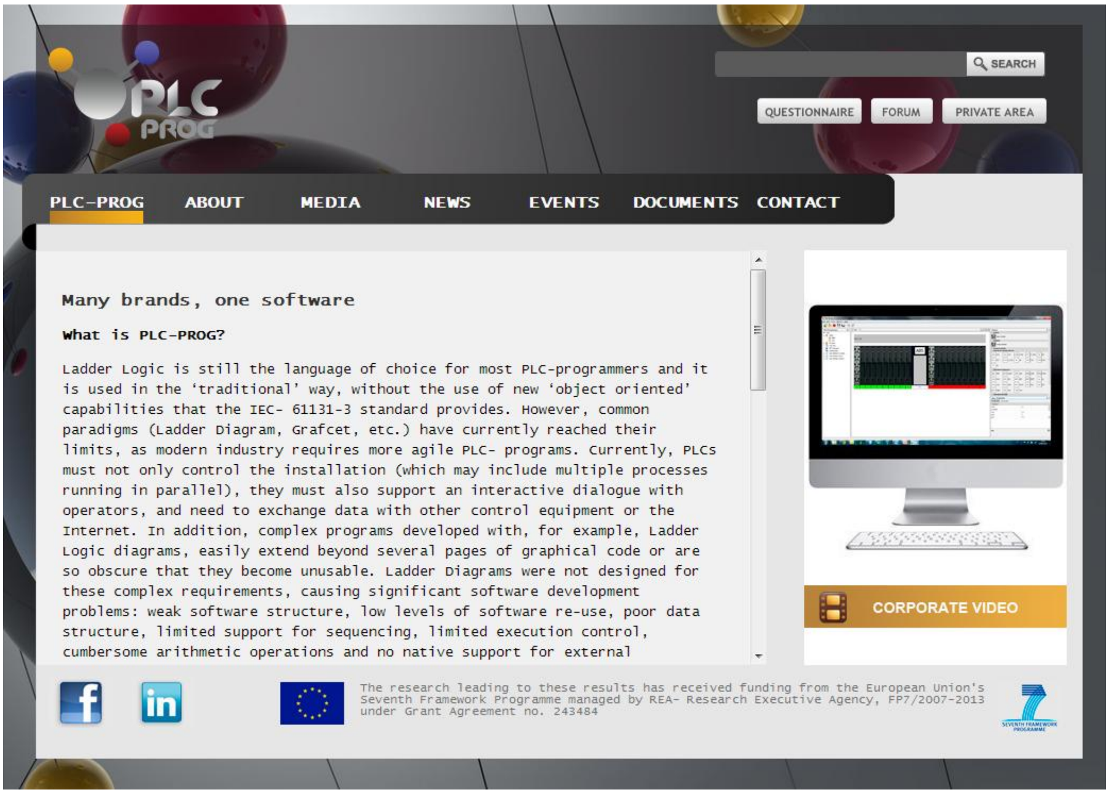
TEMPLATE A2: LIST OF DISSEMINATION ACTIVITIES

No.	Type of activities[5]	Main leader	Title	Date/Period	Place	Type of audience[6]	Size of audience	Countries addressed
41	PROJECT INFORMATION IN WEBSITES	SME AG AND SME (GAIA)	Project information published in partners websites	M12 - M36 (1 August 2011 - August 2013)	Websites	-	-	Europe
42	ARTICLES PUBLISHED IN THE POPULAR PRESS	LABOR	PLC PROG: Anew graphical, object oriented and brand independent programming framework for PLCs	2012-09-11	Brussels			Europe

4.2.1. Web site

A project website (http://www.plc-prog.eu/) representing the first communication tool and main channel towards the target industrial audience of PLC-PROG technology has been designed and developed according specifications and further improvement suggestions.

It contains different sections to explain the project and the product, the concept and objectives, images, news, events, public documentation and access to private area, wiki and social networks.



Many brands, one software

What is PLC-PROG?

Ladder Logic is still the language of choice for most PLC-programmers and it is used in the 'traditional' way, without the use of new 'object oriented' capabilities that the IEC- 61131-3 standard provides. However, common paradigms (Ladder Diagram, Grafcet, etc.) have currently reached their limits, as modern industry requires more agile PLC- programs. Currently, PLCs must not only control the installation (which may include multiple processes running in parallel), they must also support an interactive dialogue with operators, and need to exchange data with other control equipment or the Internet. In addition, complex programs developed with, for example, Ladder Logic diagrams, easily extend beyond several pages of graphical code or are so obscure that they become unusable. Ladder Diagrams were not designed for these complex requirements, causing significant software development problems: weak software structure, low levels of software re-use, poor data structure, limited support for sequencing, limited execution control, cumbersome arithmetic operations and no native support for external

The research leading to these results has received funding from the European Union's Seventh Framework Programme managed by REA- Research Executive Agency, FP7/2007-2013 under Grant Agreement no. 243484

Fig. 50 - Website homepage

On the web site it is possible to download public documents and deliverables.

4.2.2. Corporate Video, Brochure and poster

A Corporate Video is available on

- project's website http://www.plc-prog.eu/video
- Youtube channel http://www.youtube.com/watch?v=MlLqD5RExMQ&feature=youtu.be.

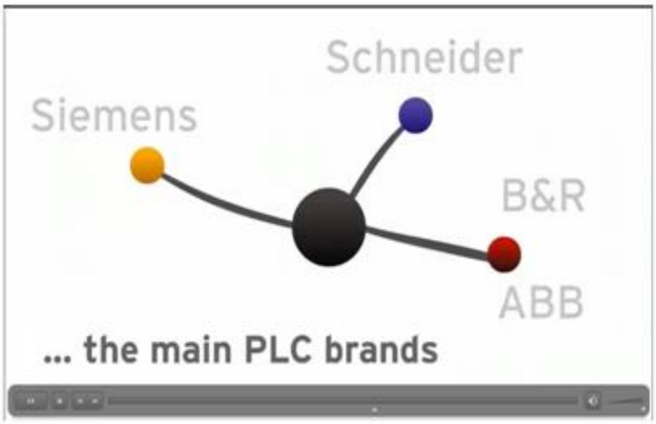
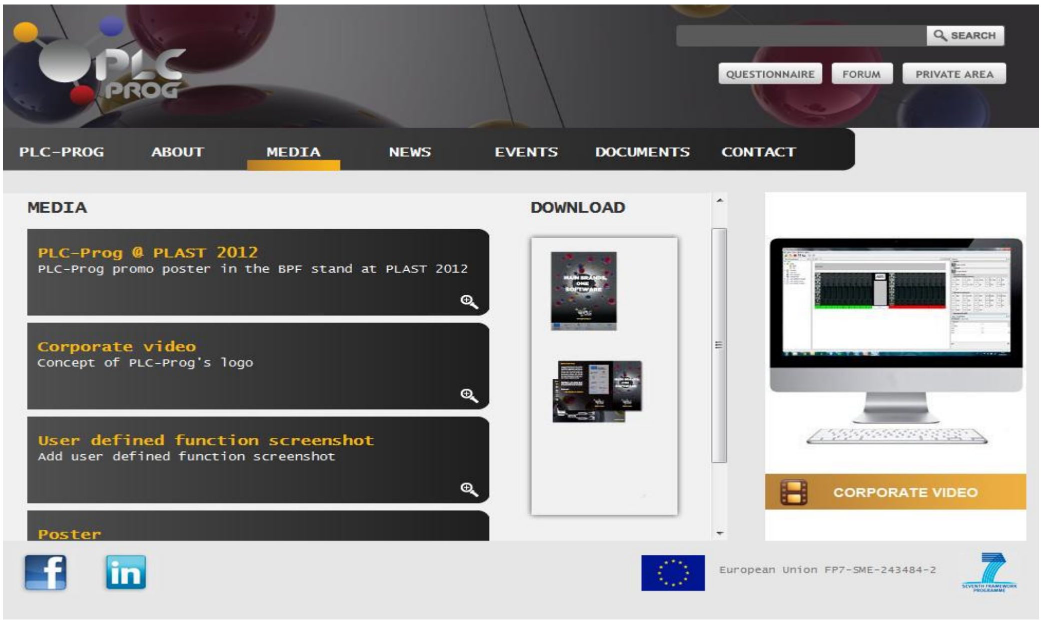


Fig. 51 - The concept of "Many Brands, One Software"

Projects' brochure and poster can be downloaded from the website at the following links:

- ✓ http://www.plc-prog.eu/files_general/1329385979_triptic_web.pdf
- ✓ http://www.plc-prog.eu/files_general/1329385979_poster_web.pdf



MEDIA

PLC-Prog @ PLAST 2012
PLC-Prog promo poster in the BPF stand at PLAST 2012

Corporate video
Concept of PLC-Prog's logo

User defined function screenshot
Add user defined function screenshot

Poster

European Union FP7-SME-243484-2

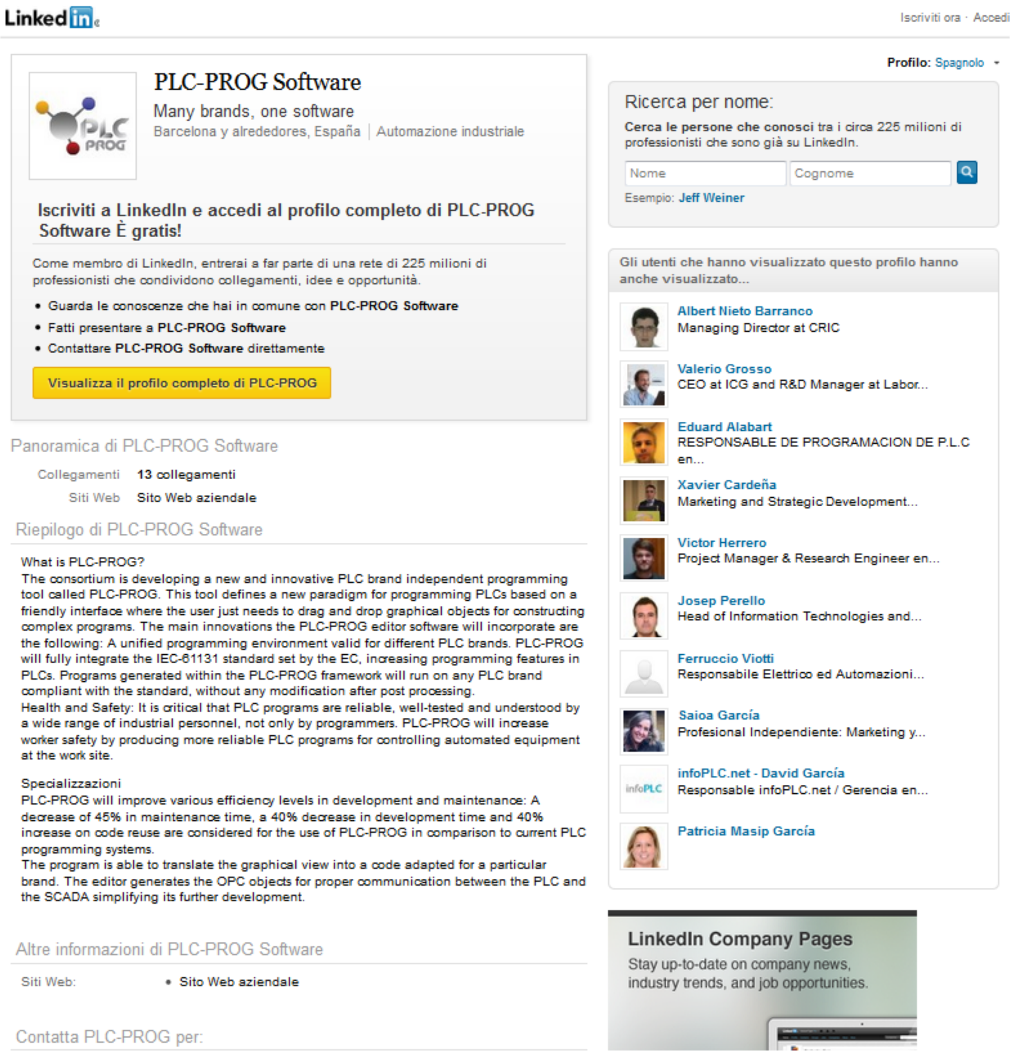
Fig. 52 - Media webpage for the download of the project brochure and poster

4.2.3. Pages on social networks for PLC-PROG project

Specific PCL PROG pages were created on the 2 most important existing social and professional networks *Facebook* and *LinkedIn*.

From the project's website, it is possible to access the pages dedicated to PLC-PROG at the following links:

- https://www.facebook.com/pages/PLC-Prog/143398389113402
- http://es.linkedin.com/pub/plc-prog-software/47/462/35



Linked in

Iscriviti ora · Accedi

PLC-PROG Software

Many brands, one software

Barcelona y alrededores, España | Automazione industriale

Iscriviti a LinkedIn e accedi al profilo completo di PLC-PROG Software È gratis!

Come membro di LinkedIn, entrerai a far parte di una rete di 225 milioni di professionisti che condividono collegamenti, idee e opportunità.

- Guarda le conoscenze che hai in comune con **PLC-PROG Software**
- Fatti presentare a **PLC-PROG Software**
- Contattare **PLC-PROG Software** direttamente

Visualizza il profilo completo di PLC-PROG

Panoramica di PLC-PROG Software

Collegamenti	13 collegamenti
Siti Web	Sito Web aziendale

Riepilogo di PLC-PROG Software

What is PLC-PROG?

The consortium is developing a new and innovative PLC brand independent programming tool called PLC-PROG. This tool defines a new paradigm for programming PLCs based on a friendly interface where the user just needs to drag and drop graphical objects for constructing complex programs. The main innovations the PLC-PROG editor software will incorporate are the following: A unified programming environment valid for different PLC brands. PLC-PROG will fully integrate the IEC-61131 standard set by the EC, increasing programming features in PLCs. Programs generated within the PLC-PROG framework will run on any PLC brand compliant with the standard, without any modification after post processing.

Health and Safety: It is critical that PLC programs are reliable, well-tested and understood by a wide range of industrial personnel, not only by programmers. PLC-PROG will increase worker safety by producing more reliable PLC programs for controlling automated equipment at the work site.

Specializzazioni

PLC-PROG will improve various efficiency levels in development and maintenance: A decrease of 45% in maintenance time, a 40% decrease in development time and 40% increase on code reuse are considered for the use of PLC-PROG in comparison to current PLC programming systems.

The program is able to translate the graphical view into a code adapted for a particular brand. The editor generates the OPC objects for proper communication between the PLC and the SCADA simplifying its further development.

Altre informazioni di PLC-PROG Software

Siti Web:
- Sito Web aziendale

Contatta PLC-PROG per:

Profilo: Spagnolo

Ricerca per nome:

Cerca le persone che conosci tra i circa 225 milioni di professionisti che sono già su LinkedIn.

Nome | Cognome

Esempio: **Jeff Weiner**

Gli utenti che hanno visualizzato questo profilo hanno anche visualizzato...

Albert Nieto Barranco
Managing Director at CRIC

Valerio Grosso
CEO at ICG and R&D Manager at Labor...

Eduard Alabart
RESPONSABLE DE PROGRAMACION DE P.L.C en...

Xavier Cardeña
Marketing and Strategic Development...

Victor Herrero
Project Manager & Research Engineer en...

Josep Perello
Head of Information Technologies and...

Ferruccio Viotti
Responsabile Elettrico ed Automazioni...

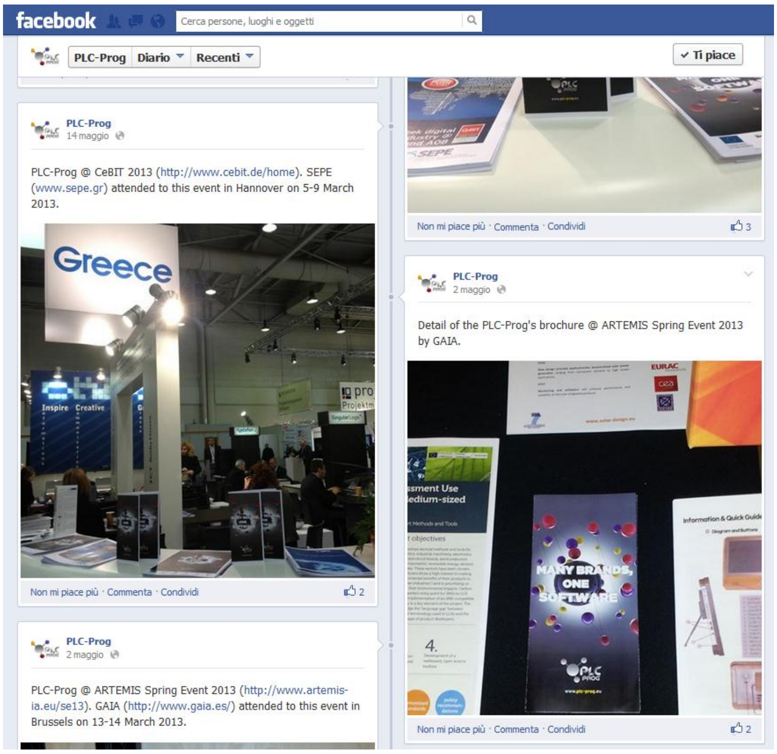
Saioa García
Profesional Independiente: Marketing y...

infoPLC.net - David García
Responsable infoPLC.net / Gerencia en...

Patricia Masip García

LinkedIn Company Pages
Stay up-to-date on company news, industry trends, and job opportunities.

Fig. 53 - Extract from the dedicated LinkedIn Page



PLC-Prog
14 maggio

PLC-Prog @ CeBIT 2013 (http://www.cebit.de/home). SEPE (www.sepe.gr) attended to this event in Hannover on 5-9 March 2013.

Non mi piace più · Commenta · Condividi 2

PLC-Prog
2 maggio

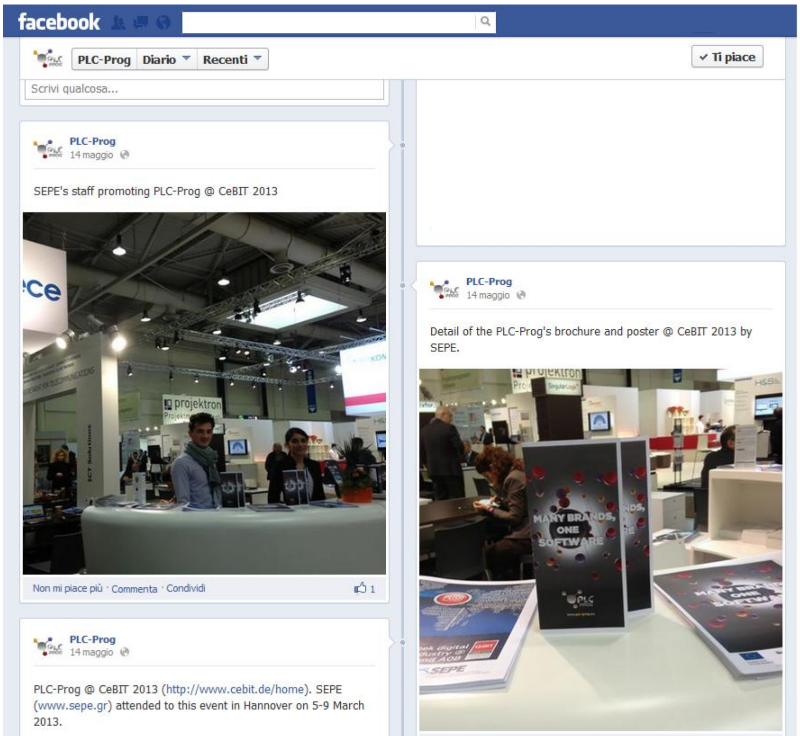
PLC-Prog @ ARTEMIS Spring Event 2013 (http://www.artemis-ia.eu/se13). GAIA (http://www.gaia.es/) attended to this event in Brussels on 13-14 March 2013.

Non mi piace più · Commenta · Condividi 3

PLC-Prog
2 maggio

Detail of the PLC-Prog's brochure @ ARTEMIS Spring Event 2013 by GAIA.

Non mi piace più · Commenta · Condividi 2



PLC-Prog
14 maggio

SEPE's staff promoting PLC-Prog @ CeBIT 2013

Non mi piace più · Commenta · Condividi 1

PLC-Prog
14 maggio

PLC-Prog @ CeBIT 2013 (http://www.cebit.de/home). SEPE (www.sepe.gr) attended to this event in Hannover on 5-9 March 2013.

PLC-Prog
14 maggio

Detail of the PLC-Prog's brochure and poster @ CeBIT 2013 by SEPE.

Fig. 54 - PLC-PROG Facebook Page

4.3. Exploitation of results

Exploitable Knowledge (description)	Exploitable product(s) or measure(s)	Sector(s) of application	Time for Commercial use	Patents or other IPR protection	Owner & Other Partners involved
PLC Application	An integrated framework for PLC programming development	Industrial automation, plastics and moulding industry	2016	Copyright and Trade Mark registration evaluated and sought after the end of the project	AGs and SMEs
PLC Code Generator	On the PLC side the system translates the graphical view into generated code in a *.txt* format, which comes to be adapted in a second time by the post-processors for the chosen brands	ICT companies	2015	Copyright to be sought after the end of the project	AGs and SMEs

Exploitable Knowledge (description)	Exploitable product(s) or measure(s)	Sector(s) of application	Time for Commercial use	Patents or other IPR protection	Owner & Other Partners involved
Post Compilers	the Post Compilers compile the .txt into the machine code used the by the specific brand compiler	ICT and software development companies	2015	Copyright to be evaluated and sought after the end of the project	AGs and SMEs

4.3.1. PLC Application

Description of result

This result is a comprehensive assembly of the basic modules for PLC development and of the SCADA OO modules for communication. The complete PLC-PROG system has been tested and validated during the project.

The whole project is based on the general objective of developing an *integrated framework for PLC programming development*.

This is carried out by means of a number of intermediate objectives that will constitute a unique technological solution:

- Development of a PLC programming framework valid for a number of PLC and SCADA (Supervisory Control and Data Acquisition) brands.

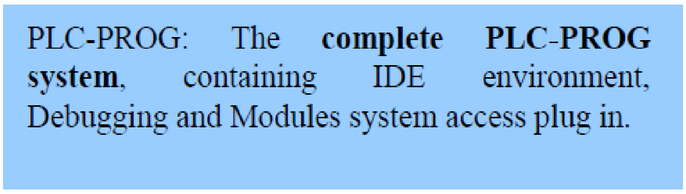
This includes:

i) the development of a graphical approach to developing and documenting PLC programs in an intuitive manner;

ii) the development of post-processors for translating a PLC-PROG project into native PLC code for 3 major PLC brands proving compliant with the IEC-61131 standard;

iii) the development of post-processors for integrating the PLC-PROG project into 3 major SCADA commercial programs.

- Development of the basic virtual modules for proper PLC software development. The number of modules will be determined (no less than 20 for each of the test cases), and, by the beginning of the third year, a number of basic PLC development modules will be included in the PLC-PROG system.
- Development of Supervisory Control and Data Acquisition modules (OO-SCADA): these activities have been organized so as to generate the modules for communication between PLC and SCADA. In this way, communication and data integrity between SCADA and PLC are granted.
- Validation of the quality of the system. This step will be described after the validation has started, in the second draft of this document.



PLC-PROG: The **complete PLC-PROG system**, containing IDE environment, Debugging and Modules system access plug in.

Fig. 55 - Result 1- the PLC-PROG system

Possible exploitation

As already stated in the previous version, the PLC-PROG system has an outstanding innovation character with respect to state-of-the-art PLC programming tools and processes in use in the industrial sectors. The expected exploitation for this framework sees the PLCs of industrial machines programmed by means of PLC-PROG. The integral control developed of this machine has been tested, including the control and optimisation of the automated process, the acquisition and storage of the main process parameters to optimise it, the man-machine interface to operate and supervise the machine both locally and remotely, and the communication with factory wide intranets to exchange information with quality. The exploitation strategy for such a complete and commercially valuable result has been discussed with the whole partnership.

At this stage, it is agreed that both SMEs and the SME-AGs partners within PLC-PROG will be involved in the exploitation of this result - owned by the Associations, while the RTDs will have a supporting role only for the technical related aspects.

The possibility to **patent** this result in the EU countries of the Consortium has been evaluated. However, all the main results in the field of both PCs and PLCs programming, like the PLC-PROG software, are not considered patentable results in the EU.

The only country allowing to patent a software application is USA, which is, at this stage, out of the project target, and the procedure proves highly expensive; this is main reason why the Consortium members will not take into consideration the application for a patent on the PLC developed application at least a very preliminary exploitation stage.

However, **copyright enforcement and trade mark registration** have been proposed for PLC-PROG applications; a deep analysis of the effective functioning and comfort in the use of the programming environment have been performed and the following assessment led to a clearer context for copyright/trade mark registration actions. Stated this, the interest of the SMEs/SME AGs regarding this action is high and has been explicitly discussed during the exploitation sessions held during this last year of the project.

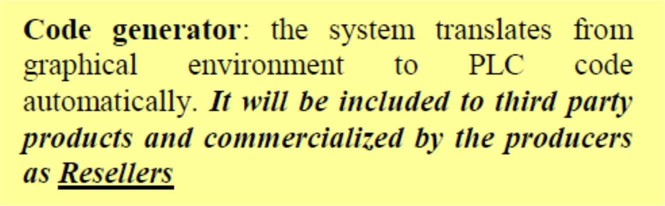
4.3.2. PLC-PROG Code Generator

Description of result

The post-processors allows to translate a PLC-PROG project into a native PLC code for the three main brands of PLC that proved compliant with the European IEC-61131 standard. The development of predefined templates for the post-processors have been finalized during the third year of the project.

The structure of PLC-PROG system has been described in principle in the documents reporting about the architecture, the functioning and the PLC programs development schematics.

The code generator plays an important role in the functioning of this innovative system: in fact, the development of the program starts in the IDE with a graphical approach. On the PLC side the system translates the graphical view into generated code in a *.txt* format, which comes to be adapted in a second time by the post-processors for the chosen brands.



Code generator: the system translates from graphical environment to PLC code automatically. ***It will be included to third party products and commercialized by the producers as Resellers***

Fig. 56 - Result 2- The Code Generator

Possible exploitation

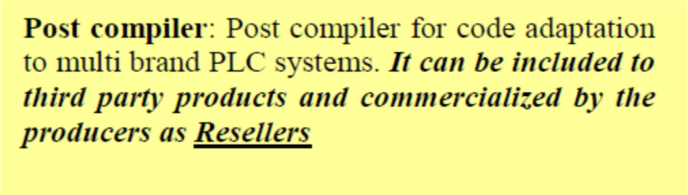
The project partners evaluated the possibility to protect this software result by **Copyright**, so as to prevent any part of PLC-PROG programming framework from being copied or replicated.

At the moment, no action has been undertaken for this protection; application and evaluation of the **copyright application process** for this result was already performed during the second year of activity, and it was agreed among the partners to further investigate how the code generator in PLC-PROG could be protected behind the end of the project.

4.3.3. PLC-PROG Post Compilers

Description of result

The Multi-Brand Compiler System, once the code has been generated from the Generator described in Result no. 2, allows to adapt the *.txt* to each specific brand characteristics, and this programming list is finally compiled into the machine code by the specific brand compiler.



Post compiler: Post compiler for code adaptation to multi brand PLC systems. ***It can be included to third party products and commercialized by the producers as Resellers***

Fig. 57 - Result 3- The post compiler

Possible exploitation

The Multi-Brand Compiler System, as well as the previous two results of PLC-PROG, has been planned to be protected by **Copyright**, even if no action has been undertaken for this protection. Indeed, the relative application have been agreed among the partners, and further investigation of how this result could be protected has been planned to be carried out after the end of the project.

4.3.4. Exploitation Strategy

In order to support and justify the strategy designed for the exploitation of the PLC-PROG project results, it is worth to preface as the current main beneficiaries (both AGs and SMEs involved) confirmed strongly own belief and, therefore direct commitment, about the relevant impact achievable by the adoption and deployment of PLC-PROG system, mainly in terms of increase in SMEs' competitiveness, economic benefits for the members of the Consortium and impact on standards. In particular:

- ✓ PLC-PROG system will provide the SMEs that use automation with a cost effective, practical PLC programming tool, **optimising resources** and resulting in **increased profitability and competitiveness**.
- ✓ the PLC-PROG system will not only **hasten the development cycle,** but mostly reduce the errors and number of "bugs" that are almost inherent in modern software development by allowing the use of pre-built and fully tested objects (or modules).
- ✓ it will also significantly **enhance the knowledge-base** with the use of the OO approach and improve the efficiency and productivity of automation engineers, reducing programming times and installation errors.
- ✓ **Brand-independence** feature will allow partner SME end-users to more easily update their facilities by taking advantage of new, lower cost and more productive PLC models launched in the market by any brand, resulting in better price-performance ratios.

In this vision, **the main interest for the 3 Associations involved: GAIA, SEPE and BPF; is the opportunity to offer to their members alternative programming products and their "customised" applications** *(additional and specific functions, modules and libraries)* **as well as a set of additional and professional services** aimed at assuring an high level of customers/users satisfactions *(such as: technical support and assistance, coaching and training sessions, etc...)*, besides a continuous updating about the most relevant outcomes and finding coming from the several PLC domains.

On the other hand, the **SMEs currently involved will also benefit directly from the adoption of the PLC-PROG system which will allow them to gain competitiveness and to increase profitability** by improving PLC performance, providing support through operator specific information and graphic design accessibility, lowering operating costs and enhancing maintenance management. Moreover, this other main category of beneficiary will be actively involved in the exploitation strategy by providing own technical expertises for supporting the SME AGs' provision and, therefore, to assure the most completed set of capabilities in the PLC programming and application fields.

Based on these results' exploitation purposes, the entire Consortium carried out an assessment analysis of the main results planned in order to define an exploitation strategy aimed at valorising all capabilities and expertises available in a valuable way as well as to assure a prosecution of the activities even behind the end of the project.

As main outcome of this process, the **3 SMEs Associations** assessed the approach which foresees to set up a clusterised new entity which **will be able to act as main interface among the industrial environment *(demand side)* and the research and scientific community *(supply side)*.** In particular, this PLC-PROG cluster activities shall be driven by promoting the scientific and technical exchange, and information between universities, enterprises and common final users in the field of PLC programming and industrial automation as well as, the maintenance and further development of the programming platform resulting from the research project.

Based on the main outcomes coming from the market search and business analysis performed during the project, the Consortium to evaluated the most effective legal entity to be established in order to assure the highest level of project results valorisation and partners' benefits.

In particular, a trade off analysis was provided between 2 main business legal forms: **the Joint Venture *(mostly a Co-operative Joint Venture)* and the Associations Cluster *(mostly based on the establishment of Framework Agreement)*** with the aim to address the final choice toward the type of legal entity able to assure the implementation of the most profitable commercialized strategy for the set of products identified by the PLC-PROG partners.

At this regard, and based on the main interactions occurred among the main beneficiaries as well as all the legal constraints considered, it is possible to anticipate that the most suitable option at this stage is the establishment of a **Base Framework Agreement,** reflecting the way by which the PLC-PROG system will be commercialized and, therefore, the business agreements to be defined among the AGs (co-owners of the results) and the associated SMEs (main licensees), which will act as technical services providers.

As following step, the entire Consortium worked on the design of a valuable **business model** *(refer to the PLC-PROG exploitation strategy diagram)* identifying the business and cooperation flows among the main beneficiaries and their relative type of interaction (agreements) and on the basis of the following objectives/assumptions[7],

- Maximising the expected benefits coming from development and adoption of the PLC-PROG platform by putting in place all the expertises and knowledge available within the Consortium.
- Creating a cooperative working framework involving all the partners, even behind the end of the research project, with the aim to confer to the SME AGs an active exploitation role and, therefore, to give the primary opportunity to their SMEs to acquire relevant technical and commercial advantages in own market segments.
- Establishing the Quality Assurance programs and Perform Quality Control of the promoted Open Source system, and its further extensions.
- Maintaining and updating the guidelines for using the PLC Open Source License and closely monitoring of the market for compliance and unauthorised use of the PLC-PROG programming middleware.
- Supporting the establishment and the expansion of a PLC community (made up of final users, software developers, industrial and research organizations) in order to increase the

[7] These objectives must be aligned with the final objectives as will agree in the final version of D8.4 and D8.11.

promotional lever of the PLC-PROG results as well as a wide use and adoption of the system developed.

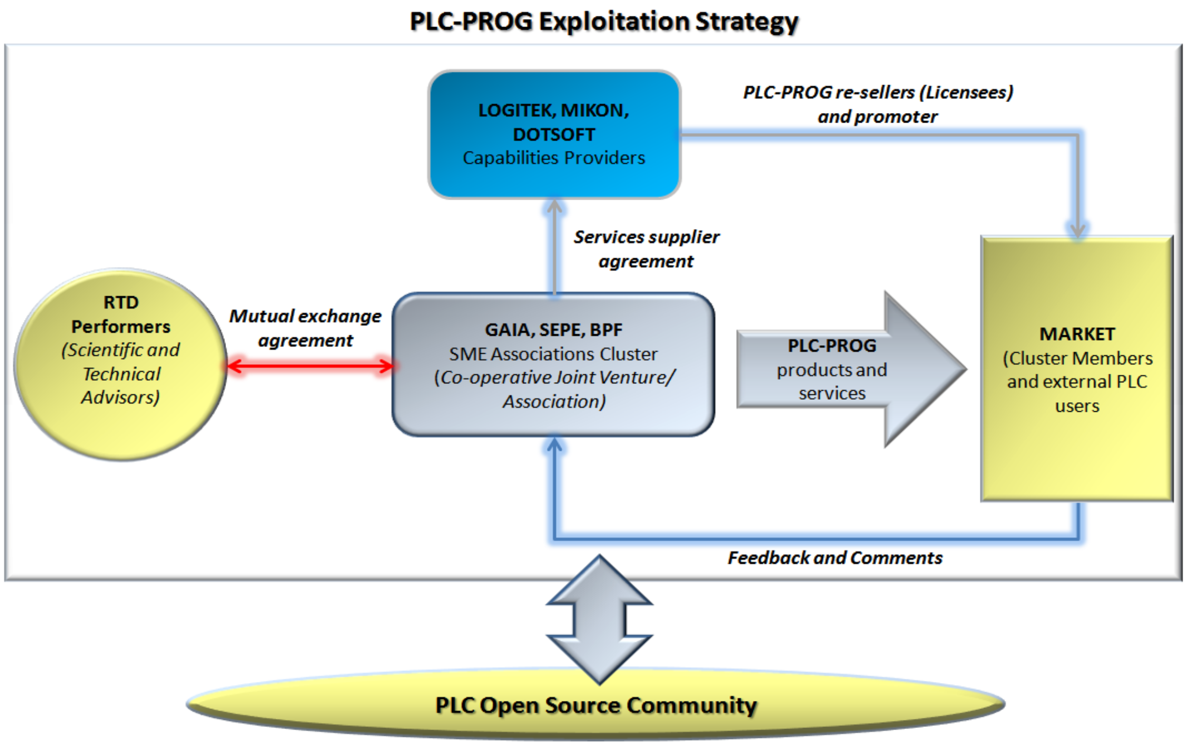

PLC-PROG Exploitation Strategy
LOGITEK, MIKON, DOTSOFT
Capabilities Providers
PLC-PROG re-sellers (Licensees) and promoter
Services supplier agreement
RTD Performers (Scientific and Technical Advisors)
Mutual exchange agreement
GAIA, SEPE, BPF
SME Associations Cluster
(Co-operative Joint Venture/ Association)
PLC-PROG products and services
MARKET
(Cluster Members and external PLC users
Feedback and Comments
PLC Open Source Community

Fig. 58 – PLC-PROG exploitation strategy

At this scope, it is worth to underline that:

- the design process has been carried out in order **to assure a completely coverage all the needed role and expertise identified in the targeted value chain** *(from the production to the customer services operations)* aimed to the commercialization and diffusion of the PLC-PROG system implemented;
- the **business model** obtained **represents the optimal condition to exploit the project results** taking into account only the expertise and capabilities of the involved partners *(including the RTD Performers);*

- the evaluation **to get involved other Third Parties** as services provider ***(mainly referring to the commercialization and maintenance operations)*** after the end of project is also in considered in order to assure a proper starting-up of the activities related to the distribution and commercialization of the product and services defined.

Route for Commercialization and Promotion

In this context, it is worth to underline as the proposed exploitation model has been designed according to a specific distribution model (Open Source) selected by the Associations as well as a well composed set of PLC-PROG product and services:

1. **PLC-PROG system**: it will be offered and distributed in open source modality to the members of the PLC-PROG cluster and external users, and it includes the IDE, all the libraries and functionalities implemented during and behind the project. At this stage, 2 potential commercialisation scenarios have been identified according to the needs of the potential users/client:

    I. **PLC-PROG cluster's member**: this type of user involves all the current members of the 3 Associations and the new ones will decide to join in. These members could make an unlimited use of system (and its potential updates and new components) within the same current subscription fees already paid to own Association through the acceptance of the use license's terms. In additional, they will have available a professional assistance service, training material, free participation at courses and coaching workshops, reduced consulting fees for the development of specific SW components or libraries.

    II. **PLC external users**: this category involves all the users will decide to not join in the AGs but are interested to use the system in any case for own specific needs. In this scenario the AGs will provide a customised offer of products services against the payment of subscription fees. These fees will be defined on the temporal basis (i.e. annual fee) or in relation to the number of sectorial functionalities and libraries requested (i.e. wood, plastic, metal, etc...); they will also include the relative assistance services as well as training material and participation at courses/workshops.

2. **PLC-PROG professional services:** since the users of PLC-PROG could need to request the development of "customised" components or the provision of technical and training consultancies for their business scopes and operation field, the AGs, with the support of the SMEs involved, will design and provide a set of professional consulting services on demand, mainly aiming at:

   I. **developing specific SW components** and sectorial **libraries**, including the opportunity to make use of a dedicated technical assistance and maintenance services:

      - consultancy, tech support and certification of engineers will be available through specialized dedicated companies and it includes the validation of new objects, design and best practices, and support on applications;
      - tech support includes online and offline 1-2-3 levels of support on product;
      - centers must include competence with laboratory with the different PLCs supported.

   II. organizing targeted **training courses** or **coaching workshops**, including the provision of relative material. The Training is necessary for the proper use of PLC-PROG and it can be done at 2 levels:

      - **Basic Level** (1 day)
        - This is a 1-day basic training to start with PLC-PROG
      - **Certified Training Course** (2 days)
        - Certification is necessary to ensure quality of the projects for the end users
        - This training includes an exam
        - Certification promotes PLC-PROG community

- A list of certified engineers and companies will be published
- A Certification program, materials and promotion must be created

At this purpose, a proper assessment was provided by the Consortium with the aim to define a valuable **pricing strategy** on basis of the exploitation and distribution strategy identified for the future commercialization and distribution of PLC-PROG products and services; in particular, the following preliminary assumption were agreed:

Basic PLC-PROG version	○ Free version ○ Includes only IDE ○ Limited in functionality ○ No certification ○ No support ○ Designed for testing and evaluation ○ Valid for 1 year renewable
Professional PLC-PROG version	○ Includes IDE + PostProcessor + SCADA ○ Annual price under contract ○ 300 € year for 1 user ○ 500 € for > 5 users ○ 1000 € year for > 5 users ○ Includes 1 free basic training ○ Includes tech support ○ Valid for 1 year renewable
PLC-PROG Corporative version	○ For maintenance purposes ○ Only for end users ○ No date limit ○ Price: tbd

Basic Training	○ 1 day ○ 200 € ○ Includes training materials
Certified Training	○ 2 days ○ 800 € ○ Includes training materials

The strategic decision to adopt an open source software marketing and distribution channels will greatly increase the like hood that PLC-PROG reaches the broadest market possible, and levels the playing field with the larger, better-heeled competitors, whilst, as detailed above, the revenue model for AGs will be one based on a service revenue stream rather than a licence revenue stream (e.g. specific program extensions or technical support for PLC-PROG end-users). In fact, the PLC-PROG open source software will be copyrighted, but released under licences which allow free re-distribution.

Moreover, a proper trade mark analysis will be carried out after PLC-PROG project, and initial distribution will take place through the partners' existing customer base, as well as through contacts developed during the dissemination activities of the project. At this purpose, it is worth to underline as the efficient provision of the defined products and services will be assured by the PLC-PROG AG’s beneficiary through the creation of a proper web platform which will act as main interface of dialogue and distribution channel with associates and external clients.

The operational management and the maintenance of this platform, as well as the development of its further improvements and functionalities, will be handled by the AGs directly, which could also count on the technical capabilities and expertises assured by the SMEs (LOGITEK, MIKON, DOTSOFT), as well as on the scientific advises coming from the group of RTD Performers.

Type of Exploitable Foreground[8]	Description of exploitable foreground	Confidential Click on YES/NO	Foreseen embargo date dd/mm/yyyy	Exploitable product(s) or measure(s)	Sector(s) of application[9]	Timetable, commercial or any other use	Patents or other IPR exploitation (licences)	Owner & Other Beneficiary(s) involved
Commercial exploitation of R&D results	PLC-PROG TOOL	NO	30/09/2013	IDE CODE POSTPROCESS OR SCADA	A3.2.2 - Freshwater aquaculture C28 - Manufacture of machinery and equipment n.e.c. C31 - Manufacture of furniture E - Water supply; sewerage; waste managment and	2014 ONWARDS	IT WILL BE SOLD THOUGH LICENCES	GAIA (OWNER) SEPE (OWNER) BPF (OWNER) LOGITEK (PRIMARY USER AND RE-SELLER) MIKON (PRIMARY USER AND RE-SELLER) DOTSOFT (PRIMARY USER AND RE-SELLER)

[19] A drop down list allows choosing the type of foreground: General advancement of knowledge, Commercial exploitation of R&D results, Exploitation of R&D results via standards, exploitation of results through EU policies, exploitation of results through (social) innovation.

[9] A drop down list allows choosing the type sector (NACE nomenclature) : http://ec.europa.eu/competition/mergers/cases/index/nace_all.html

Type of Exploitable Foreground[8]	Description of exploitable foreground	Confidential Click on YES/NO	Foreseen embargo date dd/mm/yyyy	Exploitable product(s) or measure(s)	Sector(s) of application[9]	Timetable, commercial or any other use	Patents or other IPR exploitation (licences)	Owner & Other Beneficiary(s) involved
					remediation activities			
Commercial exploitation of R&D results	TRAINING COURSE AND MATERIAL RELATED TO PLC PROG	NO	30/09/2013	TRAINING	P - Education	2014 ONWARDS	A SERVICE WHICH WILL BE OFFERED TOGETHER WITH THE MAIN PRODUCT, THE PLC PROG TOOL	(OWNER) SEPE (OWNER) BPF (OWNER) LOGITEK (PRIMARY USER AND RE-SELLER) MIKON (PRIMARY USER AND RE-SELLER) DOTSOFT (PRIMARY USER AND RE-SELLER)
Commercial exploitation of R&D results	CONSULTANCY SERVICES RELATED TO PLC PROG TOOL	NO	30/09/2013	CONSULTANCY SERVICES	M73 - Advertising and market research N - Administrative and support service activities	2014 ONWARDS	A SERVICE WHICH WILL BE OFFERED TOGETHER WITH THE MAIN PRODUCT, THE PLC PROG TOOL	GAIA(OWNER) SEPE (OWNER) BPF (OWNER) LOGITEK (PRIMARY USER AND RE-SELLER) MIKON (PRIMARY USER AND RE-SELLER) DOTSOFT (PRIMARY USER AND RE-SELLER)

5. Public website address and relevant contact details.

Project website: www.plc-prog.eu

Coordinator details

Name: Mr. Valerio Grosso

Organisation: Labor Srl

Address: Via Giacomo Peroni 386- 00131 Roma (Italy)

Web site: www.labor-eu.net

Tel: +39 06 40040354

Fax+39 06 40040357

E-mail: v.grosso@labor-roma.it

SME Associations participants:

- Association of Industries For Elextronic and Information Technologies in the Basque Country - GAIA (SPAIN) http://www.gaia.es
- Federation of Hellenic Information Technology and Communication Entreprises – SEPE (GREECE) http://www.sepe.gr
- The British Plastics Federation – BPF (UK) http://www.bpf.co.uk

SME participants:

- MIKON SYSTEMS (ROMANIA) http://www.mikon.ro/
- LOGITEK SA (SPAIN) http://www.logiteksa.com
- DOTSOFT OLOKLIROMENES EFARMOGES DIADIKTIOY KAI VASEON DEDOMENOM AE (GREECE) http://www.dotsoft.gr

RTD Performers:

- LABOR SRL (ITALY) www.labor-eu.net
- UNIVERSIDAD POLITECNICA DE VALENCIA (SPAIN) http://www.upv.es/
- CENTRE DE RECERCA I INVESTIGACIO DE CATALUNYA S.A. (SPAIN) http://www.cric.cat