MODELING HYBRID SYSTEMS USING PETRI NETS

MÁRCIA V. COSTA MIRANDA† AND ANTONIO MARCUS N. LIMA‡

†*Departamento de Ciências Fundamentais e Sociais, email: miranda@dee.ufpb.br*
‡*Departamento de Engenharia Elétrica, email: amnlima@dee.ufpb.br*
Universidade Federal da Paraíba
Caixa Postal 10.105, 58109-970 Campina Grande, PB, Brazil

Abstract— The focus of this paper is modeling, simulation and analysis of hybrid systems using Petri nets. A systematic procedure for building the hybrid model is proposed. In the proposed methodology the designer decides what components should be represented by continuous-time dynamics and what should be represented by discrete event dynamics. This decision is based on how the continuous-time dynamics of a given component affects the global plant behavior. In the mathematical framework adopted for this investigation the open-loop or free behavior of the plant and the controller are represented separately. In this approach, the nonlinear characteristics of the hybrid systems are taken into account to provide a more realistic representation for the plant. The utilization of the proposed methodology is illustrated with a liquid-level plant.

Key Words— Hybrid Systems, Hybrid Petri Nets, Colored Petri Nets

1 INTRODUCTION

Systems that exhibit discrete event and continuous-time features are called *hybrid systems*. The continuous-time and discrete event dynamics interact and one influences the other. For the continuous-time part this results in abrupt changes due to discrete switchings in the vector field or jumps in the states. On the other hand, the continuous-time dynamic influences the discrete part by generating discrete events (Pettersson and Lennartson, 1995; Bengt Lennartson and Tittus, 1994).

The study of hybrid systems has become more important in the last years due to the increased complexity of modern automation systems. In the earlier attempts the models proposed to describe hybrid systems were split into purely continuous-time or purely discrete event approaches. Despite their simplicities these approaches were not able to properly represent the actual hybrid behavior. Then, it is important to investigate formal modeling approaches that can fully represent the complete dynamics exhibited by such systems.

Formal methods simplify the understanding of the system dynamics, allowing to identify the main variables and parameters, help to check the structural and dynamic properties and also help its implementation. The modeling and specification of the hybrid systems is a hard job due to the use of multiple mathematical paradigms which make difficult the definition of the properties of the system (Genrich and Schuart, 1998).

Among others formalisms available to study hybrid systems, Petri nets provide an integrated way of combining discrete and continuous aspects in a single structure and very powerful tools to perform analysis, modeling, graphical description and simulation are available.

In this paper, we study a systematic procedure for specifying, modeling and simulating hybrid systems using Petri nets. In this approach, the nonlinear characteristics of the hybrid systems are taken into account and a more realistic representation is obtained. The methodology has been applied to study the plant studied in (Pettersson and Lennartson, 1995). We make a comparative study between Hybrid Petri Nets and Colored Petri Nets implementations and, for that, we have employed Design/CPN with time stamps (CPN group at the University of Aarhus, 1999) and the VisualObjectNet (Drath, 1999).

2 MODELING METHODOLOGY

The development of models for hybrid systems is a recent and relatively rich research field. Different models have already been proposed (Allen Back and Myers, 1993; Antsaklis et al., 1993; Branicky, 1996; Nerode and Kohn, 1993; Pettersson, 1999; Pettersson and Lennartson, 1995; Tavernini, 1987) but we are far from establishing an ultimate modeling paradigm.

The hybrid systems models are based on different paradigms of modeling (Tavernini, 1987; Allen Back and Myers, 1993; Nerode and Kohn, 1993; Antsaklis et al., 1993; Branicky, 1996; Pettersson, 1999). The use of one or another paradigm corresponds to a given "abstracting cut" and aims at studying a specific class of problems. Independently of the modeling paradigm, we need to use a systematic procedure to construct the plant model. With this plant model we can perform system analysis to check the structural and dynamic properties and further synthesize a control law to insure the performance and disturbance rejection required for the given application. This systematic procedure is not available so far.

For the purposes of the present study we used the model introduced by Pettersson (Pettersson, 1999; Pettersson and Lennartson, 1995). This model can represent the most relevant hybrid system behavior like vector field switching and state jumps as well as the influence of the continuous-time dynamics over the discrete event dynamics. Another important feature of this model is the explicit separation between the plant model, open loop hybrid plant (OLHP), and the controller model which further simplifies the analysis of the open-loop performance and the achievement of the closed-loop specification.

Petri nets are used to study hybrid systems because they have several relevant features like, for example, powerful graphical description, capability to integrate the control model and the system behavior in the one single model. Besides, they naturally model systems with parallelism, nondeterminism, concurrence that are not easy to represent with other formalisms.

As mentioned before we decided to use Design/CPN (CPN group at the University of Aarhus, 1999) and VisualObjectNet (Drath, 1999) packages to study hybrid systems. For both packages it is required to provide a Petri net model that represent the plant behavior. The main purpose of this paper is to show the modeling steps required to derive these Petri nets as well as to demonstrate how to access some features of the open-loop and the closed-loop performance through these tools.

The modeling approach we have used can be explained with the help of the diagram shown in Fig. 1.

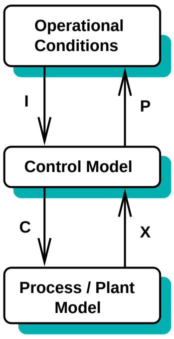

Figura 1. Modeling Method

The modeling methodology requires the specification of the plant behavior and the definition of its operational conditions (I) with respect to the plant state (P). The controller generates orders (C) according to the events (X) which represent the changes in the plant (switchings and jumps). In the first place we define the operational conditions of the plant and consider separately the plant and the controller.

A systematic procedure used to derive the plant model represents a tradeoff between two very distinct objectives. At the same time we want to be sure that the obtained model can express what we desire about the plant and we do this by restricting the degree of freedom of the designer without blocking its creativity. The modeling procedure that we propose in this paper consists of the following steps:

1. Identify the discrete and continuous components: By analyzing the plant, the designer defines what components should be represented by its continuous-time dynamics and what should be represented by discrete event dynamics. The decision to model a given component as discrete event type when it is not inherently discrete is based on the influence of its continuous-time dynamics at the macroscopic level, i.e: at the plant level and not the element level. If this influence is negligible the component can be modeled as a discrete event type. As we will show in the following, for the plant discussed in this paper, one discrete element is, for example, the valve since its continuous-time dynamic for closing or opening it is not relevant at the macroscopic level.

2. Build the plant model:

2.a *Discrete components model.* The model is a Petri net structure where the places are added for representing each component state and the transitions represent each state change. For example, the valve model has two places, one representing the ON and other for the OFF state and two transitions representing the state changes.

2.b *Continuous components model.* The model results from the use of several basic physical laws applied to the specific component as well as its interactions with the other components. The resulting differential equations must be translated into a Petri net structure. We have considered the use of Hybrid Petri Net (HPN) and Colored Petri Net (CPN) structures for representing the continuous dynamic. In Colored Petri Net, the differential equations which model the continuous behavior must be discretized and the reason for this will be explained in Section 5.

2.c *Interaction between discrete and continuous-time components.* The connection between discrete and continuous components is called interface and represents the event generator of the interface. It is responsible for generating the plant events which represent the corresponding boundaries in the continuous state-space of the plant. In HPN, this is modeled by continuous-time transitions where transition inscriptions represent the firing speeds. In CPN, it is required to make some modifications in the net derived in the previous step. These interactions will be better explained in Sections 4 and 5.

3. Build the controller model: The open-loop behavior is obtained in the previous step and represents the free behavior of the plant without any control. With the desired operational conditions and the plant model the designer can define the controller. The controller model is also translated into an HPN or a CPN which are patched into the plant model and will restrict the plant behavior to achieve the desired specification. The description of the control law using Petri net structures is discussed in Sections 4 and 5.

3 HYBRID SYSTEM

We have used as an example the same plant investigated in (Pettersson and Lennartson, 1995). This plant configuration is illustrated in Fig. 2. However, we have also included a more realistic representation of the continuous-time elements by considering some nonlinear features.

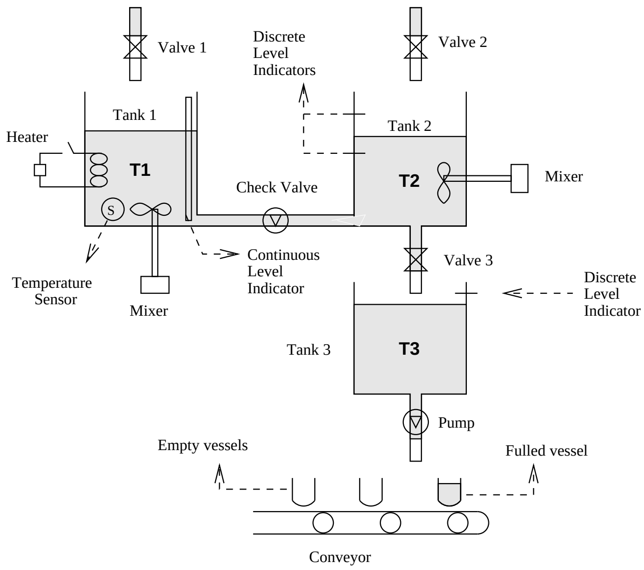

Figura 2. Liquid-level plant configuration

The plant represented in Fig. 2 consists of three coupled tanks representing the continuous-time dynamics. The discrete event dynamics is defined by the valves, pumps, level sensor states and the conveyor. In this case, the valves, pumps and conveyor are regarded as a discrete event elements since its continuous-time dynamics are not relevant the macroscopic level. i.e., the dynamics of the three tanks dominate the continuous-time plant behavior. The continuous-time dynamics is influenced by discrete event phenomena, i.e., switching vector field and state jumps. The continuous-time evolution influences the discrete part by generating events which affect the discrete states.

Vector field switching results from the state changes of the actuators like valves (open/closed) and pumps (on/off). *State jumps* occur because of the ignored continuous-time dynamics. This can be illustrated in the plant example by the pump filling a vessel on the conveyor; if the flow of the pump is fast compared to the input flow from Tank 3, the time required to fill a vessel can be neglected.

3.1 Continuous-time Dynamics

Storage tanks and their connecting pipes networks are quite common in many industrial plants. Usually the continuous-time dynamics of such systems can be accessed through the "head" or liquid height producing hydrostatic pressure and the pipe flow. The mathematical model for liquid-level and pipe flow systems is obtained by applying the mass conservation principle that prescribes: $\dot{m} = \sum_n q_{in} - \sum_m q_{out}$

where m is the fluid mass within the tank, $\sum_n q_{in}$ is the total mass flow entering the tank and $\sum_m q_{out}$ is the total mass flow leaving the tank (Franklin et al., 1994).

Fig. 3 shows an isolated tank filled with an incompressible fluid used to illustrate how to derive the continuous-time model. Applying the mass conservation principle to this tank leads to

$$\frac{d(\rho Ah)}{dt} = q_{in} - q_{out}, \; q_{out} = \frac{1}{R}(p_1 - p_a)^{\frac{1}{\alpha}} \quad (1)$$

where $p_1 = \rho gh$, A is the cross section area of the tank, R is the Reynolds number, α is a constant, $1 < \alpha < 2$, ρ is the fluid density, g acceleration of gravity and p_a is the pressure at the pipe end. The liquid flow through orifices that exhibits wall friction which can be nonlinear depending on the flow conditions. When the Reynolds number is less than 2300 the flow is laminar and $\alpha = 1$. However, when the Reynolds number is greater 2300 the flow is turbulent and consequently $\alpha \neq 1$ which makes the continuous-time dynamics of the tank quite nonlinear (Franklin et al., 1994). Equation (1), with appropriate parameters, can be used to describe all the tanks of the plant. The interaction of

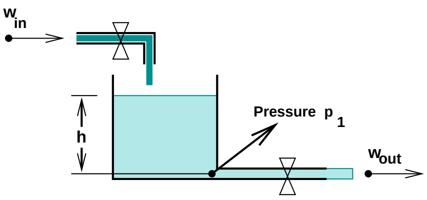

Figura 3. Liquid-level and pipe flow tank

the tanks appears, like for example between Tank 1 and Tank 2 of Fig. 2, in the term q_{out} that depends on the pipe connecting these tanks and the level difference between them. The flow between Tank 1 and Tank 2 is described by

$$q_{12} = \frac{(\rho g)^{\frac{1}{\alpha}}}{R}(h_1 - h_2)^{\frac{1}{\alpha}}, h_1 > h_2 \quad (2)$$

$$q_{21} = \frac{(\rho g)^{\frac{1}{\alpha}}}{R}(h_2 - h_1)^{\frac{1}{\alpha}}, h_2 > h_1 \quad (3)$$

while the flow between the Tank 2 and Tank 3 is described by

$$q_{23} = \frac{(\rho g)^{\frac{1}{\alpha}}}{R}(h_2 - p_a)^{\frac{1}{\alpha}} \quad (4)$$

In the present study we have considered $\alpha = 2$ for all the pipe flows.

3.2 Operational Conditions and Control Law

For the plant given in Fig. 2 it is assumed that all valves are open and all the tanks are empty as the initial operational condition. It is also assumed that the production of the vessels is unlimited. In this case, the control for the plant aims at insuring that the levels of the tanks remain between the overflow and underflow limits. This is a very simple law and was used only to illustrate the use of the methodology. Of course, more complex control laws can be included if required.

4 HYBRID PETRI NET

Hybrid Petri nets (HPN) are high level Petri nets, consisting of a combination between ordinary and continuous Petri net. HPN were introduced by David and Alla (David and Alla, 1992), allowing to represent continuous-time and discrete aspects of hybrid systems into a single framework. HPN contains continuous places and transitions, discrete places and transitions and the marking of a continuous places is a real number. The continuous-time part of the HPN can be used to represent the continuous-time dynamics of the tanks and the discrete part represents the discrete sensors and actuators.

4.1 Plant Model

In the construction of the HPN model, the designer must define what discontinuous events should be considered and what kind of structures should be used to model these events. In Fig. 4 is illustrated the HPN structures to model the discrete phenomena of hybrid systems.

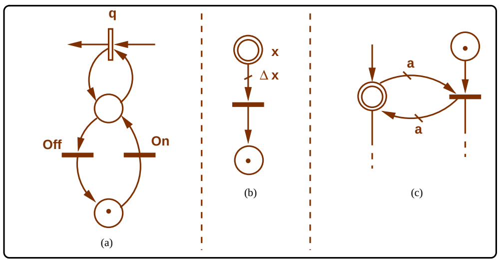

Figura 4. Switching structures: (a) Switch (b) State Jump (c) Boundary Crossing

The *vector field switching* are modeled in HPN by using switching elements as shown in Fig. 4(a). The state of the discrete part affects the flow through the continuous-time transition labelled as q. If the discrete transition On fires the marking changes to the next discrete state, resulting in a flow through the continuous transition. When the discrete transition Off fires the flow stops.

State jumps can be represented by connecting a continuous place to a discrete transition, as shown in Fig. 4(b). The weight of the intermediate arc (Δx) defines the amount of state change corresponding to the jump.

The continuous-time state can affect the discrete behavior, as shown in Fig. 4(c). The discrete transition fires only when the continuous-time marking has reached the level corresponding to weight a.

Using these structures and following the procedure described previously the plant model in terms of HPN can be obtained as given in Fig. 5. This model was implemented by using the Visual Object Net ++ tool (Drath, 1999). This tool allows to attach equations to continuous transitions which represent the firing speeds. Besides, VisualObject Net provides three kinds of arcs: normal, inhibitor and static test arcs. The former ones are similar to the arcs normally used with Petri nets. However, there is no token flow when places and transitions are connected by static test arcs. The utility of these arcs is only to take the marking value of a place.

The HPN model given in Fig. 5 represents the plant (OLHP) with the controller structure. The transition sensors q_1, q_2, q_{12} and q_3 model the fluid flows and are affected by switching structure shown in Fig. 4(a). The places h_1, h_2 and h_3 model the levels of Tanks 1, 2 and 3, respectively. The transition q_1 models the input flow of Tank 1 when the valve V1 is open and has a constant flow. The transition q_2 models the input flow of tank 2 when the valve V2 is open and has a constant flow too. The equation representing these flows is similar to (1).

The flow between the tanks T1 and T2 is modeled by the transitions q_{12} (q_{21}) and *T_Ret*. This can be represented by arcs from continuous place affecting discrete transitions of the switch representing the check valve. The flow speeds of these continuous transitions are determined by functions fq12() that implements (2) and fq21() that implements (3). The flow between the Tanks T2 and T3 is modeled by tran-

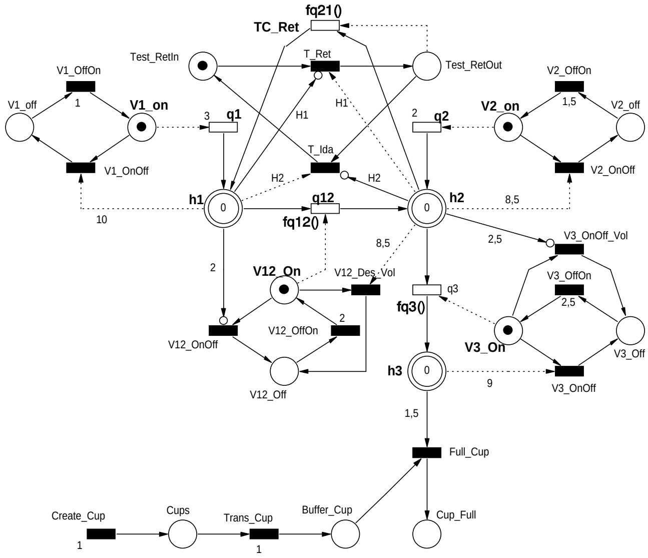

Figura 5. Hybrid Petri net for the OLHP

sition q3. The equation implemented to represent this flow is similar to (4), modeled by function fq3().

The continuous place h_3 and the discrete place Buffer_Cup are connected by a discrete transition which is fired when h_3 is above a given limit and there is, at least, one token at Buffer_Cup. When this transition fires a token is added at Cup_Full place to model the filling of the cup.

4.2 Controller

The control law is modeled by the connection between the places representing the tanks levels and the transitions representing the valves. This connection is represented in the HPN model by *static test arcs* which check the level of the tanks with the maximum allowed levels (overflow). An inhibitor arc is connected between the places representing the levels of tanks and the transitions representing the valves to verify the minimum levels of the tanks (underflow).

4.3 Simulation

The simulation of this plant model was executed within the Visual Object Net ++ tool. Fig. 6 shows the level of Tank T1. It can be noted that the level never exceeds the overflow limit (10l) and never falls bellow the underflow limit (2l). Fig. 7 shows the level of tank T2. It can also be noted that the level never exceeds the overflow limit (8.5l) and never falls bellow the underflow limit (2.5l). Fig. 8 shows the level of tank T3 and its behavior also respects the limits specified by the control law.

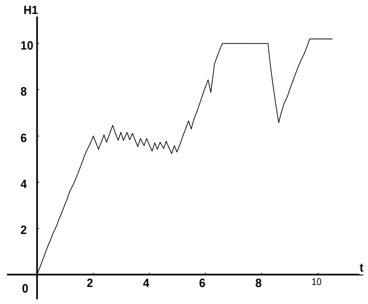

Figura 6. Time evolution of the level of Tank of T1

5 COLORED PETRI NET

The plant shown in Fig. 2 was also represented by Hierarchical Colored Petri net (HCPN). As it has been done with HPN, the designer must translate the hybrid plant dynamics into the HCPN framework.

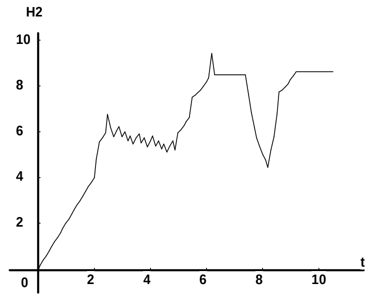

Figura 7. Time evolution of the level of Tank of T2

The representation of the discrete event components is almost direct and the *vector field switching* are also modeled as we did for HPN using the elements shown in Fig. 4(a). In this case the resulting structure can be much more compact since the HCPN allows the use complex data types. With respect to the plant model, the vector field switching is illustrated by the connections between the place Valvules to the transition Flow_T1 as shown in Fig. 10. Similarly, the *state jumps* can be represented as shown in Fig. 4(b). In the case of the plant model this is represented by the connections between place t_flow_level_3 and transition Flow_Pump where the inscription Calc_nivel in the arc represents a function for computing the state change due to the jump in Fig. 11.

For implementing continuous features in HCPN, it was necessary to make some modifications to adapt them to discrete framework supported by DesignCPN. For this a discrete time model was defined to represent the liquid fluid flow for a relatively short time period. Each tank is represented by a place (t_flow_level_1, t_flow_level_2 and t_flow_level_3) and pipe connection with other tank is done through a transition. The interaction of the continuous-time dynamics was represented in the following way: i) The fluid flow from Tank 1 through valve V1 and from Tank 2 through valve V2 are represented by transitions Flow_T1 and Flow_T2, respectively. The inscriptions in the arcs Calc_nivel represent the functions that implement the equation $\dot{m} = \sum_n q_{in} - \sum_m q_{out}$; ii) The fluid flow between Tank 1 and Tank 2 is represented by transition Flow_T12 and only exists when the check valve is open. The inscriptions in the arcs Calc_q12_T1 and Calc_q12_T2 represent the functions that implement the flow equivalent of (2) and (3), respectively; iii) The fluid flow between Tank 2 and Tank 3 is represented by transition Flow_T3. The inscriptions in the arc Calc_nivel_q3 represent the function that implement (4). All these differential equations are discretized by Euller first order approximation (Astron and Wittenmark, 1990). The discretized version for this continuous time differential equation is:

$$h_1(t) = h_1(t - \Delta t) + \Delta t(q_1 - q_{12}) \tag{5}$$

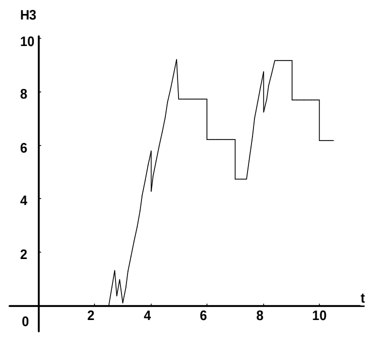

Figura 8. Time evolution of the level of Tank of T3

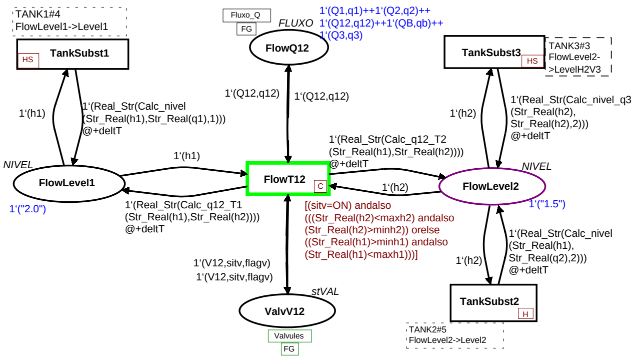

Figura 9. CPN of Plant model

For calculating h_2 and h_3 levels, equations similar to (5) were used. This approximation was been necessary because DesignCPN tool does not implement continuous time equations directly, so that we used recursive equations to complete the modeling process.

For this model seven pages were built: the Main_Page, one page for representing each tank and one page for the control of each tank. Main_Page, TANK_1, TANK_2 and TANK_3 model the plant of the process system. Control_TANK1, Control_TANK2 and Control_TANK3 model the control of the tanks. The page Control contains ML-functions to exhibit the simulation of the controlled system behavior. The page Performance_Page contains ML-functions to produce a diagram of the performance analysis. The page FluidFlowbetweenthreeTanks1 is the page with the messages diagram of the simulation process.

5.1 Plant Model

The plant model is shown in Fig. 9. The markings in the places FlowLevel1 and FlowLevel2 represent the level of Tank 1 and 2, respectively. The substitution transition TankSubst1 models the behavior of Tank 1 according to the state of valve V1. The substitution transition TankSubst2 models the behavior of Tank 2 according to the state of valve V2. The substitution transition TankSubst3 models the behavior of Tank 3 and the vessels on the conveyor. The transition FlowT12 fires when the valve V12 is ON (open), representing the flow between tanks 1 and 2 (q_{12} or q_{21}). The flow between these tanks is determined by solving differential equations very similar to (1).

Tank 1: The model for the Tank 1 is shown in Fig. 10 and represents the flow on Tank 1 according to the state of valve V1 ($q1$). The marking in place Level_1 represents the level of the Tank 1. The tokens in the fusion place Valvules represent the valves interacting with Tank 1 and the tokens in the fusion place Flow_Q represent the valves flows connected with Tank 1. The transition Flow_T1 fires when the guard condition is satisfied, that is, the situation of valve V1 is ON (open) and the level of the T1 is less than its underflow limit. The transition Control_T1 is a substitution transition representing the control on Tank T1. The model for the behavior of Tank 2 is very similar to the Tank 1 model, and will not be presented in this paper.

Tank 3 and Vessels: The model for the Tank 3 and the filling of the vessels is shown in Fig. 11. The marking in the place level_2_V3 represents the level of the Tank 2 and the marking in the place t_flow_level_3 represents the level of the Tank 3. The tokens in fusion place Valv_V3 represent the valves are interacting with Tank 3 and tokens in fusion place Flow_Q3 represent the valves flows which are connected with Tank 3. The transition Flow_T3 fires when the valve V3 is ON (open), the level of T3 is less than its overflow limit and the level of T2 is greater than its underflow limit. The levels of the Tank 2 and the Tank 3 are calculated according to the equation representing the flow on valve V3 ($q3$). The transition Control_T3 is a substitution transition representing the control on Tank T3. The transition Flow_Pump fires when the pump is active and there is enough material in Tank 3 to fill a vessel. The place Vessel represents vessels waiting to be filled on the conveyor and the place Fulled_Vessel represents vessels filled.

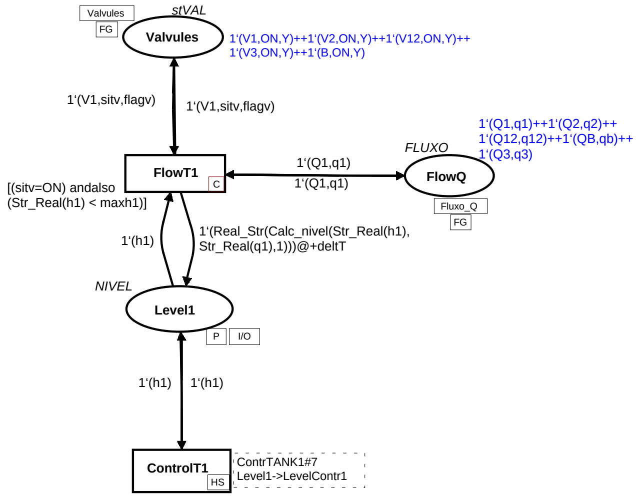

Figura 10. CPN of Tank 1 model

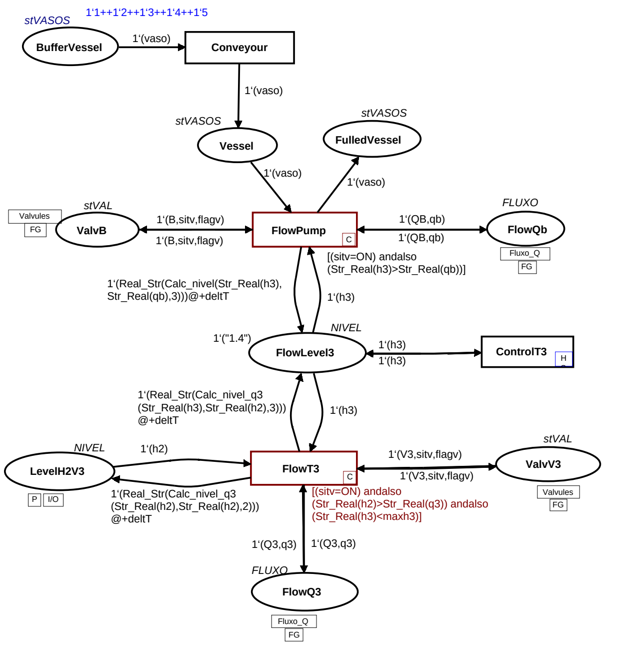

Figura 11. CPN of Tank 3 model

5.2 Controller

In Fig. 12 is illustrated the control on Tank 1. The control consists to check if the level of the tanks are not greater than its overflow limits and are not less than its underflow limits. The transition Test_Lim_H1 is fired when valve V1 is ON (open) and the Tank 1 level is greater than its overflow limit. In this case, the situation of valve V1 is changed to OFF (closed). The transition Open_V1 is fired when the level of the Tank T1 is less than or equal to its underflow limit, changing valve V1 state to ON (open). The control to the other tanks is similar to this model, so we will not present it in this paper.

5.3 Analysis of the model

The purpose of analyzing a CPN model is to verify if the system behaves correctly as well as to help the correction of

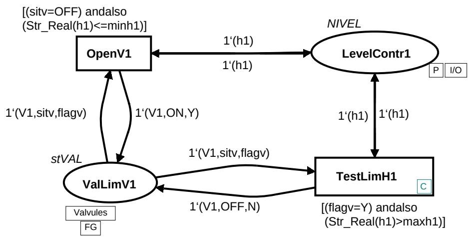

Figura 12. CPN representing the control on Tank 1

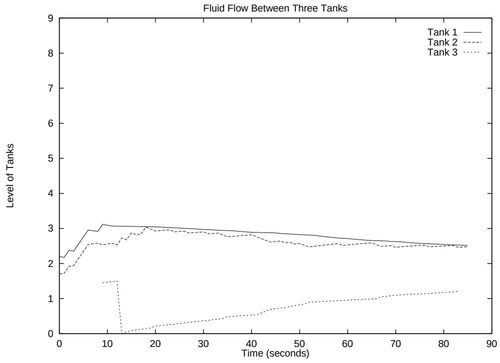

Figura 13. Performance Diagram

the system specification. The DesignCPN provides a set of tools to analyze CPN models under different scenarios for the system. In this case we used the functions of the ML functions to build the message sequence diagrams and the performance diagram as presented in the following.

The diagrams are obtained for one of the possible runs of the CPN model and the performance diagram illustrates the behavior of the system for given time interval.

In Fig. 13 is shown the performance analysis diagram of the controlled system. It can be noted the level of the three tanks never exceeds the overflow limit and never fall above the underflow limit, as specified by the control law. The simulation was done for the following scenario: the initial marking of the valves is 1'(V1,ON,Y)++1'(V2,ON,Y)++1'(V3,ON,Y) representing that V1, V2 and V3 are open. When $T3$ has enough fluid to fill a vessel then the pump B establishes a flow between $T3$ and the vessel. When the levels of $T1$ and $T2$ are greater than its overflow limits, the valves are closed. In this case, the fluid flow occurs between $T2$ and $T3$ until the level of $T2$ is less than the overflow value, then the flow between $T1$ and $T2$ is allowed. The *message sequence diagram* is shown in Fig. 14 and it is consistent with the performance analysis obtained before.

6 CONCLUSIONS

This paper has shown how to construct the model for an hybrid plant. The proposed procedure leads the designer such that the final model is a Petri net structure and he can decide to use either HPN or HCPN. In the context of this investigation the choice for one or another formalism relies on the availability of software tools for simulating and analyzing the model properties. By choosing HPN the designer will obtain a model that is more friendly from the visual point of view. However, the software tool for exploiting the model does not provide functions for verification of structural properties but can execute dynamic simulation. However, by choosing CPN the resulting model is not so friendly due to the necessary *ad-hoc* patches but the software tool provides much more support for model verification. In both cases it was possible to include the controller and observe the closed-loop behavior was correctly achieved with respect to the given specification. We have included non-

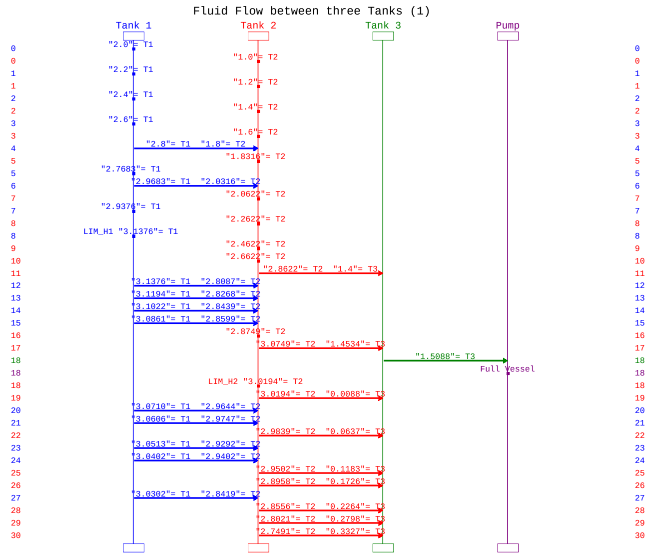

Figura 14. Messages Diagram

linear behavior of the hybrid systems through Petri nets, obtaining a more realistic model for the plant.

REFERENCES

Allen Back, J. G. and Myers, M. (1993). A dynamical simulation facility for hybrid systems, *Hybrid Systems* pp. 255–267. New York.

Antsaklis, P. J., Stiver, J. A. and Lemmon, M. D. (1993). Hybrid systems modeling and autonomous control systems, *Hybrid Systems* pp. 266–285. New York.

Astron, K. J. and Wittenmark, B. (1990). *Computer-Controlled System - Theory and Design*, Prentice-Hall International, Inc. Second Edition.

Bengt Lennartson, B. E. and Tittus, M. (1994). Hybrid systems in process control, *In Proc. of the 33rd CDC* pp. 3587–3592. Orlando, FL, USA.

Branicky, M. S. (1996). General hybrid dynamical systems: Modeling, analysis and control, *Hybrid Systems III - Verification and Control* .

CPN group at the University of Aarhus, D. (1999). Design/cpn - reference manual-version 4.0, *University of Aahus* . http://www.daimi.au.dk/designCPN.

David, R. and Alla, H. (1992). *Petri Nets & Grafcet - Tools for Modelling Discrete Event Systems*, Prentice Hall International. Great Britain.

Drath, R. (1999). Visual object net ++ - reference manual-version 2.0, *Ilmenau University of Technology - Department of Automatic Control and System Engineering* . Germany.

Franklin, G. F., Powell, J. D. and Emami-Nalini, A. (1994). *Feedback Control of Dynamic Systems*, Addison-Wesley Publishing Company.

Genrich, H. J. and Schuart, I. (1998). Modelling and verification of hybrid system using hierarchical coloured petri nets, *Proceedings of 3rd International Conferance on Automation of Mixed Processes: Dynamic Hybrid Systems (ADPM)* . Reims, France.

Nerode, A. and Kohn, W. (1993). Models for hybrid systems: Automata, topologies, stability, *Hybrid Systems* pp. 2309–2314. New York.

Pettersson, S. (1999). *Analysis and Design of Hybrid Systems*, PhD thesis, Chalmers University of Technology, Department of Signals and Systems.

Pettersson, S. and Lennartson, B. (1995). Hybrid modelling focused on hybrid petri nets, *2nd European Workshop on Hybrid systems* pp. 303–309. Grenoble.

Tavernini, L. (1987). Differential automata and their discrete simulators, *Nonlinear Analysis, Theory, Methods, and Applications* **11**: 665–683.