BEMO-COFRA

Brazil-Europe Monitoring and Control Framework

(Project No. 288133)

D5.1.2 Final infrastructure for distributed control logic

Published by the BEMO-COFRA Consortium
Dissemination Level: Public

Project co-funded by the European Commission within the 7th Framework Programme
and
Conselho Nacional de Desenvolvimento Científico e Tecnológico
Objective ICT-2011-EU-Brazil

Document control page

Document file: D5.1.2 Final infrastructure for distributed control logic v1.0
Document version: 1.0
Document owner: Juliana Vilela (COMAU)

Work package: WP5 – Distributed Control Logic and Enabling Features
Task: T5.1 – Distributed Control Logic
Deliverable type: P

Document status: ☒ approved by the document owner for internal review
☐ approved for submission to the EC

Document history:

Version	Author(s)	Date	Summary of changes made
0.1	Juliana Vilela	30-09-13	ToC.
0.2	Juliana Vilela	18-11-13	Chp 1,2 and 3.
0.3	Juliana Vilela	19-11-13	Chp 4.
0.9	Juliana Vilela	21-11-13	Chp 5 and 6.
1.0	Juliana Vilela	27-11-13	Ready for submission.

Internal review history:

Reviewed by	Date	Summary of comments
Ferry	22-11-13	Approved with comments.

Index:

Figures

1. Executive summary

The presented deliverable aims show the methodology used at the control logic development of the BEMO-COFRA project as an update of the previous deliverable D5.1.1 Initial infrastructure for distributed control logic.

At this deliverable the Supervisory Control Theory (Wonham, 1989) is presented as a solution to the empirical methods vastly used at the industries for the control logic development. Also a brief discussion about the difference of decentralized and distributed control is showed to clarify and ensure the distributed aspects of the methodology presented.

As an example all the development of the final demo control logic is showed in all steps.

2. Introduction

This deliverable is an update of the previous deliverable D5.1.1 Initial infrastructure for distributed control logic, explaining all the main points and methodologies used at the control logic development.

2.1 Background

The BEMO-COFRA project aims to develop an innovative distributed framework which allows networked monitoring and control of large-scale complex systems by integrating heterogeneous smart objects, legacy devices and sub-systems, cooperating to support holistic management and to achieve overall system efficiency with respect to energy and raw materials.

The BEMO-COFRA features a Service oriented Architecture (SoA) and a middleware able to expose smart objects, legacy devices and sub-systems' capabilities by means of web services thus supporting syntactic and semantic interoperability among different technologies coexisting in the overall monitoring and control framework. Wireless Sensor and Actuator Network (WSAN) devices, legacy sub-systems and devices will thus be able to interact and cooperate, orchestrated by a manager in charge of enforcing a distributed logic with the overall monitoring and control network.

BEMO-COFRA reuses the results of the well-reputed Hydra IP and Pobicos STREP and the recently started ebbits IP featuring a Service Oriented Architecture (SOA) and a middleware able to expose smart objects, legacy devices and sub-systems' capabilities by means of web services. Syntactic and semantic interoperability among coexisting technologies in the overall monitoring and control framework is made available.

The integration of heterogeneous smart objects, legacy devices and sub-systems will achieve overall systems' efficiency with respect to energy and raw materials and support holistic management. The BEMO-COFRA project will address both technological aspects and user needs to promote a wider adoption of large-scale networked monitoring and control solutions.

2.2 Purpose, context and scope of this deliverable

This deliverable is composed by 5 chapters:

1- Executive Summary;

2- Introduction;

3- Distributed control logic: a brief discussion of the difference between distributed control logic and decentralized control logic;

4- Discrete event systems: an explanation of the discrete event systems methodology to develop a control logic design.

5- Development: the steps of the control logic development for the final demo scenario.

3. Distributed Control Logic

The distributed control system or DCS optimize the process control, reduce the costs and improve the products quality and process security. This system is composed by several devices that have complementary functions. Redundant networks allow the data processing decentralization by remote units. Also, the DCS gives a human-machine interface (HMI), programmable logic controllers (PLC) and communication devices over all the plant in a network. The Processing Units, distributed along the plant process the signal from the field devices into information according to the control logic programmed.

The DCS is composed by three basic elements:

1- Process interface: control unit and data acquisition;

2- Human-machine Interface: HMI;

3- Data network (Data Highway).

The figure 1 shows the basic structure for a DCS.

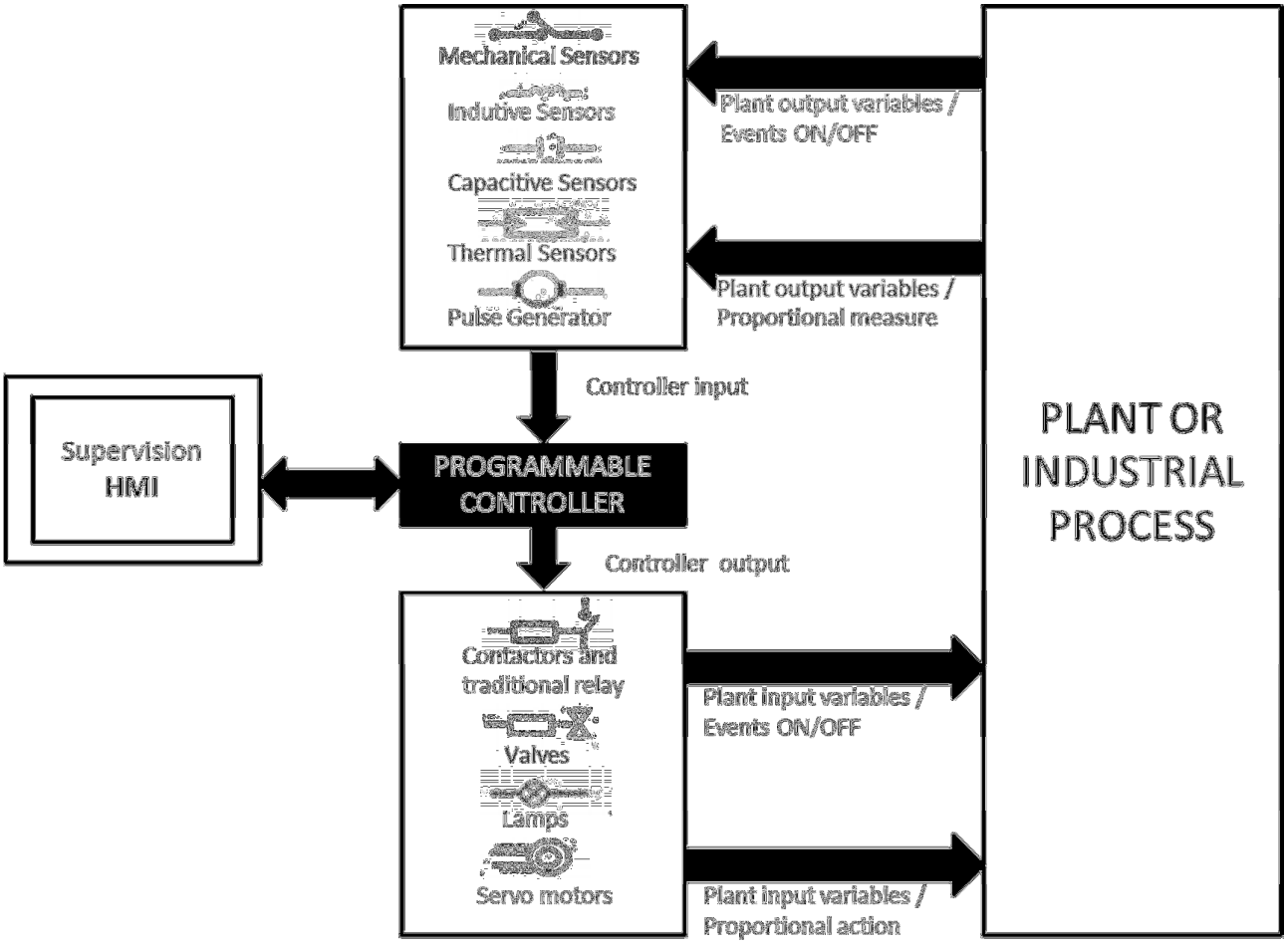


Figure 1: Basic Structure for a Distributed Control System.

The control systems can be cathegorized in two main classes : centralized and decentralized. The main differences between them are showed on the table bellow.

Centralized control system	Decentralized control system
- One general supervision room connected to the PLC's - Inter-communication with the corporate network & provides an on-line information - Data consistency - High costs with cables - System availability = central computer availability - High difficulty to amplify the system -> complexity - Single point of failure	- System with several sensors and actuators - Different sampling rates - Synchronization problems - Sensor redundancy - Data fusion (sensors integration) - Nodes with processing abilities - Communication node to node - Without central processing

It is very important discern between distributed control and decentralized control. A huge mistake is made when we talk about these two kinds of control systems. Bellow the differences between this two systems are explicated:

Distributed Control:

- Refer to the control loop topology;
- Sensor, controller and actuator are physically distributed all over the plant;
- Typically the loop has 1 sensor, 1 actuator and 1 controller.

Decentralized Control:

- Refer to the controller topology;
- The control is implemented in a distributed way, whit local data processing;
- Typically there are several sensors and actuator at the same control loop.

4. Discrete Event Systems

Nowadays, the automation has a high importance within the industries and issues with the control of automated systems has been turned more complex. The traditional empirical methods based on the programmer experience can result in inappropriate and inefficient control solutions. In this ways, the Supervisory Control Theory (SCT) showed by Ramadge and Wonham (1989) seems an adequate tool once guarantee an optimized control logic (minimally restrictive and no blocking) and satisfies the control specifications. The use of a formal methodology to implement control systems allows a standardization of the development, tests and structure of PLC's programs and solutions free of mistakes, blocks and away from empirical methods.

The Supervisory Control Theory is composed by procedures and methods to controllers' synthesis. Using the SCT, an optimized control solution is obtained in a systematic way. The complexity of the actual industries and plants makes very hard the designer of a monolithic controller even using the SCT as a methodology. So, the local modular approach in SCT appears as a better methodology that explores the plant modularity. Thus, when it is necessary change the control logic, for inclusion or exclusion of devices or changes at the layout, this change is facilitated once it is necessary change only the controllers of the part modified.

The SCT has a good acceptance at the academia but it is rarely used in the industrial applications. The reason of it is the physical implementation problem. The PLCs are based on signals processed cyclically and the SCT works with discrete events on time. But, many approaches were developed to solve these problems and some of them will be showed in this chapter.

4.1 Theory of Supervisory Control

At the control languages approach proposed by Ramadge and Wonham (1989), the plant to control is modelled by a generator G. The generator is an automaton $G = (\Sigma, Q, \delta, q_0, Q_m)$, where Σ is an event alphabet, Q is a state set, $q_0 \in Q$ is the initial state, $Q_m \subseteq Q$ is set of marked events and $\delta: \Sigma \times Q \rightarrow Q$ the transition function, a partial function defined in each state of Q for a subset of Σ. Being Σ^* the set of all finite words of Σ, including the null word ε. Thus, G is characterized by two subsets of Σ^* called generated language of G (all the words that the plant can generate), denote by $L(G)$, and marked language of G (words that represent complete tasks), denote by $L_m(G)$.

Controllable events $\Sigma_c \subseteq \Sigma$ are the ones that can be activated or disabled by external agents. Uncontrollable events $\Sigma_u \subseteq \Sigma$ can't be avoided and because of that are considered always enabled.

A language $K \subseteq L(G)$ is controllable in relation to G if $\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$ (the up trace represents the close-prefix). It means that if an uncontrollable event occurs after a word of K, this new word still remains at the set K. The set of controllable languages of G is represented by $C(M, G) = \{ K | K \subseteq M$ and K is controllable i.r.t $G\}$ and because is closed over union, contains one and only supreme element, called $\mathrm{Sup}C(M, G)$.

Automata can be illustrated by state transition diagrams, which are directed graphs where the nodes represent states and the arrows events. In these diagrams, the marked states are represented by nodes with double circles and the initial state is showed by an arrow. The figure 2 shows an example of an automaton.

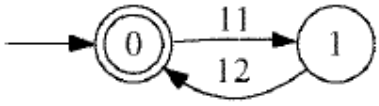


Figure 2: Automaton with 2 states and 2 events.

To develop the control logic, the plant behaviour and the specifications are modelled as events transition diagrams. The supervisors' synthesis methodology explores the fact that manufacturing systems can be modelled as a set of concurrent and asynchronous discrete event systems, this methodology is known as Product System. To represent a system according to the product system methodology from the initial modelling the synchronous subsystems (with events in common) are composed. Being the representation by product systems of a manufacturing plant with subsystems $G_i = (\Sigma_i, Q_i, \delta_i, q_{0i}, Q_{mi})$, $i = 1,..,n$. For $j = 1,..,m$ the local specifications E_{xj} defined on the subset of events $\Sigma_{xj} \subseteq \Sigma$.

The local plant associated to the specification $\mathrm{E_{xj}}$ is defined by $G_{xj} = (\Sigma_{xj}, Q_{xj}, \delta_{xj}, q_{0xj}, Q_{mxj})$ and is composed by the synchronous composition of the subsystems of the original modelling that are directly (and indirectly) restricted by E_{xj}.

It is necessary express the local specifications E_{xj}, $j = 1, \dots, m$, according to this local plants G_{xj}, $j = 1, \dots, m$, composing them synchronously $E_{xj} = E_{xj} \parallel L_m(G_{xj})$ to achieve the optimal local modular supervisors or controllers. Thus, the supreme controllable languages $SupC(E_{xj}, G_{xj})$, $j = 1, \dots, m$, are calculated for each pair of local specification and modular controllers. The local modularity of the supervisors (if one supervisor doesn't affect or block other supervisor) can be check composing synchronously all the resulting controllers, and the generated automaton is TRIM (no-blocking). This condition guarantees that the local modular supervisors don't lose performance in relation to the centralized monolithic controller.

4.2 Implementation on the PLC

The structure divided in plant and supervisor to the Supervisory Control Theory allows the system runs correctly, but the implementation of this structure at the PLC is not trivial. Some issues arise during the implementation of the plant and supervisors on the PLC and they are listed below:

Signals and Events: the SCT considers the events as asynchronous and they occur at discrete instants of time, their occurrence determine the transition of the activated states of the supervisors. The PLC processes variables and their values are updated in each scan. Two effects can happen: the avalanche effect and the inability to recognize the events order. The avalanche effect is the occurrence of one event that results in an inappropriate state transitions, what can come out at the loss of the synchrony between the plant and the supervisor implemented. Other question is during the PLC scan time the input variables are updated, so if two input variables change their values during the same scan time, won't be possible know the other of activation. In this case, if the input variables are uncontrollable events the system won't be able to recognize the order of these events.

Causality: The SCT assumes that all events are generated spontaneously by the plant and the supervisors should send the disabling signals of events to the plant like showed at the figure 3. However, at the real world the uncontrollable events don't come fro the physical plant behavior, but from the answers of the plant to the supervisors' command. Thus, the causality problem comes with the question: Who is responsible to generate the controllable events, the supervisor or the plan?

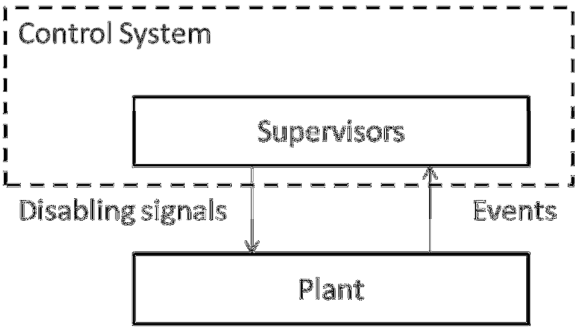


Figure 3: SCT dynamic.

Events Choice: The SCT allows the plant a behaviour with more flexibility with a minimally restrictive supervisor. After the events sequence happens, there could exist many behaviour options for the plant to follow. Thus, when the supervisor is responsible to generate part of the events, he can choose between the choices at one state.

Inaccurate synchronization: During the program execution a change in an input signal may happen and this change only be recognized at the beginning of the next PLC scan cycle. In this case the communication between the plant (devices connected to the PLC) and the PLC is delayed.

Beyond these issues, the abstraction needed to design the plants is considered a problem too. To model the plant you only use the required events by the coordinated tasks, avoiding the space of states explosion and allowing the synthesis of the supervisors, so you should know all the details of the plant and its dynamics before model it.

The implementation of the supervisors at the PLC follows the figure 4. First of all the transitions related to uncontrollable events are programmed, in sequence with all the transitions related to the controllable events, thus at the same scan cycle all the events generated by the plant identified at the input variables can be treated. Following by the update of the states of the plant and the modular supervisors. If a choice problem occur it is necessary observe the disablement set. A specific routine will disable one event of the state, if more than one state are able for it.

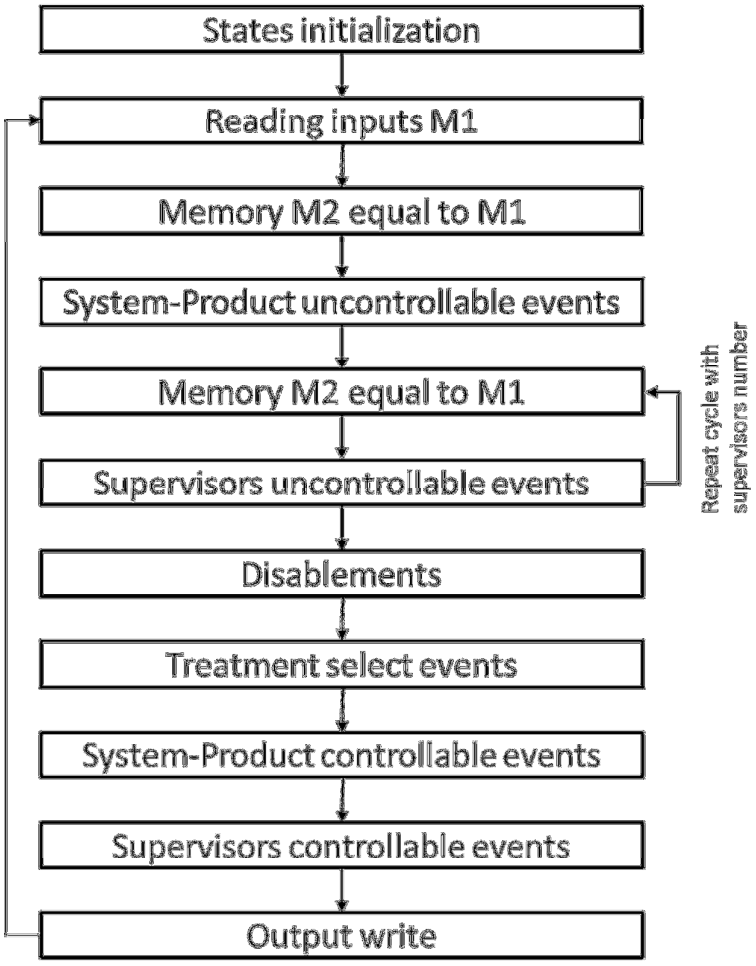


Figure 4: Workflow to the PLC implementation.

5. Development

The chapters above showed all the methodology behind the control logic development to the BEMO-COFRA project. At this chapter, all the steps of the final demo control logic design will be exposed. The final demo scenario were developed based on the use cases that are commonly found in different car factories, so it beholds the generic requirement of the project.

5.1 Final Demo Scenario

For the final demo we will work with a station from the car manufacturing process, which will weld car roofs, preparing them for the whole car assembly. This station has to deal with two different models of roofs, with sunroof and without it, and be able to produce them without a previous production plan. So if at any moment the plant manager receive an order to improve the production of sunroofs, he can only add more items of this roof model at the station, and the BEMO-COFRA will be able to detect this change and set the right configuration to carry on with the production. As an improvement of flexibility, accuracy and quality at this demo the station will be able to check if the weld spots were done or not, if not the station will re-weld them, avoiding the error propagation. Also more wireless sensors will be part of the demo, increasing the wireless traffic, and a wireless actuator will be placed at this station. The consumption of energy will be measured too and showed on time for the user. The figure 5 shows a schema for the final demo.

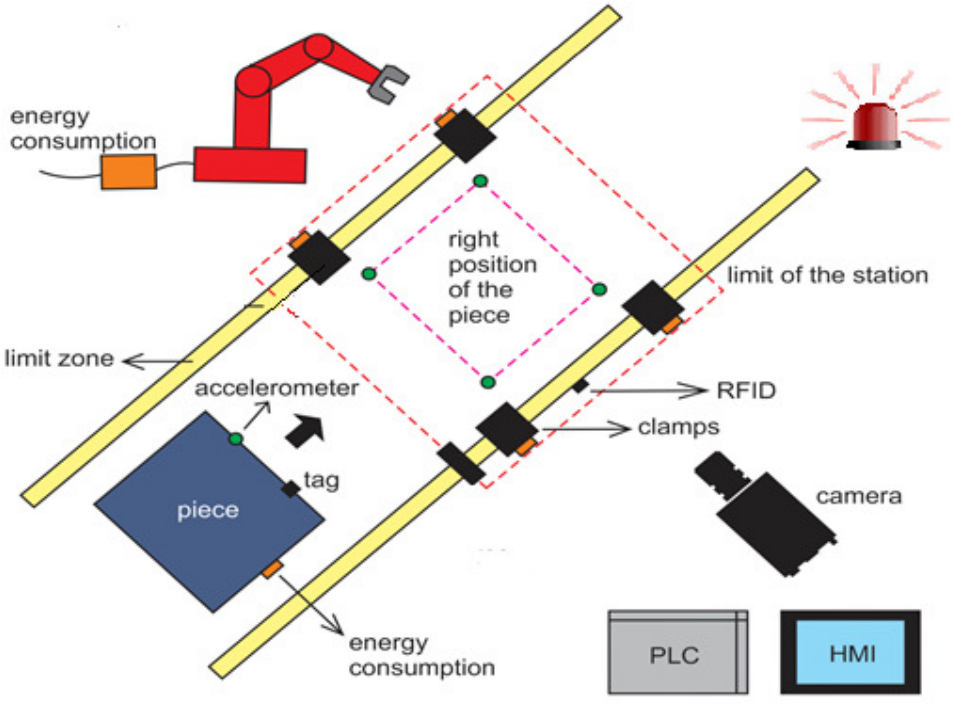


Figure 5: Demo scenario schema.

The workflow showed at the next figures is basically the activity sequence of the final demo. In a summarized description the final demo scenario process is:

-The conveyor brings a new roof piece for the station. The parameters of the conveyor are observed to guarantee its right running. Extra sensors were added to improve the security, so if the SKID, for any reason, go out of a limited area, it is automatically stopped and an alarm is triggered with a red light and a buzzer.

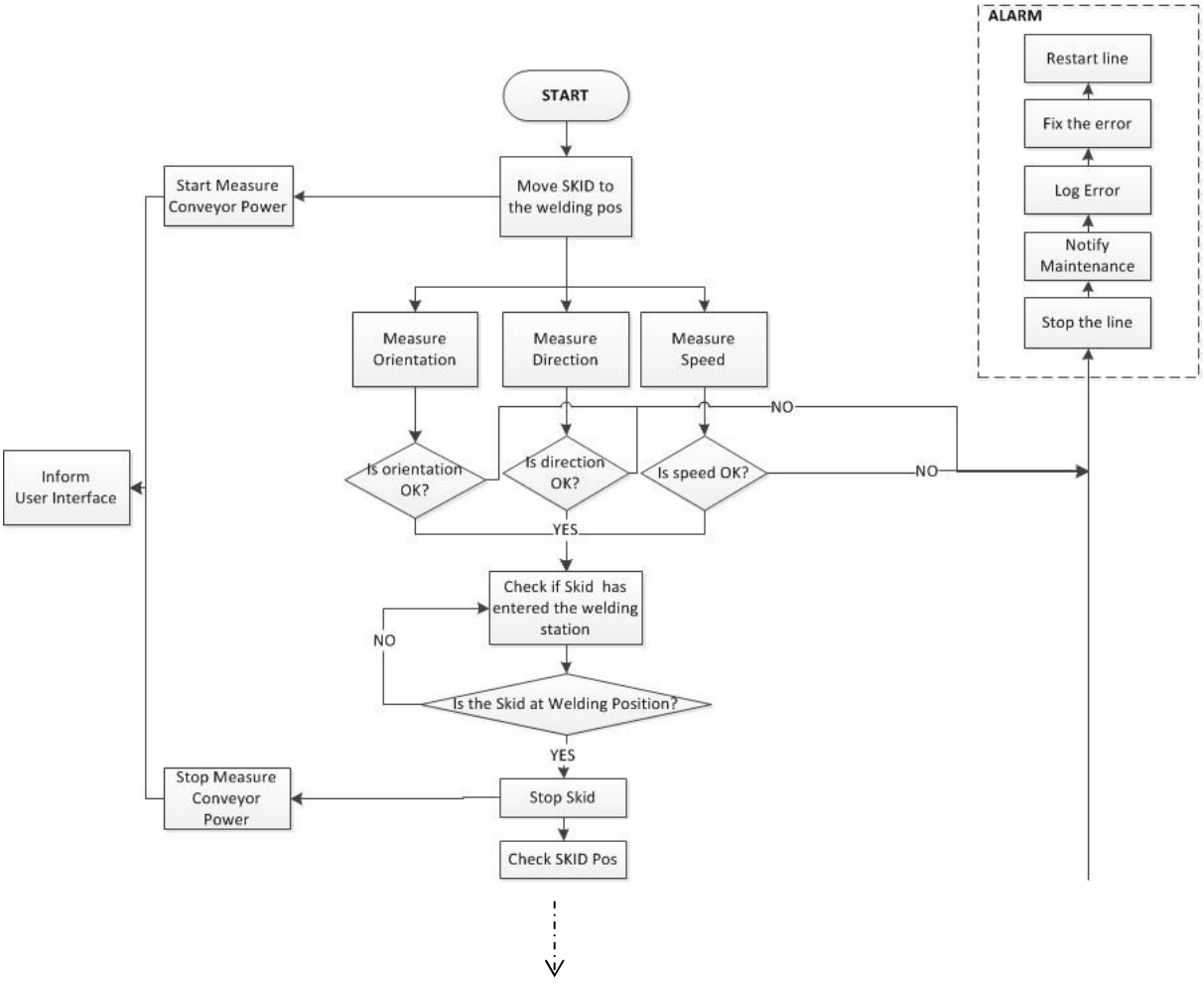
-As soon as the RFID sensor detect the SKID inside the station zone, the SKID stops and the wireless camera check the roof model and if it is at the right position.

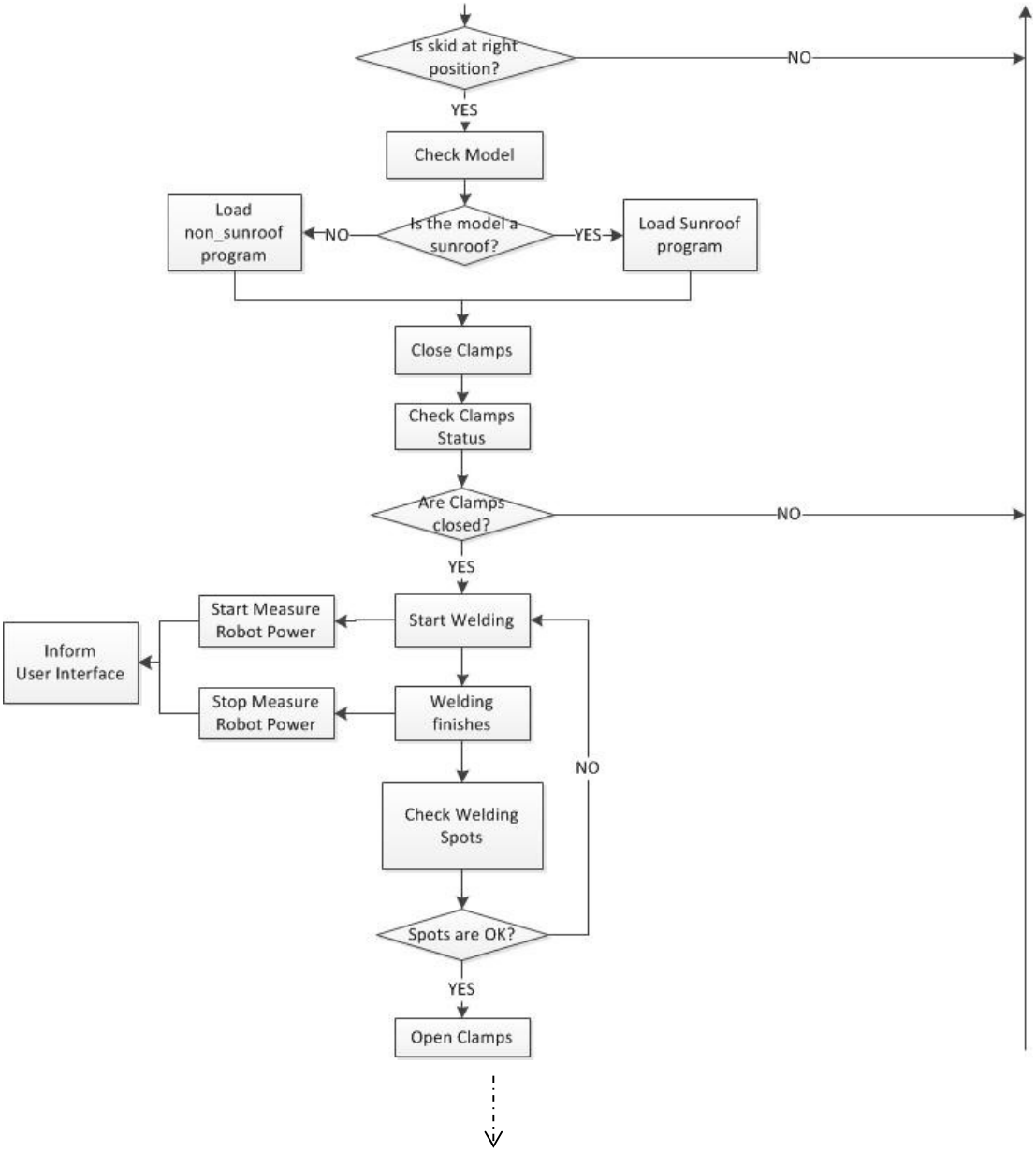
- At this moment the clamps are authorized to close according to the model, actually from this moment all the next procedures are related to the roof model. If an error occur during the clamps closure, the alarm is triggered and the station stops until the error fixture.

-With the piece blocked the robot starts the welding procedure, again if an error occur during this procedure the alarm is triggered.

-Finished the welding the camera will detect if all weld spots were done or not. If not, the robot repeat the weld procedure.

-When the camera detect all the weld spots as done, the clamps are allowed to open and then the SKID can move taking the roof piece for the next station.




Is skid at right position?
NO
YES
Check Model
Load non_sunroof program
NO
Is the model a sunroof?
YES
Load Sunroof program
Close Clamps
Check Clamps Status
Are Clamps closed?
NO
YES
Start Measure Robot Power
Start Welding
Inform User Interface
Stop Measure Robot Power
Welding finishes
NO
Check Welding Spots
Spots are OK?
YES
Open Clamps

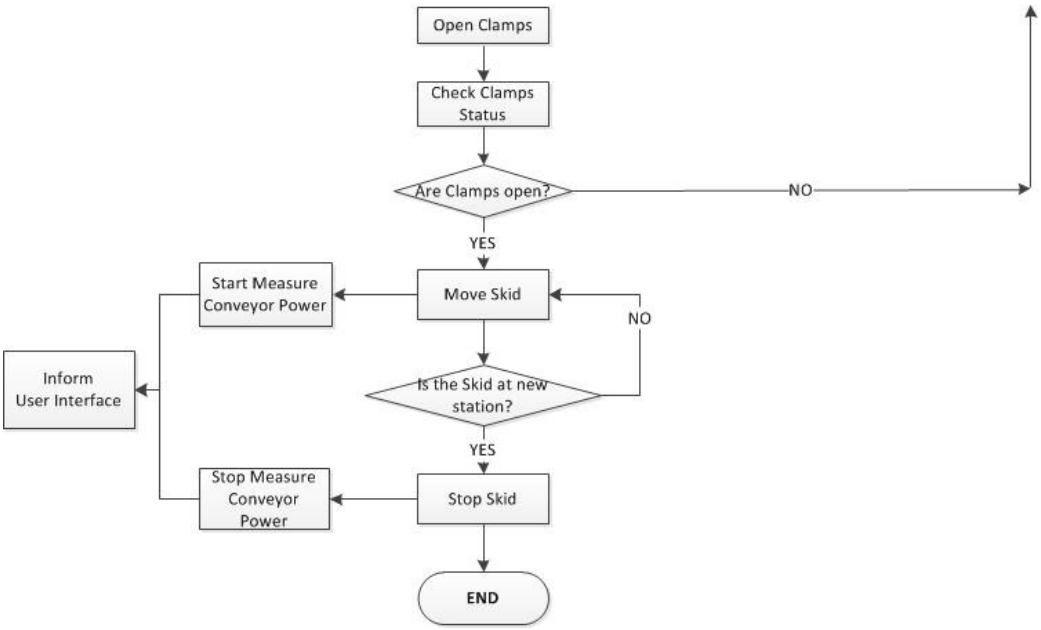


Figure 6: Final demo workflow.

5.2 Plant Models

From the description of the final demo scenario the main devices and events where highlighted to be used on the modelling process. Only the devices related to the control logic will be model. They are:

- Conveyor system: it will be divided in two models, one before the station, Conveyor 1, and other after the station, Conveyor 2.
- Camera: the camera will have two different tasks at the scenario; the first task is to detect the model of the roof and if it is at the right position, model called as Camera 1; the next task is to detect if the weld spots were done or not by the robot, model Camera 2.
- Robot: like the other devices the robot executes two different tasks too. At the first moment it will weld the roof according to the model, Robot 1; next on, if the camera detects missed weld points, the robot redo them, Robot 2.
- Clamps: the clamps were modelled in only one automaton, called Grampos.
- Start: to begin the final demo it's necessary activate the scenario, this will be done with a real or virtual button with the start function.
- RFID: a RFID sensor will be integrated at the plant to identify if the skid guide through the conveyor system reaches the right position at the station.
- Status: at the final demo scenario there are sensors to detect the speed and the orientation of the conveyor system, these two information are integrated at one model identified as Status.

The main events used to model these plants are related at the next table, with the discrimination of which ones are controllable (C) or uncontrollable (U) events, their labels are showed too.

Label	Event	Type
1	Push start	C
2	System started	U
11	Turn On – Conveyor 1	C
13	Turn Off – Conveyor 1	C
21	Camera 1: recognize model and position	C
22	Model 1 – position OK	U
24	Model 2 – position OK	U
32	Model 1 – position NOK	U
34	Model 2 – position NOK	U
23	Camera 2: check weld spots Model 1	C
26	Model 1 – weld spots OK	U
36	Model 1 – weld spots NOK	U
25	Camera 2: check weld spots Model 2	C
28	Model 2 – weld spots OK	U
38	Model 2 – weld spots NOK	U
49	Stop Robot	C
41	Robot 1: welding Model 1	C
42	Robot 1: end welding Model 1	U
40	Robot Error	U
43	Robot 1: welding Model 2	C
44	Robot 1: end welding Model 2	U
45	Robot 2: re-welding Model 1	C
46	Robot 2: end re-welding Model 1	U
47	Robot 2: re-welding Model 2	C
48	Robot 2: end re-welding Model 2	U
51	Open clamps – Model 1	C
52	Clamps opened – Model 1	U
53	Open clamps – Model 2	C
54	Clamps opened – Model 2	U
55	Close clamps – Model 1	C
56	Clamps closed – Model 1	U
57	Close clamps - Model 2	C
58	Clamps closed – Model 2	U
50	Clamps Error	U
62	Status Conveyor – OK	U

64	Status Conveyor – NOK	U
72	RFID position – OK	U
74	RFID position – NOK	U
91	Turn on – Alarm	C
93	Turn off – Alarm	C

The automaton generated by the program, TCT® - Wonham (http://www.control.utoronto.ca/cgi-bin/dlxptct.cgi) for each asynchronous plant are showed below.

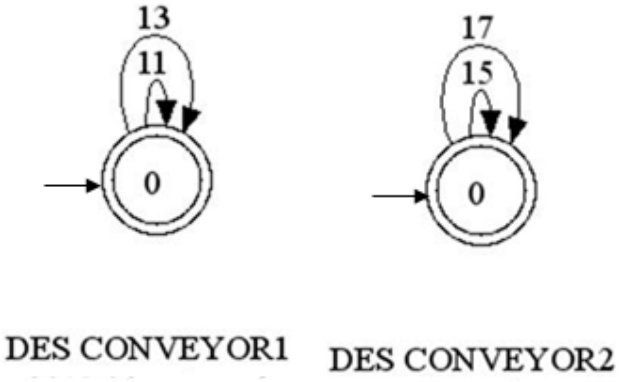


Figure 7: Automata of the Conveyor System.

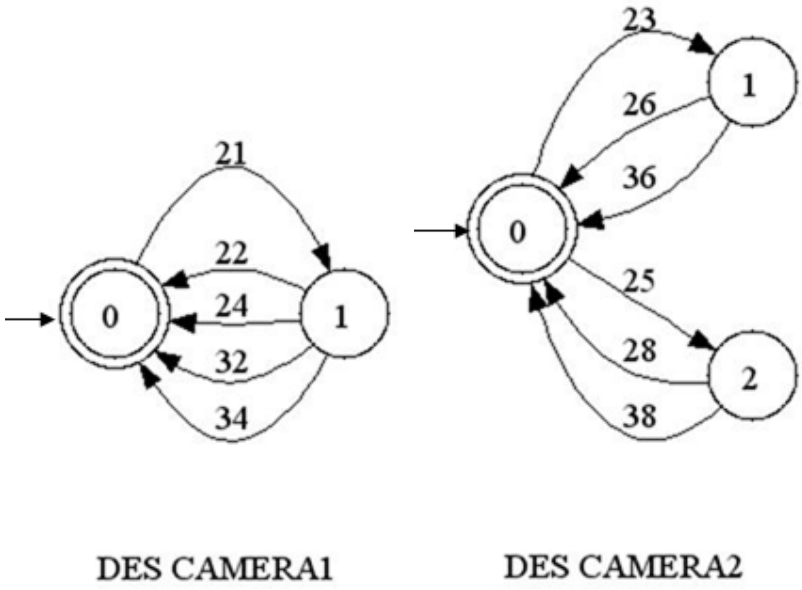


Figure 8: Automata of the Camera.

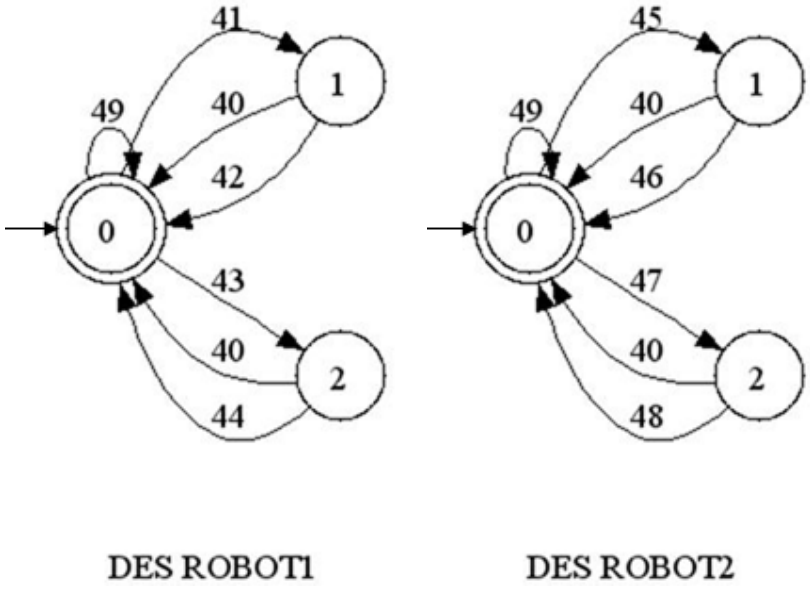


Figure 9: Automata of the Robot.

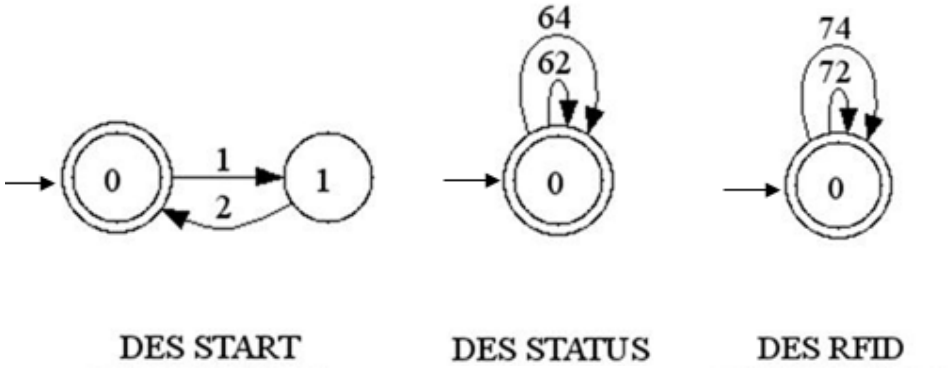
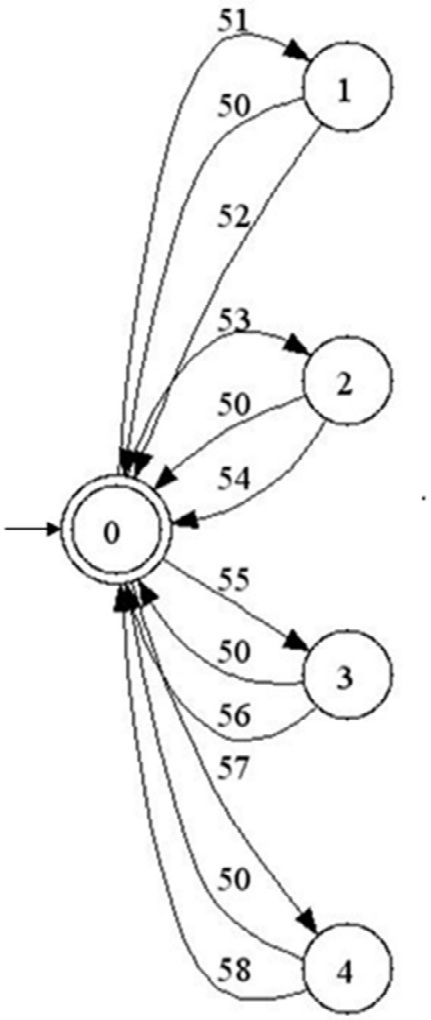


Figure 10: Automata of the Start, Status sensors and RFID sensor.



DES GRAMPOS

Figure 11: Automaton of the clamps.

5.3 Specifications Models

To determine the behaviour of the plant and its restrictions, specifications are designed and modelled. For the final demo scenario 13 specifications were designed and are better explained next.

Specification 1: The conveyor 1 only can be turn on if the RFID doesn't detect the right position, the conveyor Status is ok and the Start button is activated. If the conveyor Status is not OK, the conveyor 1 is stopped and an alarm is raised. When the RFID sensor reaches the right position and the conveyor Status is ok, the conveyor 1 is turn off. The figure 12 shows the automata of the specification 1. The local plant for this specification is composed by asynchronous composition of automata Conveyor 1, Start, Status, RFID and Alarm.

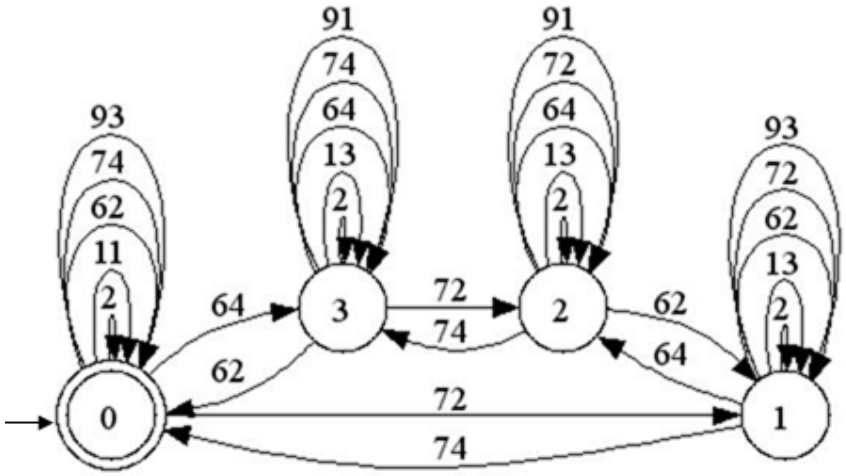


Figure 12: Automaton of the specification 1.

Specification 2: The Camera 1 only starts work when the conveyor 1 is stopped. The local plant of this specification is composed by the asynchronous plant Camera 1 and Conveyor 1.

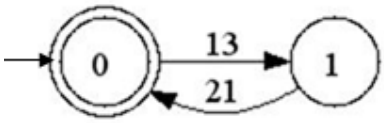


Figure 13: Automaton of the specification 2.

Specification 3: If the camera 1 results that the piece is at the right position and it is of the model 1, the clamps related to this model are allowed to be closed. If the camera 1 returns that piece of the model 1 is not at the right position, the clamps can't be closed. The asynchronous composing of the Camera 1 and Clamps automata is the local plant for this specification.

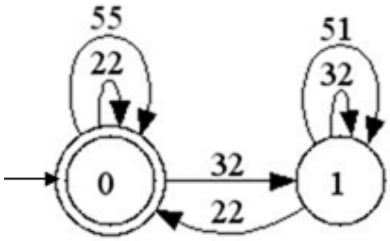
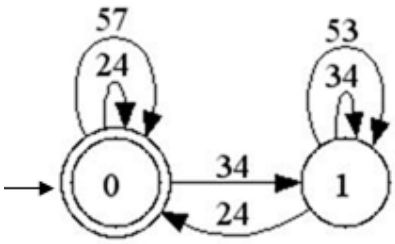


Figure 14: Automaton of the specification 3.

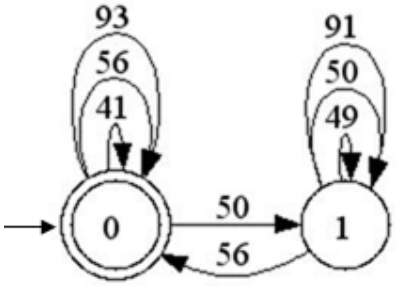
Specification 4: This specification has the same goal of the specification 3, but related to the model 2 of the piece. The local plant is the same too.



DES SPF4

Figure 15: Automaton of the specification 4.

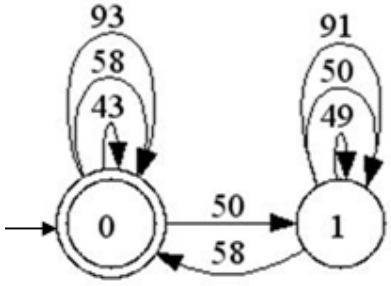
Specification 5: When the clamps returns that they are properly closed for the model 1, the robot can start weld. But if the clamps return an error, the robot cannot start work and the alarm is set up. The local plant is composed by the automata of Robot 1, Clamps and Alarm.



DES SPF5

Figure 16: Automaton of the specification 5.

Specification 6: This specification is the same of the specification 5, but related to the model 2.



DES SPF6

Figure 17: Automaton of the specification 6.

Specification 7: If the robot finishes the weld or the re-weld process for the model 1, the camera is able to start the recognition of the weld points. However, if the robot returns an error message the alarm is turn on and the camera stops work. The local plant is composed by the asynchronous plants of the Camera 2, Robot 1, Robot 2 and Alarm.

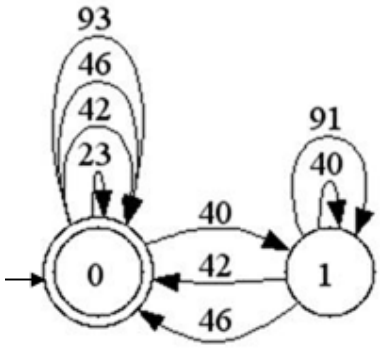


Figure 18: Automaton of the specification 7.

Specification 8: is the same logic of the specification 7, but related to the model 2.

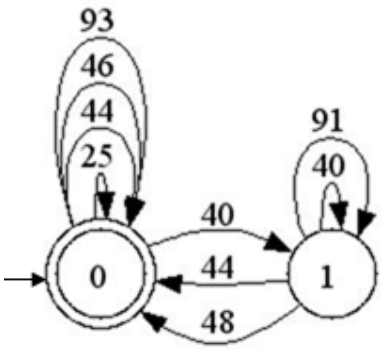
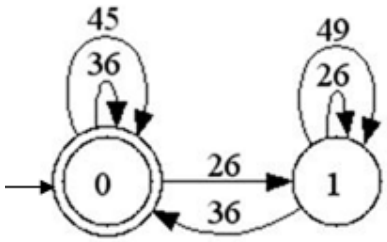


Figure 19: Automaton of the specification 8.

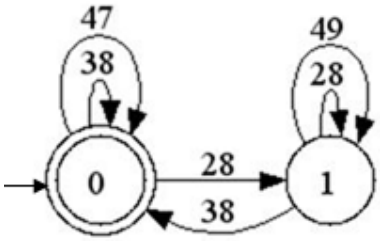
Specification 9: If the camera doesn't recognize the weld points of the model 1, the robot needs redo them. The local plant is composed by the Robot 2 and Camera 2 automata.



DES SPF9

Figure 20: Automaton of the specification 9.

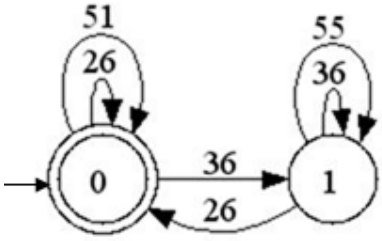
Specification 10: The same logic of the specification 9, but for the model 2.



DES SPF10

Figure 21: Automaton of the specification 10.

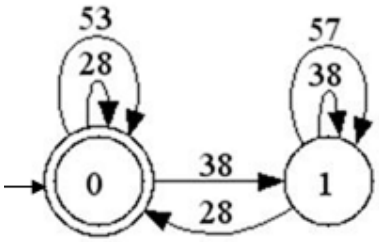
Specification 11: The clamps related to the model 1 are able to be open when the camera returns that the weld points are ok. The automata of the clamps and camera 2 make up the local plant of this specification.



DES SPF11

Figure 22: Automaton of the specification 11.

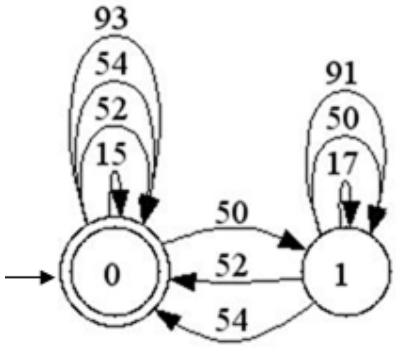
Specification 12: Even the specification 11, but for the model 2.



DES SPF12

Figure 23: Automaton of the specification 12.

Specification 13: If the clamps are properly open no matter the model of the piece, the conveyor 2 is able to turn on, but if an error is detected from the clamps the conveyor 2 cannot start or is stopped and an alarm is raised. The local plant is composed by the automata Conveyor 2, Clamps and Alarm.



DES SPF13

Figure 24: Automaton of the specification 13.

5.4 Supervisory Controllers

The asynchronous composition between the local plants and the specifications results in a set of languages, the supreme language of this set is the supervisory control resultant of the design. At the program used to develop the control language there is a function that results at the supervisory controller and also it is possible reduce it for the minimal automaton version.

Local modular controller 1: $RSUP_1 = Conveyor_1 \parallel Start \parallel Status \parallel RFID \parallel Alarm \parallel SPF1$

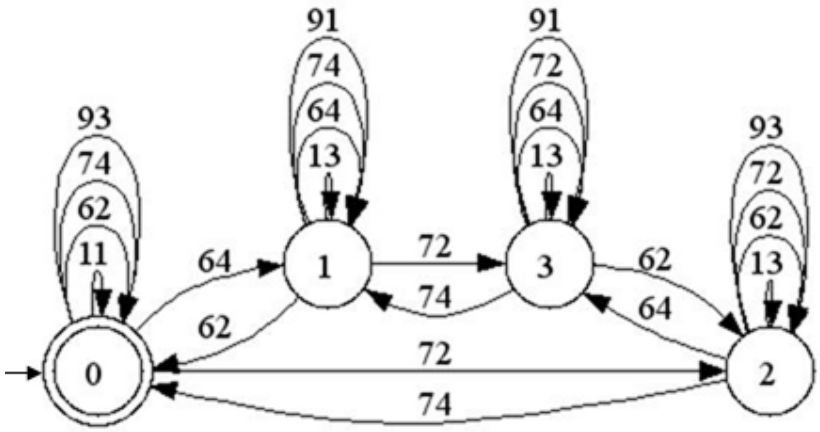


Figure 25: Automaton of the local modular controller 1.

Local modular controller 2: $RSUP_2 = Camera_1 \parallel Conveyor_1 \parallel SPF_2$

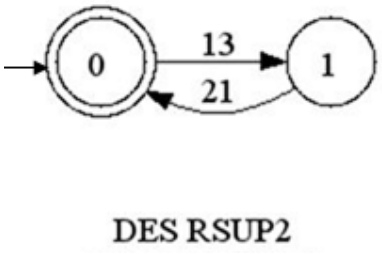


Figure 26: Automaton of the local modular controller 2.

Local modular controller 3: $RSUP_3 = Camera_1 \parallel Grampos \parallel SPF_3 \parallel SPF_4$

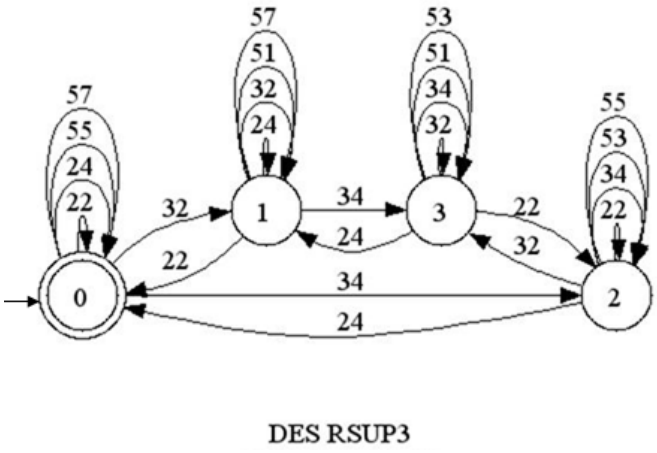


Figure 27: Automaton of the local modular controller 3.

Local modular controller 4: $RSUP_4 = Robot_1 \parallel Grampos \parallel Alarm \parallel SPF_5 \parallel SPF_6$

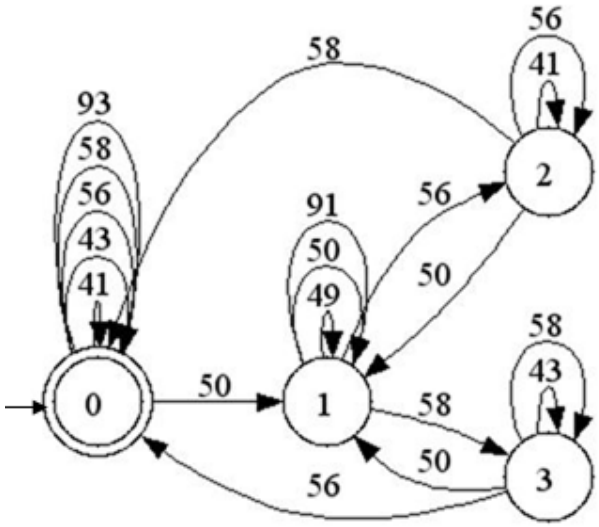


Figure 28: Automaton of the local modular controller 4

Local modular controller 5: $RSUP_5 = Robot_1 \parallel Camera_2 \parallel Alarm \parallel SPF_7 \parallel SPF_8$

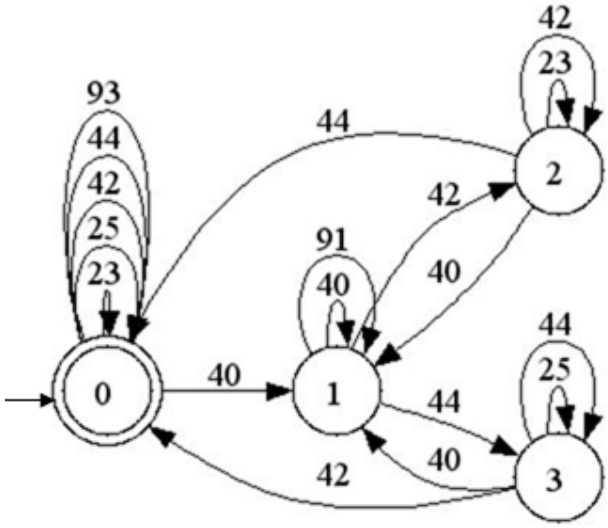


Figure 29: Automaton of the modular controller 5.

Local modular controller 6: $RSUP_6 = Robot_2 \parallel Camera_2 \parallel SPF_9 \parallel SPF_{10}$

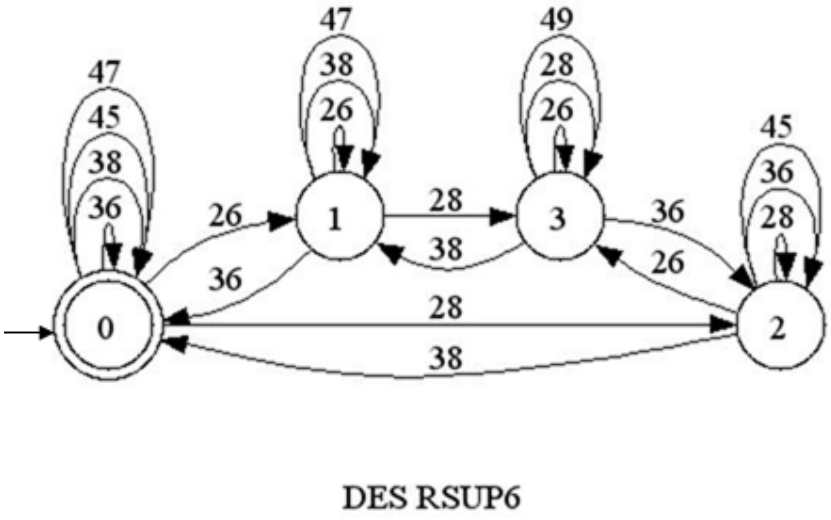


Figure 30: Automaton of the local modular controller 6.

Local modular controller 7: $RSUP_7 = Grampos \parallel Camera_2 \parallel SPF_{11} \parallel SPF_{12}$

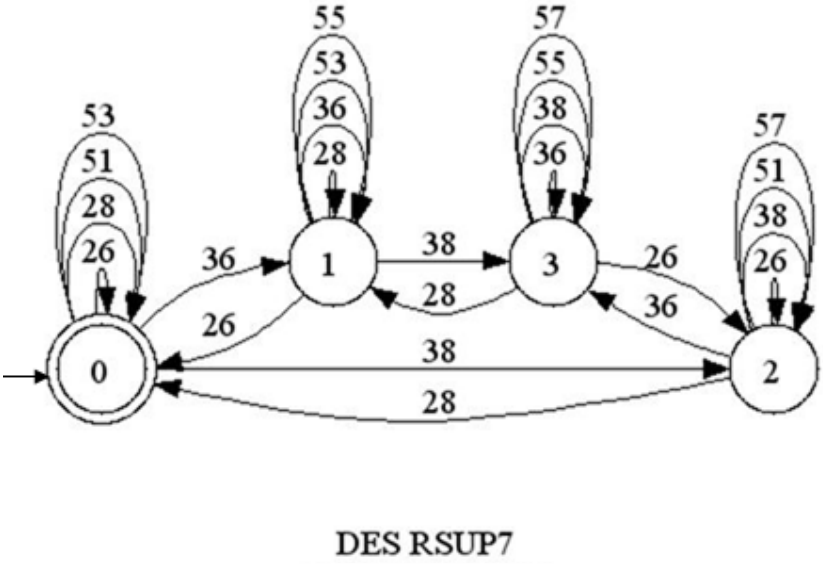


Figure 31: Automaton of the local modular controller 7.

Local modular controller 8: $RSUP_8 = Conveyor_2 \parallel Grampos \parallel Alarm \parallel SPF_{13}$

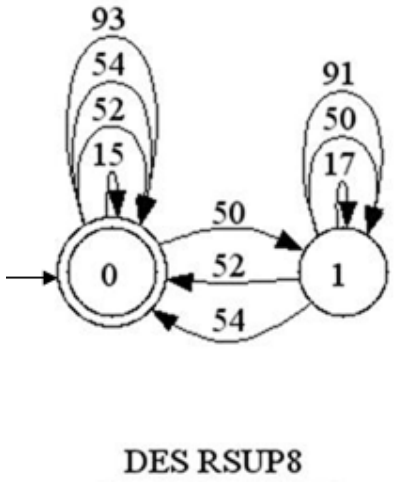


Figure 32: Automaton of the local modular controller 8.

5.5 Control Architecture

The architecture helps the implementation of the supervisors into the PLC. The architecture is composed by four levels: Modular supervisors, Product System, Operating sequences and Real system. The figure 33 shows the structure of the control architecture. This structure is the same independent of the supervisors' number or devices in the plant.

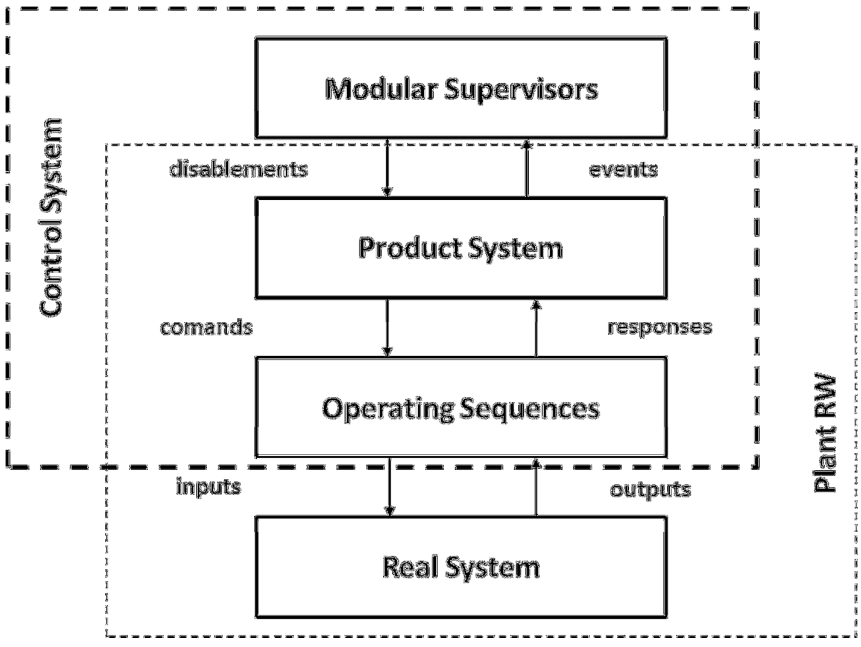


Figure 33: Control Architecture.

At the level of Modular Supervisors, the supervisors obtained at the section before are implemented at the PLC and active states are updated with the signals that come from the low level Product System. This level associates to the active states a set of disabling signals of the controllable events that control the Product system level.

The devices' models are implemented at the Product system level, where also the controllable events are executed when not disabled at the superior level, Modular Supervisors. The uncontrollable events are related to the behavior of the Operating Sequences level.

At the Operating Sequences level is made the interface between the control solution and the physical plant. This level receives the commands from the Product System level, gives signals to the physical plant inputs and watches the physical plant outputs signals, transposing them to the Product system level as answers.

The physical plant is represented by the Real System level at the architecture.

6. References

D5.1.1 Initial infrastructure for distributed control logic

[Wonham, 2013] Supervisory Control of Discrete-Event Systems

[Pena P. N.,Cury J. E. R., 2009] Verification of Nonconflict of Supervisors Using Abstractions

[Queiroz, M. H., Cury J. E. R.] Controle Supervisório Modular de Sistemas de Manufatura

[Queiroz, M. H., Santos E. A. P., Cury J. E.] Síntese Modular do Controle Supervisório em Diagrama Escada Para Uma Célula de Manufatura