6

Flexible Manufacturing System Simulation Using Petri Nets

Carlos Mireles, Alfonso Noriega and Gerardo Leyva

1. Introduction to Petri Nets.

In 1962 German mathematician Kart Adam on his Ph.D. work "Kommunikation mit automaten", he proposed a new way for modelling & representating systems where events and transitions are present. This new representation is now known as Petri Nets (PN).

Since then, several researchers have used this tool. In the USA and in Europe several developments have been done. In 1980, in Europe, a work table was controlled using PN and the related work was presented at one International Conference. Researchers in France have done great contributions to the development and the applications of PN. Some of them used the PN to describe Programmable Logic Controllers (PLC) and this application had a big influence in the development of the Grafcet.

2. Petri Nets Theory

2.1 Basic Concepts

2.1.1 Petri Net

Graphic and executable technique to specify and analize dynamic & concurrent discrete event systems.

2.1.2 Formal

Petri Nets analysis is a mathematical technique well defined. Many static and dynamic properties of a PN (and therefore for a system represented by a PN) can be mathematically proved.

Source: Manufacturing the Future, Concepts - Technologies - Visions , ISBN 3-86611-198-3, pp. 908, ARS/plV, Germany, July 2006, Edited by: Kordic, V.; Lazinica, A. & Merdan, M.

2.1.3 Graphics

This technique belongs to the area of mathematics known as "Graph Theory", which makes a PN being able to represented both by a graphic and by mathematic expressions. This graphic property provides a better understanding of the system which is represented by the PN.

2.1.4 Executable

A PN can be executed and then the dynamics of the system can be observed.

2.1.5 Concurrency

Multiple independent entities can be represented and supported by PN.

2.1.6 Discrete event dynamic systems

A system that can change its current state based in both its current state and the transition conditions between states.

2.2 Structure of a Petri Net

2.2.1 Formal Definition

A Petri Net has a set of Places, a set of Transitions, an Input Function and an Output Function.
The structure of a Petri Net is the array (L, T, E, S, m_0) where:

- L is the set of places in the graph.
- T is the set of transitions in the graph. L & T satisfy the following conditions $L \cup T = \varnothing$ $L \cap T = \varnothing$.
- $E : LxT \to \{0,1\}$ is the input function which specify the connectivity between Places & Transitions.
- $S : LxT \to \{0,1\}$ is the output function which specify the connectivity between Transitions & Places.

2.2.2 Graphic Representation

A Petri Net is an oriented graph that contains two types of nodes: Places and Transitions which are connected by oriented arcs that connect Places with Transitions –connectivity between nodes of the same kind is not allowed. In the graphic representation, Places are shown as circles, Transitions are shown as bars and the arcs are shown as arrows. Places represent conditions and Transitions represent events. A Transition has certain number of places either as inputs or outputs which are pre and post conditions. See figure 1.

2.2.3 Marking a Petri Net

The marking of a Petri Net is a positive integer μ_i for every place Li. A mark is represented as a dot within the circle for a given place. These marks move between the places which provide the dynamic feature of the Petri Net.

A Petri Net is considered marked when at least one place has a mark.One place can have N marks, where N is a positive integer. If N = 0 then the place has no marks. A marker M is a function $\mathrm{M}:\mathrm{L}\to\mathrm{N}$ that can be expressed by:

$$\mathrm{M}=\begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_N \end{bmatrix} \tag{1}$$

m_i is the number of dots for place L_i.

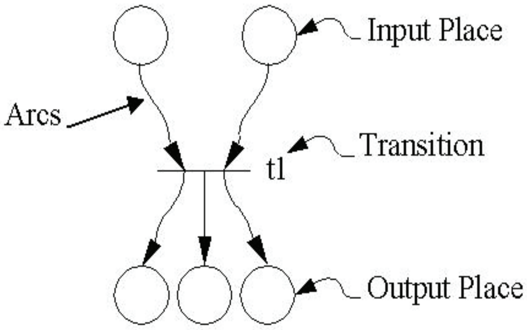


Figure 1. Graphic representation of a PN showing the Input & Outputs places as well as the Transition.

2.2.4 Interpretation of a Petri Net

Marks are the resources. Resources can be physical entites or no physical such as messages or information.

Places are the locations where the resources are stored.

Transitions are actions that transfer the resources to generate new resources.

The weight of each arrow (a number on each arrow) is the minimum number of resources needed in a place in order to get the action indicated by the transition.

2.2.5 Triggering a Transition in a Petri Net

To trigger a Transition, it has to be validated which means that every Place connected to this Transition has to have at least one mark. The execution of a Petri Net is controlled by the number and distribution of the marks on their Places.

When a Transition is triggered, a change in the marking of the Petri Net is performed and every input place lose the number of marks indicate by the weight in the arc connecting the Transition with that input place. On the other hand, every output place gain the number of places inidicated by the weight in the arc connecting the Transition with that output place. In figure 2 are shown different cases for the triggering of a transition.

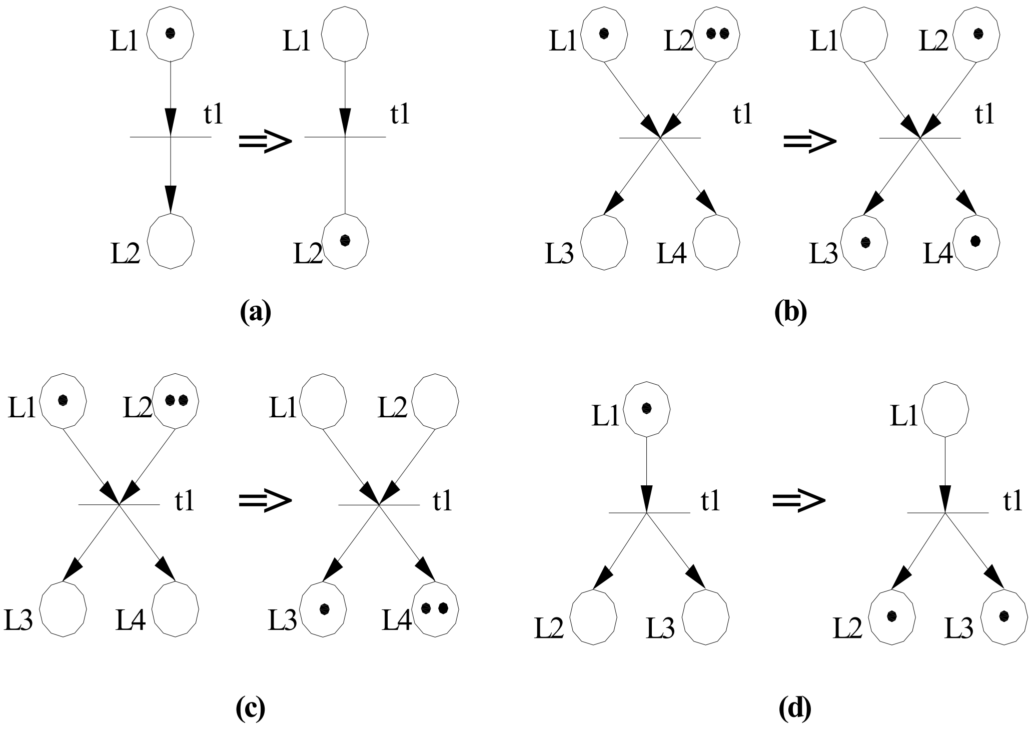


Figure 2. Different triggering cases for some Petri Nets.

3. Petri Nets showing relationships amog sub-processes.

3.1 Task Sequencing

Figure 3 shows how different task can be done in sequential way.

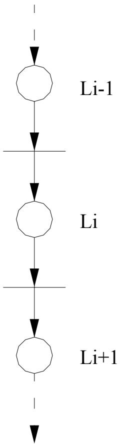


Figure 3. Task Sequencing

3.2 Task Selector

Figure 4 shows a task selector. This graph is useful when we need in a system an EXOR function between transitions t_1 & t_2.

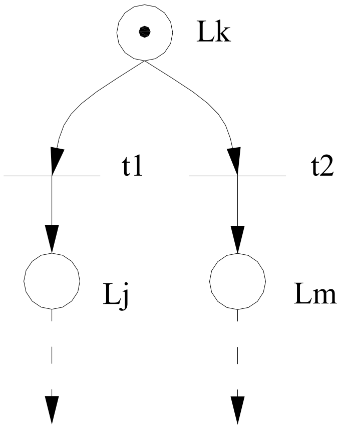
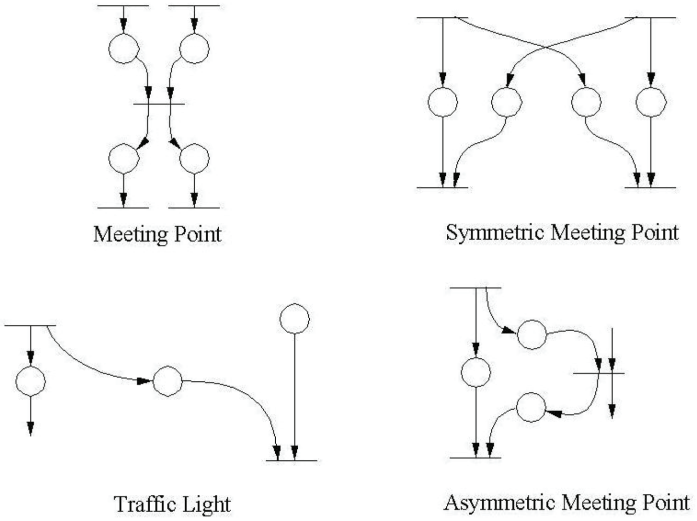


Figure 4. Task Selector

3.3 Task Synchronization

This is a very useful concept to synchronize two or more ub-processes assuming that they may have different running times or different evolution in time. Any of the processes can be out of phase respect to the others, however this model provides a wait state for any of the processes until the triggering of the transition. Synchronization is achieved when, being validated by the proper marking on the input places, the event occurs triggering the transition.


Meeting Point

Symmetric Meeting Point

Traffic Light

Asymmetric Meeting Point

Figure 5. Different models to synchronize processes

3.4 Tasks Concurrency

Figure 6 shows how concurrent processes are represented.

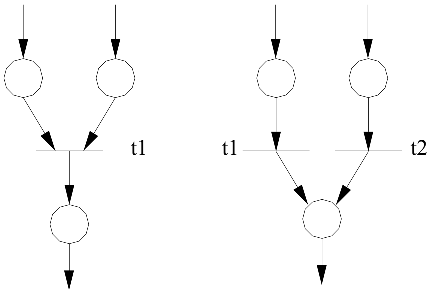


Figure 6. Tasks Concurrency

3.5 Sharing Resources

Sharing resources is done when two or more sub-processes are using a unique resource in the system –such a robot to load/unload two conveyors. The objective is to avoid the malfunction of one of the processes because the resource is not available.

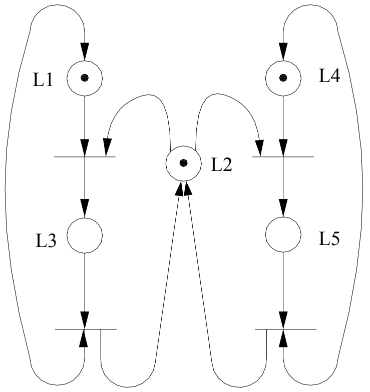


Figure 7. Sharing Resources

4. Equation of a Petri Net.

Figure 8 shows a Petri Net which will be used to explain the Petri Net equation.

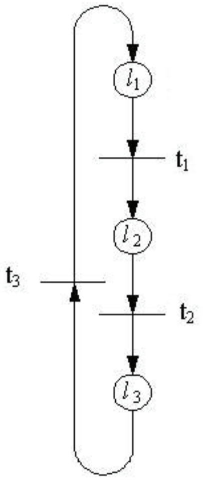


Figure 8. Sharing Resources

4.1 Input Matrix

This matrix provides the connectivity information from Places to Transitions. In this case Place l_1 goes to Transtion t_2, Place l_2 goes to Transition t_2 and so on. See figure 8.

$$E = \begin{array}{c} \begin{array}{ccc} l_1 & l_2 & l_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array} \begin{array}{c} t_1 \\ t_2 \\ t_3 \end{array} \qquad (2)$$

4.2 Output Matrix

This matrix provides the connectivity information from Transitions to Places. In this case Transtion t_2 goes to Place l_1, Transtion t_2 goes to Place l_2 and so on. See figure 8.

$$S = \begin{array}{c} \begin{array}{ccc} t_1 & t_2 & t_3 \end{array} \\ \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array} \begin{array}{c} l_1 \\ l_2 \\ l_3 \end{array} \qquad (3)$$

4.3 General Matrix of a Petri Net

The two last matrixes are used to define the Net Matrix given by:

$$A = S - E \qquad (4)$$

4.4 Evolution of a Petri Net

The evolution of the Petri Net is related with the "movement" of the marks in the graph from one place to another. The recursive equation is given by:

$$\mu_{k+1} = \mu_k + AV_k \qquad (5)$$

where:

- μ_k represents the marking vector before the evolution of the net.
- μ_{k+1} represents the marking vector after the evolution of the net.
- A represents the General Matrix of the net.
- V_k represents the Transitions that are been triggered in the current evolution step of the net.

4.5 A working Petri Net

Figure 9 shows a Petri Net before and after the evolution step (trigger of t_1).

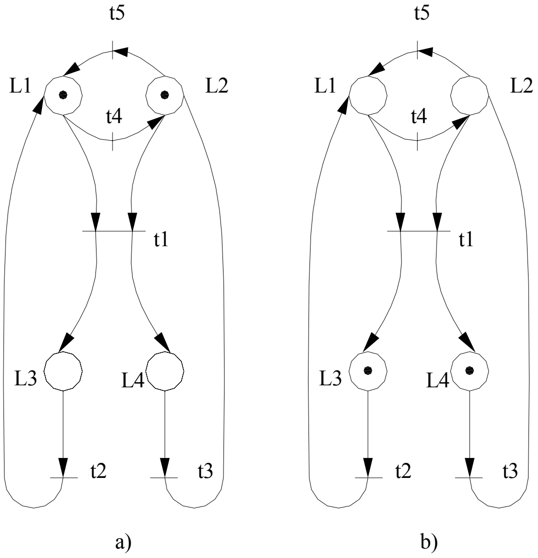


Figure 9. A Petri Net before and after the trigger of transition t_1

5. Application Examples.

5.1 Wagon Control

Figure 10 shows a wagon that moves along a rail by the action of signals R (right) and L (left). Extreme positions are detected by sensors A & B. Button M starts the sequence when wagon is in initial position.

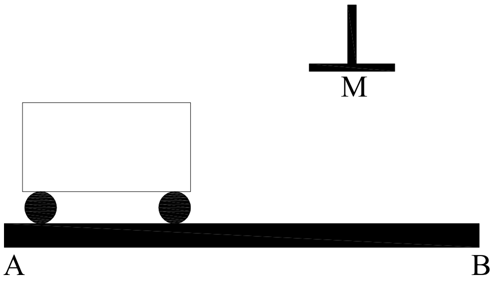


Figure 10. Wagon Control

There are three places (l_1, l_2 & l_3): Idle, Moving from A to B & Moving from B to A. There are three transitions (t_1, t_2 & t_3): Switch M, wagon detected by sensor A & wagon detetcted by sensor B. When M is press, the wagon moves from A to B, once reaches this point goes back to A. Sequence can only starts when wagon is in Idle again.

Figure 11 shows the Petri Net for this system.

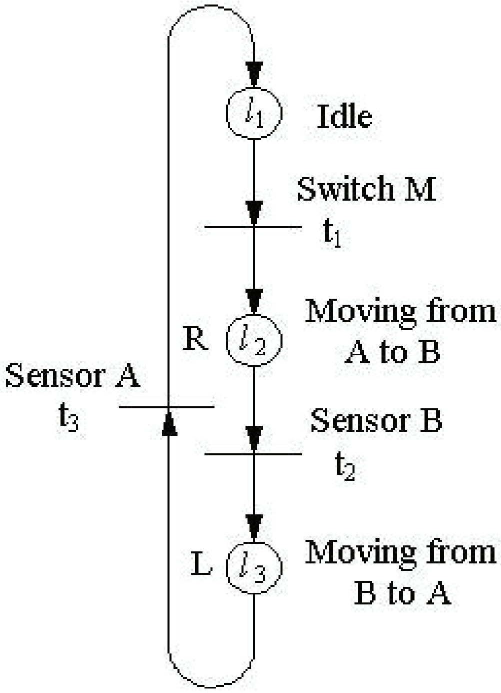


Figure 11. Petri Net for the Wagon Control system

Following the matrixes for this system:

$$E = \begin{array}{c} \begin{array}{ccc} t_1 & t_2 & t_3 \end{array} \\ \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \end{array} \begin{array}{c} l_1 \\ l_2 \\ l_3 \end{array} \quad S = \begin{array}{c} \begin{array}{ccc} t_1 & t_2 & t_3 \end{array} \\ \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \end{array} \begin{array}{c} l_1 \\ l_2 \\ l_3 \end{array} \quad V_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \quad \mu_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{6}$$

$$A = S - E$$

$$A = \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \tag{7}$$

Where $\mu_{k+1} = \mu_k + AV_k$ then:

$$\mu_1 = \mu_0 + AV_0 : \mu_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \tag{8}$$

$$\mu_1 = \mu_0 + AV_0 : \mu_1 = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ -1 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \tag{9}$$

$$\mu_1 = \mu_0 + AV_0 : \mu_1 = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} -1 & 0 & 1 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{10}$$

6. Simulation Software

In order to simulate the Petri Nets, a software kit was developed. Although there are some simulators available, we took the decision to develop our own as the main goal and to have a software system able to be interfaced with the real world. The software was developed in Lab Windows CVI, having a Graphic User Interface which allows the user to create easily the net that represents the system that is needed to simulate. Figure 12 shows the tipycal interface of the system.

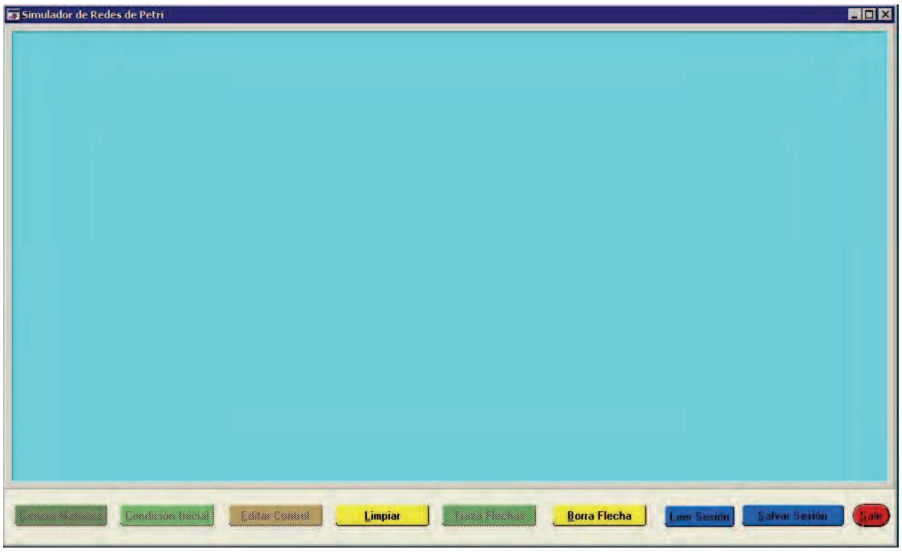


Figure 12. Main view of the simulator

The operator inputs the data related with the PN structure, the information is then interpreted by the simulator which shows the development of the PN. On figure 13, the Petri Net shown on figure 11 was introduced to the simulator.

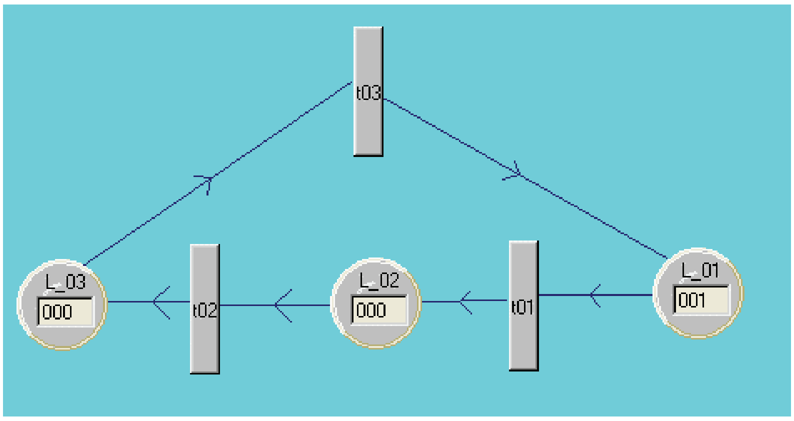


Figure 13. Petri Net for the Wagon Control system

Figures 14, 15 & 16 shows the status of the Petri Net when transitions t_1, t_2 & t_3 are triggered.

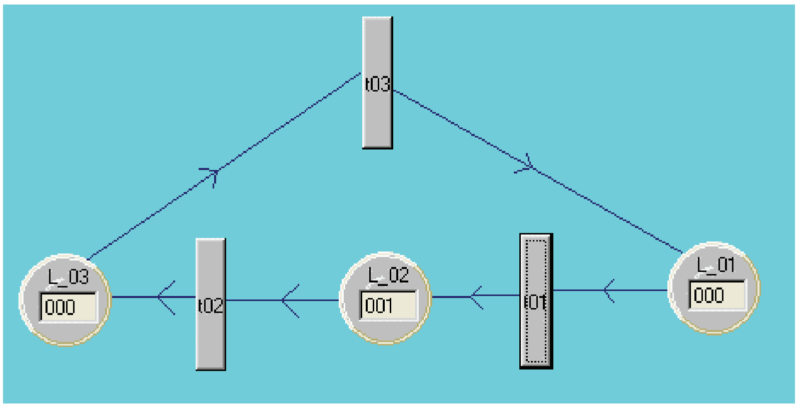


Figure 14. Petri Net for the Wagon Control system after t_1 is triggered

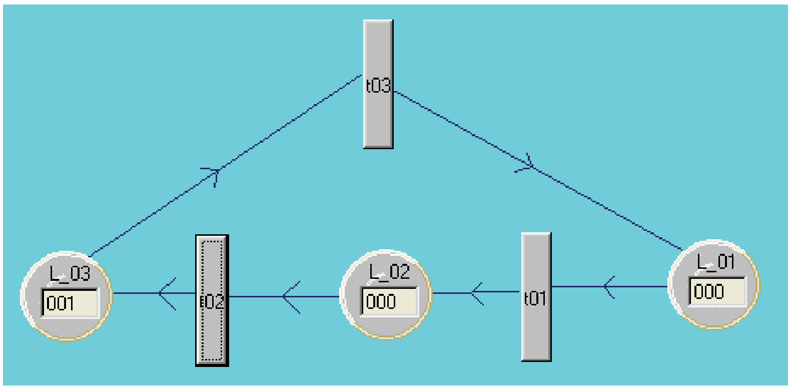


Figure 15. Petri Net for the Wagon Control system after t_2 is triggered

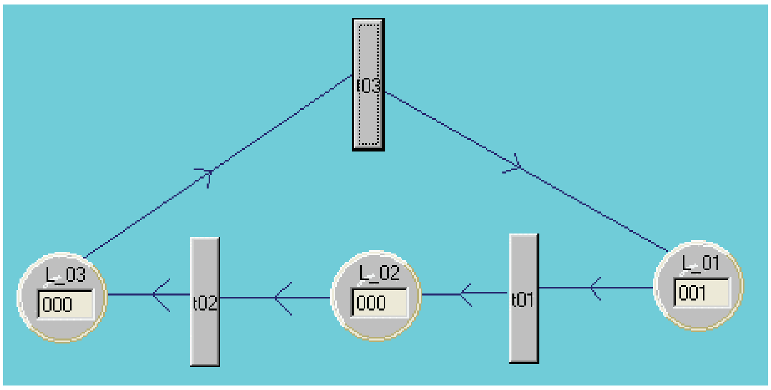


Figure 16. Petri Net for the Wagon Control system after t_3 is triggered

Inside the simulator each circle represents a Place where the wagon is moving to the right, to the left, or simply is on the idle state waiting for the button M to be pushed (transition t_1) to start a sequence.

It is worth to say that this simulator can easily be used as a Control System just by adding an Acquisition Board for sampling the actual signals which in turn can activate the transitions. The purpose with this was to develop a non expensive control system which can be used by very small companies that do not have the capital to buy very expensive equipment. In comparison with a PLC controller the cost involved when implementing this approach is very low and also offers an intuitive way of developing the control system.

7. Benefits of a Petri Net.

Petri Nets have the following benefits:

a) They have a graphic representation which allows a simply and secure way to observe how the system is working.
b) They have semantics well defined that specify the net.
c) They can be used to represent several types of systems.
d) They have an explicit description for states and actions.
e) They provide interactive simulations where the results are shown directly on the graphic diagram of the net.

In future research this system can be used on the control of production lines where series of events (discrete events) need to be controlled, such as a robot feeding a conveyor, a conveyor moving cans or packages into storage areas, etc.

8. References

Mireles, C. et al. (2004), *Flexible Manufacturing System Simulation using Petri Nets*
http://www.lurpa.ens-cachan.fr/grafcet/generalites/presentation_uk.html

Manufacturing the Future

Edited by Vedran Kordic, Aleksandar Lazinica and Munir Merdan

ISBN 3-86611-198-3

Hard cover, 908 pages

Publisher Pro Literatur Verlag, Germany / ARS, Austria

Published online 01, July, 2006

Published in print edition July, 2006