A General Method for the Analysis and the Logical Generation of Discrete Mathematical Systems in Programmable Logical Controller

Daniel M. Dubois *

Department of Applied Informatics and Artificial Intelligence, HEC Management School, N1, University of Liège, rue Louvrex 14, 4000 Liège, Belgium

daniel.dubois@ulg.ac.be

Antonio Mascia

EURO VIEW SERVICES SA, Chaussée de Lodelinsart 273, 6060 Gilly, Belgium

management@euro-view.com

Abstract

This paper deals with a general method for the analysis and the logical generation of discrete systems in Programmable Logical Controller (PLC). The Boolean operators are implemented with a generic and unique algebraic model as event-dependent discrete equations, which can be executed in a sequential order. With this method, a generator of sequential logical tables can be designed, simulated and executed for implementing discrete dynamical systems. The purpose of this research is to design self-rewrite operating systems in PLC, which automatically checks the logic of the implemented discrete dynamical systems, for simulation and execution of sequential operations.

1 Introduction to Industrial Automation

This paper deals with industrial automation in relation with CAST [Pichler and Schwärtzel, 1992], and is the continuation of our work on the prototype GENSYSPRO [Dubois and Mascia, 1995a], [Dubois and Mascia, 1995b], and [Mascia and Dubois, 1995].

Several problems exist in the industrial automation with the Programmable Logical Controller (PLC):
1. In the last 30 years, the global cost of the automation did not decrease.
2. In the world, a completely safe 100% program does not exist, and one can only program all combinatory possibilities.
3. Every manufacturer creates a different system.
4. Every manufacturer creates a different syntax program.
5. Every manufacturer creates a different programming tool.
6. Every manufacturer needs a different training.
7. The simulation of the process needs a specific program.
8. Idem for all supervisors (graphic tool) for the control of the process.

In any process of an industrial automation, from analysis to the project exploitation, there are 7 steps:
1- CONCEPTUAL ANALYSIS: by example, a factory needs a water supply installation.
2- FUNCTIONAL ANALYSIS: by example, two pumps (input / output), start / stop, high and low level control and alarms, security systems.
3- SIMULATION THE PROCESS: this operation needs a specific program.
4- PROGRAMMING THE SYSTEM: each system constructor uses different syntaxes and instructions.
5- START UP AND VALIDATION: the correction of errors must be performed.
6- CONTROL & SUPERVISION: this step needs another tool (graphic software) and the programming task.
7- TRAINING: each system and application needs a specific training.

Each step is a translation with errors possibilities.

For the simulation, a double program is necessary.

For enhancing the procedure, the objectives consist to:
1.Reduce the global cost of automation.
2.Reduce the quantity of steps, time and tools.
3.Create completely safe programs and systems.
4.Create one unique graphic tool for the description, simulation, and execution and control the industrial process automation.
5.Increase the capabilities of the systems: execution speed.
6.Define a unique methodology (universality of program and training).

In this paper; we will present the model and method, realized in the prototype GENSYSPRO.

* CHAOS : Centre for Hyperincursion and Anticipation in Ordered Systems, Institute of Mathematics, Grande Traverse 12, 4000 Liège 1, Belgium http://www.ulg.ac.be/mathgen/CHAOS

CYBERNETICS AND SYSTEMS 2010, ROBERT TRAPPL, editor, Published by Austrian Society for Cybernetic Studies, Vienna, Austria, ISBN 978-3-85206-178-8

2 The Software GENSYSPRO

GENSYSPRO is a prototype software based on a new concept which deals with the following tools:
1- CONCEPTUAL ANALYSIS
2- FUNCTIONAL ANALYSIS
3- SIMULATION
4- PROGRAMMING.
5- START UP AND VALIDATION
6- CONTROL & SUPERVISION
7- TRAINING

All the industrial automation procedures are developed in one generic and universal tool and method. GENSYSPRO is the Solution of the problems presented in the introduction, because it integrates several tools in only one.

This allows the analysis of automatism problem:
A - Structured organization of the project (Object Oriented).
B - Graphical representation of the process.
C - Sequential description of the process.
D - Instantaneous simulation.
E - Universal method and unique training.

GENSYSPRO brings a method, because all the programmers will follow the same rules :
1 - Describe only expected actions.
2 - Describe sequentially all physical events that happened in the temporal process evolution.

Let us explain this, with a practical project described hereafter.

3 Process: Water Supply for Factory

This section describes the project for the industrial automation of the process of a water supply for a factory.

3.1 Starting the project

The analysis and graphical description of the process of the water supply for a factory is given in Figure 1.

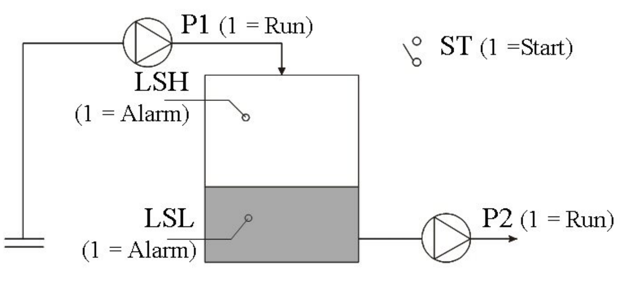

Figure 1: Analysis and graphics of the water supply.

The abbreviations are given as follows:
- **ST:** Start/stop switch water supply
- **P1:** Pump 1 (input tank)
- **P2:** Pump 2 (output tank)
- **LSL:** Level Switch Low
- **LSH:** Level Switch High

Let us now give the Boolean presentation of this process.

3.2 Boolean presentation

Let us describe the different steps of system evolution on the graphics:
- The description is done with graphic tools
- The program translates this description into Boolean data matrix

The Table 1, gives the complete process description.

TABLE 1: The four event steps of the water supply.

Table num	0001	Steps names	**Inputs**			**Outputs**	
Table name	**START**	**Levels:**	**ST**	**LSL**	**LSH**	**P1**	**P2**
Steps	001	Low	1	1	0	1	0
	002	Normal	1	0	0	1	1
	003	High	1	0	1	0	1
	004	Normal	1	0	0	0	1

The following 4 figures 2 to 5, give the graphical design, with their variables and steps.

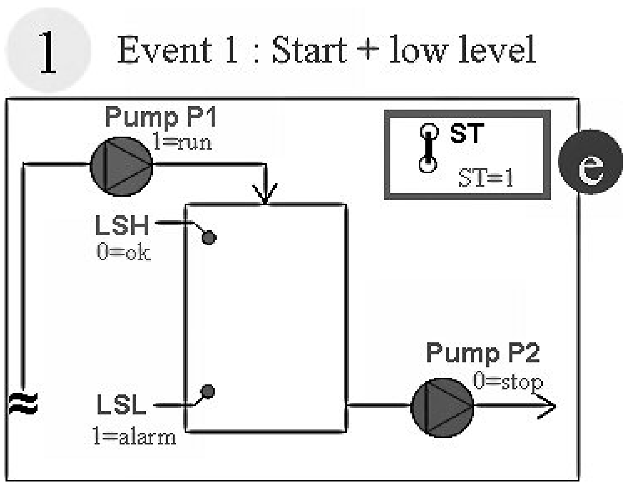

Figure 2: EVENT 1. The “e” represents the “event”.

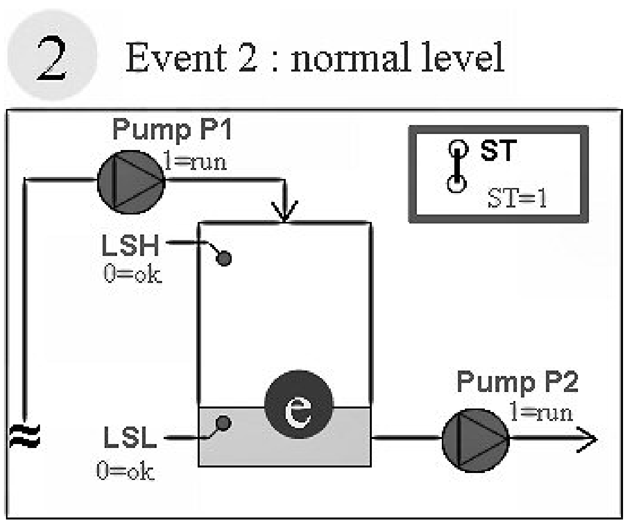

Figure 3: EVENT 2. The “e” represents the “event”.

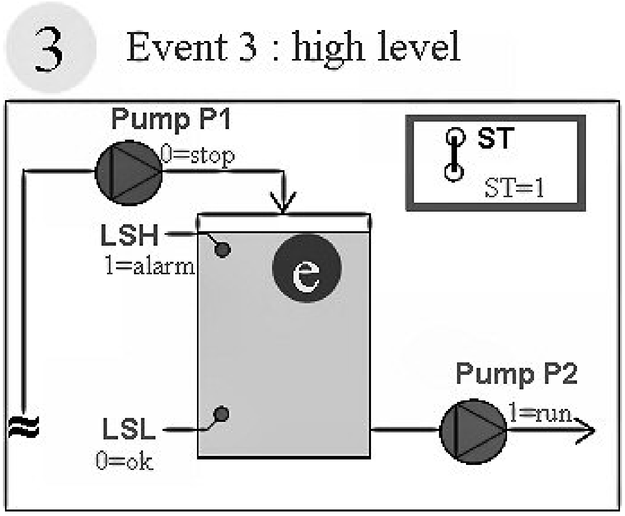

Figure 4: EVENT 3. The “e” represents the “event”.

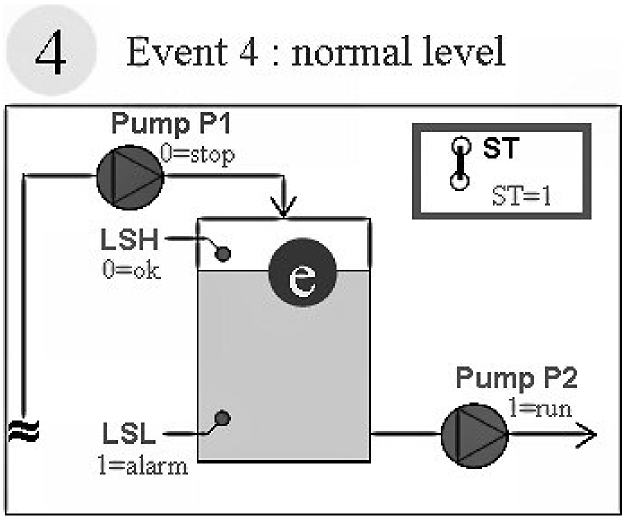

Figure 5: EVENT 4. The “e” represents the “event”.

3.3 Memory of event steps

The validation of the evolution is given by the simulation mode.

The program creates the whole table before the model generation

- The future occurrences, at event step t+1, are taken into account for anticipation.
- The past occurrences, at event steps t–1, t–2, ... are taken into account to suppress incoherencies.

The following Tables 2A, 2B and 2C give the Boolean tables of the successive event steps t–1, t, t+1.

Important remark:
The step temporal index “t–1, t, t+1” represents the numbering of steps of the events, and does not represent the time interval between two successive events. The index t becomes t+1, at each new event

TABLE 2A: Boolean table at past event step t–1

	STOP	**Inputs**			**Outputs**	
	Without water	**ST**	**LSL**	**LSH**	**P1**	**P2**
	Low level	0	1	0	0	0

TABLE 2B: Boolean table at current event step t

	START	**Inputs**			**Outputs**	
Steps		**ST**	**LSL**	**LSH**	**P1**	**P2**
001	Low level	1	1	0	1	0
002	Normal level	1	0	0	1	1
003	High level	1	0	1	0	1
004	Normal level	1	0	0	0	1

TABLE 2C: Boolean table at future event step t+1

	START	**Inputs**			**Outputs**	
Steps	Without water	**ST**	**LSL**	**LSH**	**P1**	**P2**
001	Low level	1	1	0	1	0

3.4 Algebraic Model

The generating formulas deal only on the execution step action.

The models are created following these rules:

- only consider lines whose outputs are at 1
- inputs are linked by the arithmetic multiplication (.)
  if input = 1 then variable tag name
  if input = 0 then (1 – variable) tag name
- each considered line by output variables are linked by the arithmetic addition (+)

Here are the model discrete equations for the water supply for each output P1 & P2:

$$P1 = ST.LSL.(1 - LSH).(1 - LSL_{t-1}).(1 - LSH_{t-1})$$
$$+ ST.(1 - LSL).(1 - LSH).LSL_{t-1}.(1 - LSH_{t-1})$$
$$P2 = ST.(1 - LSL).(1 - LSH).LSL_{t-1}.(1 - LSH_{t-1})$$
$$+ ST.(1 - LSL).LSH.(1 - LSL_{t-1}).(1 - LSH_{t-1})$$
$$+ ST.(1 - LSL).(1 - LSH).(1 - LSL_{t-1}).LSH_{t-1}$$

GENSYSPRO shows three functions in one step: the application design (graphic), with the simulator system, and with the supervision and control system.
The result goes to the PLC and executes the algebraic models.

4 The Software GENSYSPRO

With GENSYSPRO, the global concept needs:

- a new graphical tool,
- a new operating system,
- a new conceptual construction of the systems.

A graphical tool needs a complete object oriented database with an automatic variables generator.

The global principle of the concept is based on the event management.

The design tool, simulator tool and execution tool are the same model application: "all in one"

The DataBase overview is given in Figure 6.

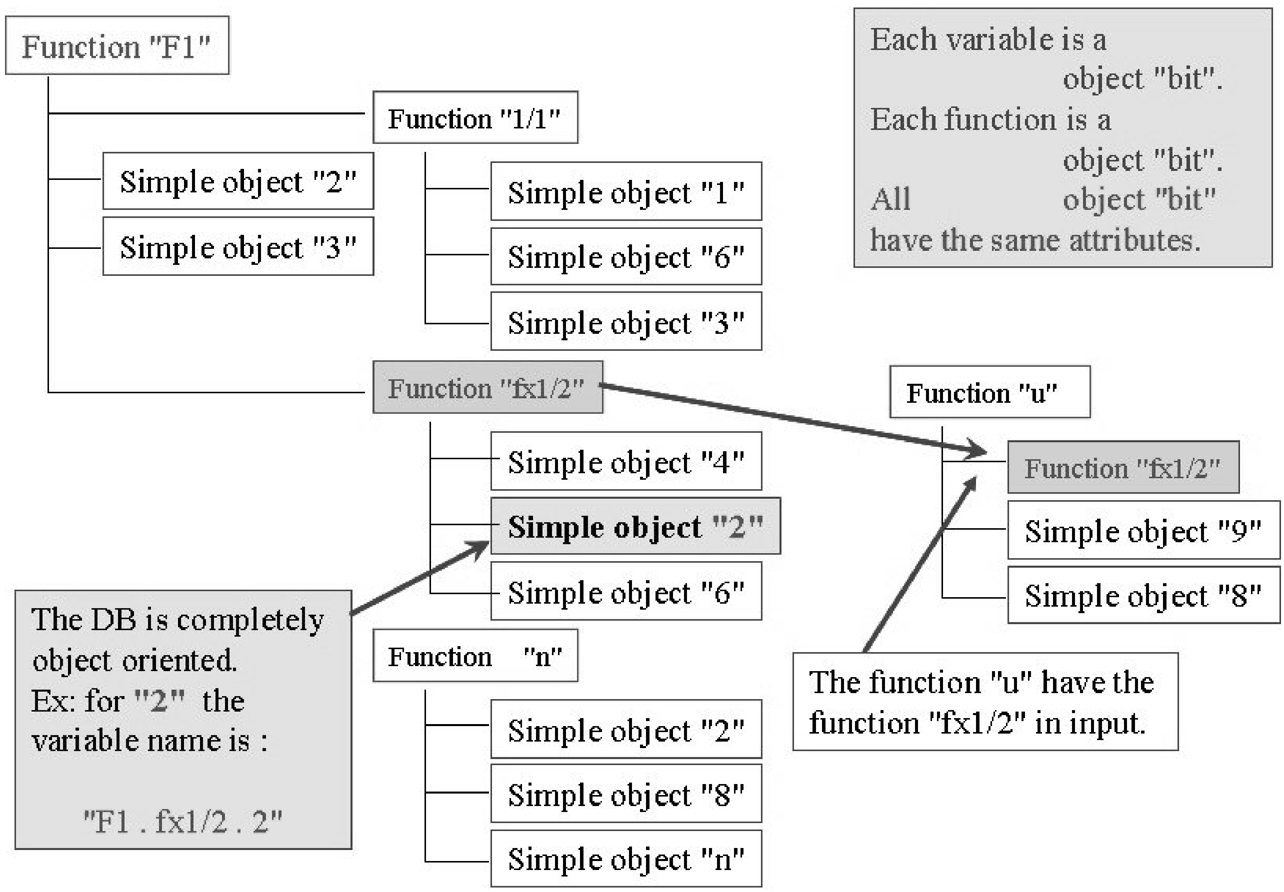

Figure 6: The DataBase (DB) overview.

The functional commands events in all the systems are given by :
- an initial step;
- many functional choices (direction) activated or not activated (example, on/off, auto/hand, local/remote, etc.)

With "XOR" between a current functional choice and a change of this, the system will be able to activate or not activate actions and optimise the time of exploitation. The system works with active functions on event. In our example, there are two functions: start and stop, as shown in Figure 7.

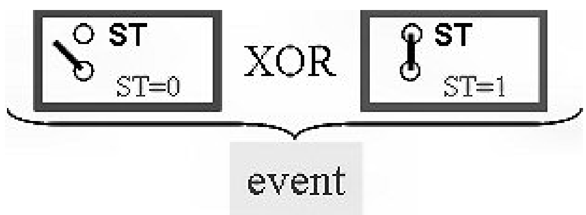

Figure 7: Start and Stop

The following Tables 3A-B-C show the intelligent inputs on events.

TABLE 3A: No event, the system does not work.

Initial Step 0						
Inputs (t-1)		**Inputs (t)**		=	**Event**	**System**
?	XOR	ST=0		**0**	**= 0**	**NO ACTION** **If no event, the system don't work**
?	XOR	LSL=1		**0**		
?	XOR	LSH=0		**0**		
System in standby : waiting unknown time on Step 0 (event expected)						

TABLE 3B: In fact, the system works only on event.

Step 0 to Step 1 = Start in low level						
Inputs (t-1)		**Inputs (t)**		=	**Event**	**System**
ST=0	XOR	ST=1		**1**	**= 1**	**ACTION** **Execute outputs & Go to Step 1**
LSL=1	XOR	LSL=1		**0**		
LSH=0	XOR	LSH=0		**0**		
System in standby : waiting unknown time on Step 1 (event expected)						

TABLE 3C: continuation of Figure 7B.

Step 1 to Step 2 = Normal level						
Inputs (t-1)		**Inputs (t)**		=	**Event**	**System**
ST=1	XOR	ST=1		**0**	**= 1**	**ACTION** **Execute outputs & Go to Step 2**
LSL=1	XOR	LSL=0		**1**		
LSH=0	XOR	LSH=0		**0**		
System in standby : waiting unknown time on Step 2 (event expected)						

5 Towards a new industrial operating system

This work permitted to outline some properties in industrial automation for developing a new industrial operating system (IOS), which gives rise to a semantic information about the process.

The main point is that such a CAST, Computer Aided Systems Theory for the Design of Intelligent Machines, presented in this paper, would open new avenues where programmation and artificial languages would disappear in profit of the Human Natural Language. An overview of such a new Operating System, is given in Figure 8, applied to the water supply.

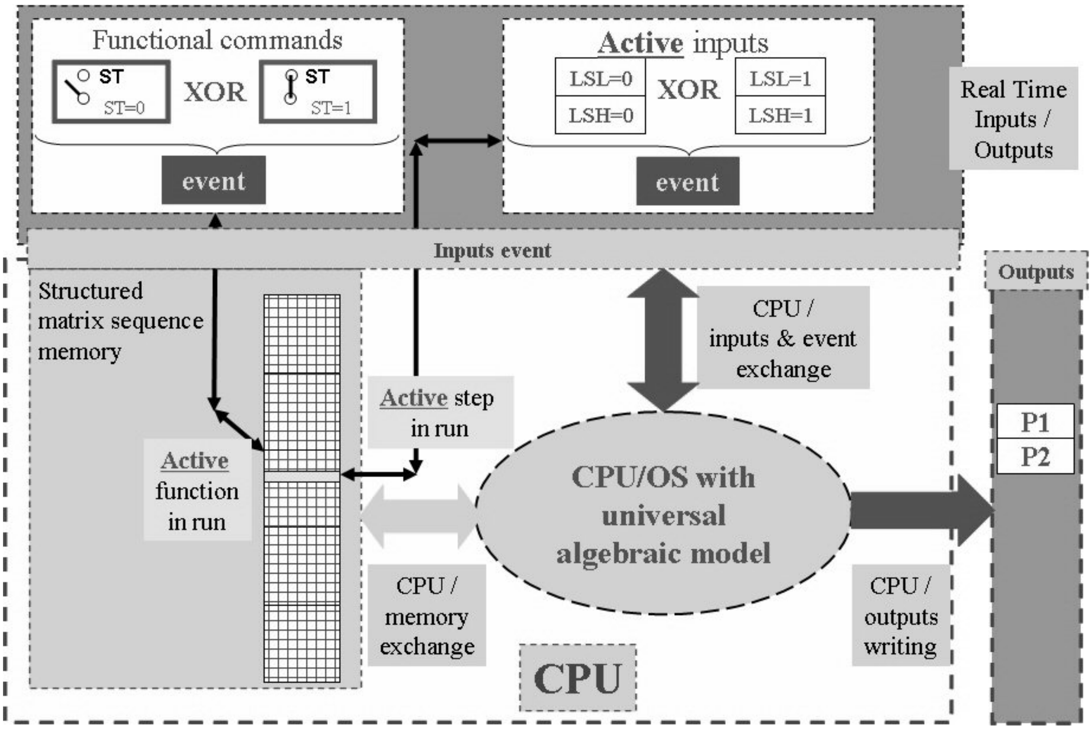

Figure 8: Overview of the new Industrial Operating System

All the actual computing systems work on a permanent cyclic recursive basis, without necessarily execute a function. Nevertheless, all the computing systems execute functional orders given by man and execute an output value only if the equation is true based on an event (evolution) of an input or on a parameter that has changed.

GENSYSPRO does execute nothing if there is no change of functional order (controlled by an XOR), and if there is no event or change of parameters (XOR on the inputs).

What is the breakthrough with GENSYSPRO, is the fact that it is at rest when no event happens, contrary to all the other industrial computing systems, which work all the time, based on an internal clock.

GENSYSPRO is thus an event-based software with a general method for the analysis and the logical generation of discrete systems in Programmable Logical Controller. The Boolean operators are implemented with a generic and unique algebraic model as event-dependent discrete equations, which can be executed in a sequential order. So, a generator of sequential logical tables can be designed, simulated and executed for implementing discrete dynamical systems.

A new type of operating systems can be designed, which automatically checks the logic of the implemented discrete dynamical systems, for simulation and execution of sequential operations. Due to lack of space in this paper, it is not possible to develop extensively all the novelty of this operating system.

In a few words, we can say that this approach deals with artificial intelligence, neural networks and, recently, with multi-agent systems [Wooldridge, 2009].

So, from the methodology presented at the beginning of the paper, the algebraic method to build the discrete equations of the Boolean tables depending on event steps, permits to create automatically a neural systems with McCulloch and Pitts [1943] formal neurons. Indeed, one of us [Dubois, 1999] showed that non-linear digital equations can be easily built from Boolean Tables. These equations are Heaviside Fixed Functions that can be used to generate directly neural networks with McCulloch and Pitts formal neurons. Classically all Boolean Tables are built with NOT, AND and OR operators, but an universal operator can be used, the so-called NAND, NOT AND (or the NOR, NOT OR). In reference to Flip-flop one-bit memory based on two NAND in computer science, models of neural memory were built. A neural memory can be designed with only one anticipatory McCulloch and Pitts neuron. This anticipatory neuron is similar to a Heaviside Cusp function. A neural network can be viewed at different time scales. At one hand, sequences of continuous time durations during which neurons compute, and, at the other hand, successive discrete times dealing with the neural logics.

These models of neural memory are completely different from the classical synaptic memory concept. With a set of fixed synaptic weights, an anticipatory neural network can perform memorization, retrieval and recognition of information in a dynamical way.

6 Conclusion

The mathematical model is really a description of the dynamics of the process. Indeed, the model automatically permits the validation and the simulation of the process design without programming.

The performances of the GENSYSPRO computing system for industrial automation are due to the fact that there is only an execution if there is a change detected by the XOR, either in the functional order (human decisions), either event based order.

Research is in progress to develop such an operating system based on natural language.

Acknowledgments

Daniel M. Dubois would like to thank the Fonds National de la Recherche Scientifique de Belgique, for the financial support to participate to this meeting.

References

[Dubois, 1999] Dubois Daniel M. (1999) Hyperincursive McCulloch and Pitts Neurons for Designing a Computing Flip-Flop Memory. Computing Anticipatory Systems: CASYS'98 - Second International Conference. Edited by Daniel M. Dubois, Published by The American Institute of Physics, AIP Conference Proceedings 465, pp. 3-21.

[Dubois and Mascia, 1995a] Dubois D. M., A. Mascia, *Méthode Générale pour l'Analyse et la Programmation des Systemes Discrets*, Proceedings of the 14th International Congress on Cybernetics, edited by the International Association for Cybernetics, pp.571-576, 1995.

[Dubois and Mascia, 1995b] Dubois D. M. and A. Mascia, *Computer Aided Generator of Models for Designing Automation Devices*, in Advances in Computer Cybernetics, volume III, edited by G. E. Lasker, published by The International Institute for Advanced Studies in Systems Research and Cybernetics, University of Windsor, Canada, pp. 23-27, 1995.

[Mascia and Dubois, 1995] Mascia A., Dubois D. M., *Générateur de Modèles Algébriques Appliqué à la Conception d'un Automatisme Industriel*, Proceedings of the 14th International Congress on Cybernetics, edited by the International Association for Cybernetics, pp.577-582, 1995.

[McCulloch and Pitts, 1943] McCulloch W. S., Pitts W. A logical calculus of the ideas immanent in nervous activity, *Bulletin of mathematical Biophysics*, vol 5, 1943, pp. 115-133.

[Pichler and Schwärtzel, 1992] F. Pichler, H. Schwärtzel (Eds.), CAST Methods in Modelling, Computer Aided Systems Theory for the Design of Intelligent Machines, Springer-Verlag, Berlin, Heidelberg, 1992.

[Wooldridge, 2009] Michael Wooldridge. An Introduction to MultiAgent Systems, John Wiley & Sons, 2009, 2nd Edition