

# DEGENERATION METHODS IN INTELLIGENT BUILDING CONTROL SYSTEM DESIGN

---

Julio Arakaki

*Pontificia Universidade Católica de São Paulo/Unifiteo – jarakaki@pucsp.br*

Paulo Eigi Miyagi

*Escola Politécnica da USP – pemiyagi@usp.br*

*This paper presents a set of special requirements to development of intelligent building control software. It includes the degeneration technique (gradual reduction of the building service level). The method involves an organized activities sequence that results in artifacts such as models and control software specifications. These specifications assure the desired dynamic behavior of system and also include degeneration features. It also presents a specific example related with the control in Intelligent Building which has been adopted as case study and illustrates the application of the proposed method.*

## 1. INTRODUCTION

The productive systems and its respective control systems are organized in components with high cooperation degree. These control systems are becoming highly complex, distributed and its modules are strongly interacted.

In general, the control systems have some degree of robustness and security, in such way that they assure the system functionality in the environments for which they had been projected. However, the complexity of the systems and its control software increase the occurrence probability of unexpected events. In this manner, the implementation of new productive systems or the maintenance of already existing systems, the inclusion of degeneration techniques is desirable because, in fault situations, a gradual reduction of the services level in a system is necessary (Arakaki, 2004).

This paper considers a method (with the degeneration technique) for the control software design in productive systems (Intelligent Building). This method has activities organized in steps. Each step describes the techniques and/or artifacts generated through of models and specifications. The last step results in technical specifications of the control software, with the degeneration requirements incorporated.

The degeneration method considers the ISO/IEC 9126 standard (ISO/IEC 9126, 1998) for the development the control software. This software engineering standardization defines quality of software attributes for the control software

specifications. The Figure 1 presents the use of control requirements (with degeneration) and ISO/IEC 9126 standard to generate the specifications of control software architecture.

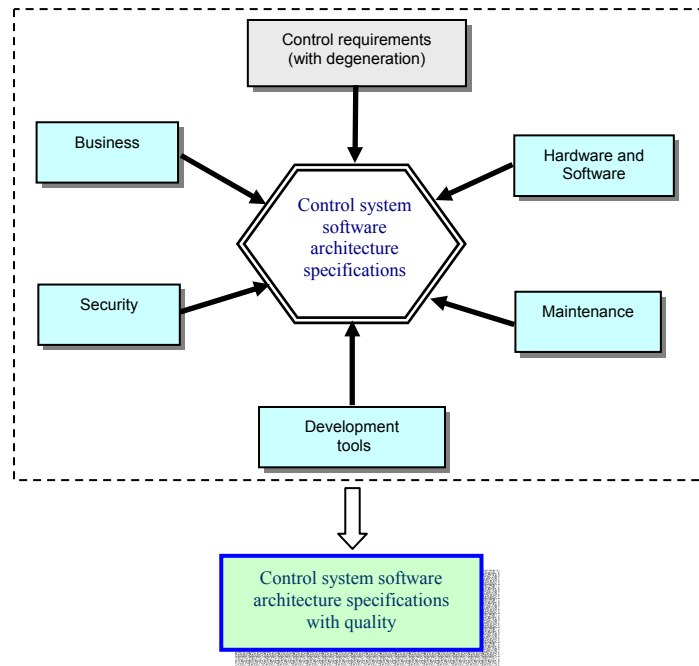


Figure 1 – ISO/IEC 9126 standard and the control requirements (with degeneration) included.

A typical architecture of the discrete event system control was presented in Miyagi (1988), Miyagi (1996) and Hasegawa (1998). In this case, to control a productive system<sup>1</sup>, the actuation is function of the command, detection signals and internal state (see Figure 2).

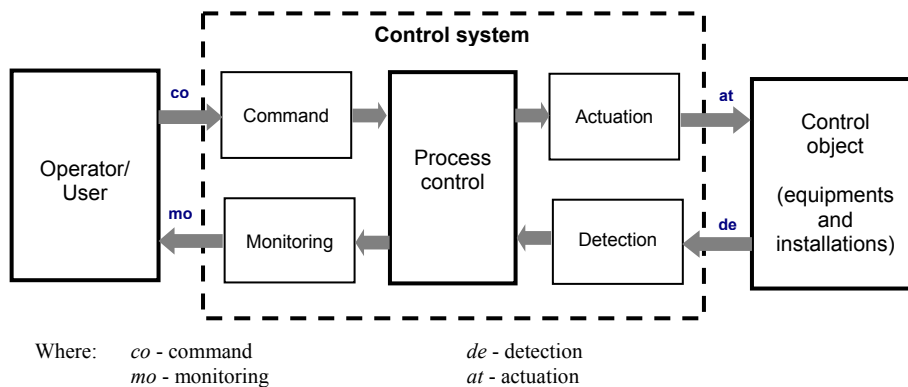


Figure 2 - Control system architecture.

<sup>1</sup> Productive System is based on the event occurrence (Discrete Event System).

The inclusion of the degeneration requirements in a discrete event system (DES) control is explained in the following item.

## 2. CONTROL SYSTEM WITH DEGENERATION

The new control system architecture is based on the typical architecture of DES, with the inclusion of the degeneration module.

The Figure 3 presents the control of discrete event system with degeneration module included. In this case, the module 'Degeneration' involves the following activities:

- Supervision of critical points;
- Re-configuration.

The actuation with degeneration ( $at_d$ ) is function ( $f_d$ ) of: command with degeneration ( $co_d$ ), detection with degeneration ( $de_d$ ) and internal state with degeneration ( $ei_{d_i}$ ):

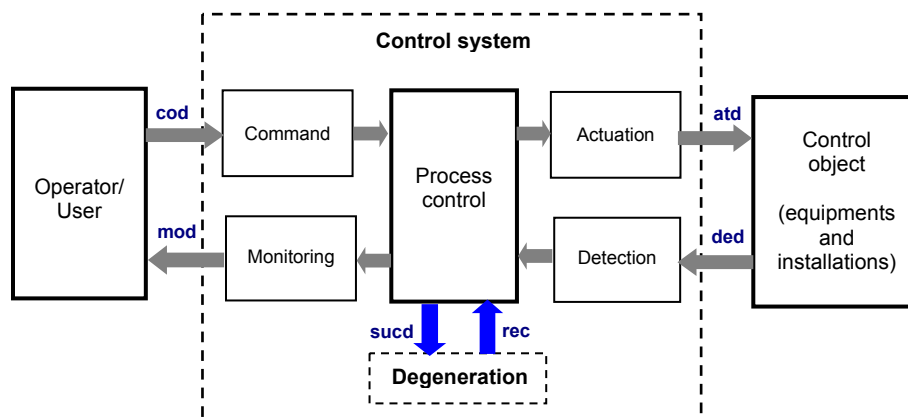
$$at_d = f_d(co_d, de_d, ei_{d_i}).$$

The next internal state with degeneration ( $ei_{d_{i+1}}$ ) is function of the re-configuration ( $rec$ ) and of the current internal state:

$$ei_{d_{i+1}} = g_d(rec, ei_{d_i})$$

The reconfiguration with degeneration ( $rec$ ) is function ( $h_d$ ) of the critical points ( $pc$ ) and of the supervision of these critical points<sup>2</sup> ( $su_{pc}$ ):

$$rec = h_d(pc, su_{pc})$$



Where: *cod* - command with degeneration      *ded* - detection with degeneration  
*mod* - monitoring with degeneration      *rec* - reconfiguration to degeneration  
*atd* - actuation with degeneration      *sucd* - critical point to degeneration

Figure 3 - Control system architecture with degeneration module.

<sup>2</sup> Component or configuration with high probability of occurrence of abnormal situations

The degeneration in a control system has the following functionality: initially an unexpected state is detected (through the identification and monitoring of the critical points in the system). After that, the context is evaluated, according to priority criteria for these critical points. Finally, the automatic mechanisms or with human intervention are set to reconfigure the control system to operate in a degraded way as the established programming. The abnormal situation is verified constantly by the components for the treatment of abnormalities (inserted in the control system) and by the components for the degeneration, until the regeneration condition is reestablished (return to the normal functionality, if possible).

### **3. CASE STUDY: INTELLIGENT BUILDING**

In a Intelligent Building (Arkin, 1995; Flax, 1991), each subsystem is managed by a control system that integrates the devices and the operations of that subsystem. These control subsystems interact with other control systems.

In this context, by applying the proposed method, the control software of the Intelligent Building is generated and allows increasing the efficiency and the functionalities of the control systems in unexpected situations.

The Intelligent Buildings had been chosen as study of case for the application of this method because they can be characterized as a productive system. They are based on the discrete events and distributed system and have subsystems with respective controls. These characteristics allow to verify and to analyze the considered method.

Normally, an intelligent building has:

- **Transport/movement** - elements that facilitate and organize the transport and the movement of people and objects. Example: the elevators, the rolling stairs, the rolling mats, and others;
- **Patrimonial and personal security and control of access** – allow the access and exit of people and objects in the interior of the building. The main devices are: electronic ratchets, cameras and internal TV circuit, bar codes readers, fingerprint reader, iris readers, special doors and alarms systems;
- **Energy management** - the energy supply for controls systems and for the building subsystems;
- **Fire** – fire identification system, alarm and fire occurrence control (with smoke detectors, automatic cut-fire doors and automatic devices to eliminate the fire);
- **Illumination** – system to interact with the elements as light bulbs, curtains and auxiliary batteries, to get an excellent illumination in all environments;
- **Ventilation and conditioned air (HVAC)** - systems for keeping comfortable environments (with ventilation, temperature and humidity systems).

Different types of information flow through the subsystems. One subsystem interacts with another. This interaction occurs through communication nets and protocols standards. Each subsystem must have its intelligent, autonomous and cooperative control. These are characteristics of distributed systems.

The inclusion of the requirements of degeneration in the subsystem of Intelligent Building control is illustrated of general form in Figure 4.

In this case, each subsystem has a degeneration component associate with a control component. All the subsystem has an interaction with the global degeneration module that changes the control requirements in a critical situation (not resolved in fault safe system). These changes allow that the control system acts with degeneration (keeping active the necessary operations) or with regeneration, returning to the normal functionality.

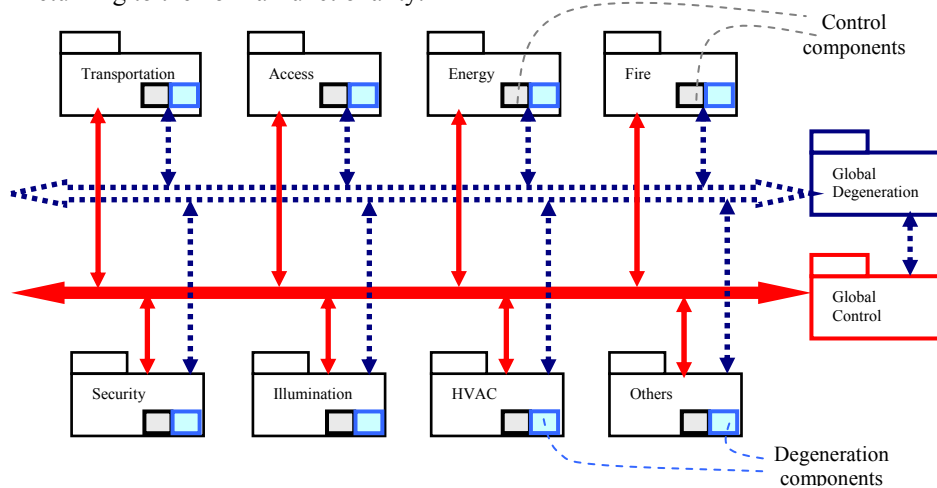


Figure 4. Subsystems in productive system (Intelligent Building with degeneration).

The considered method was applied for the subsystem of Transport and Heating Ventilation and Air Conditioning subsystem (HVAC) in a productive system (Intelligent Building). The modeling, analysis and control of the Intelligent Systems were explained in Bastidas Gustin (2002), Bastidas Gustin (2003) and Miyagi (2002).

#### 4. APPLYING THE DEGENERATION METHOD IN INTELLIGENT BUILDING

To demonstrate this method, subsystems of transport and HVAC (Villani, 2004; Villani, 2005) are used. In sequence, the necessary steps to generate the artifacts (control software specifications) are presented:

##### Step 1 – System critical points identification

In this step the critical points of the intelligent building are identified. The table 1 presents, as example, two critical points with its respective degenerations.

Table 1. Critical point samples.

Subsystem	Critical point	Degeneration
HVAC	(1) Conditioning of rooms	Reduction of the conditioning in less priority environments
Transportation	(2) Access to the specific floor	Allocation of another elevator

In the HVAC system (Figure 5), the low temperature required in the rooms is very important (ex: hospitals building). However, in days where the exterior use of the building and weather conditions are particularly critical, or still, in the case of imperfections in equipment, the water cold production system (used to set comfortable environments) can not be capable to remove all the necessary thermal load of the building. In this in case, the less priority areas, as reception, corridors, etc. are not conditioned, it must be restricted only to the natural ventilation (doors and windows available).

### Step 2 - Modeling of the degeneration component

In this step, the models represented by class diagram are developed to compose this productive system.

#### Static modeling:

It must be incorporate (to connect) the degeneration procedures to the control system model. To include the degeneration characteristics is necessary to associate, for each control component, a degeneration component.

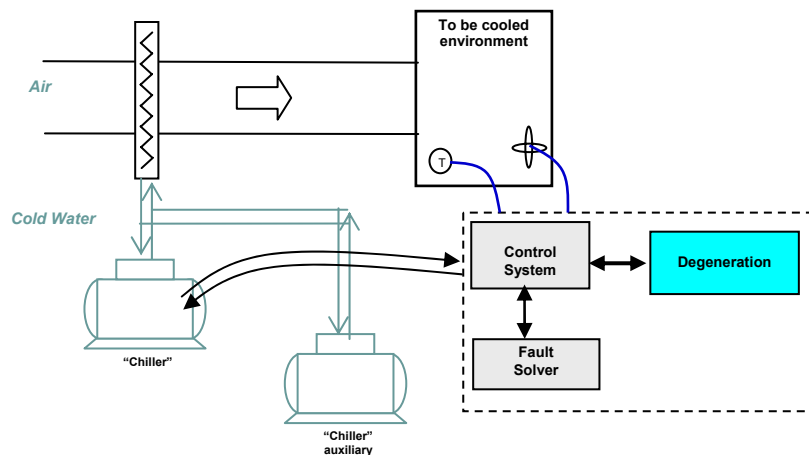


Figure 5 - HVAC sub-system with control and degeneration module.

Figure 6 illustrates a class diagram detailing the association of the control component and degeneration component for the HVAC sub-system. For example, the class structures for the Transport and Access sub-systems are similar, indicating in this way, a high probability of software reuse.

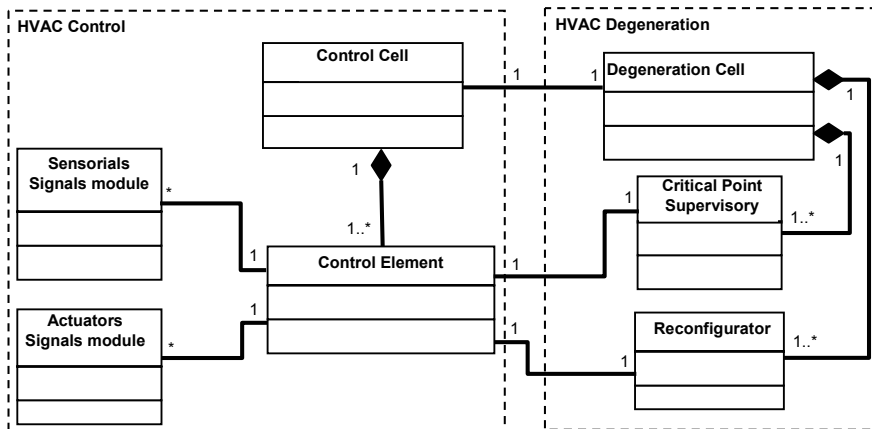


Figure 6 – HVAC with control component and degeneration component.

**Dynamic modeling:**

The dynamic modeling described here, are related with the critical points 1 and 2 presented in Table 1 (Step 1: identification of the critical points in the intelligent building systems). The model development for the other critical points is analogous.

Figure 7 illustrates a MFG (Mark Flow Graph) model (based on Petri Nets) (Hasegawa, 1998; Miyagi, 1988; Miyagi, 1996) that specifies a procedure to control the temperature, keeping it in a desired value. In this case, the desired temperature is function of sensors signals that detect the temperatures ('Cold', 'Hot') and the heating activity or the cooling activity (linked with the corresponding equipment).

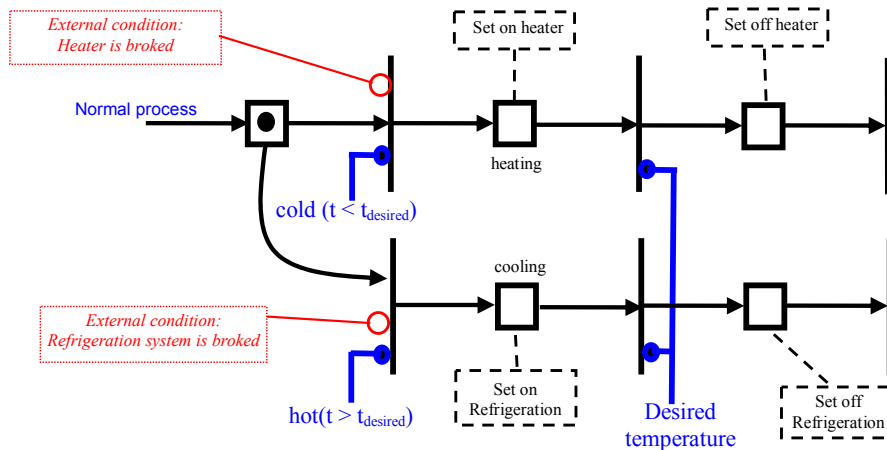


Figure 7. MFG model to keep the temperature in a desired value.

The Figure 8 presents the dynamics for the degeneration in HVAC subsystem, if an unexpected (refrigeration system broke) state happens, and then the fans are turned on until the environments are empty or the temperature reaches the

desired value, when the fans are turned off. Thus, the system continues in functioning even in the occurrence of unexpected faults.

### Step 3 – Technical specification of degeneration control software

The attributes and the services/operations for the degeneration components are detailed in this step. The technical specifications for the subsystems HVAC and Transport are presented: supervision of critical points components and reconfiguration components.

#### Component definition:

The control and degeneration components are similar for all the subsystems in analysis. The models of the degeneration components and control components for other subsystems of the intelligent building are similar to the models illustrated in figure 6.

#### Technical specification of critical point supervisory component

This component is associated with the control component. Thus, it accesses the same information offered for the 'sensor' components in the control component. In Table 2 the specifications for this component are presented (HVAC and Transportation).

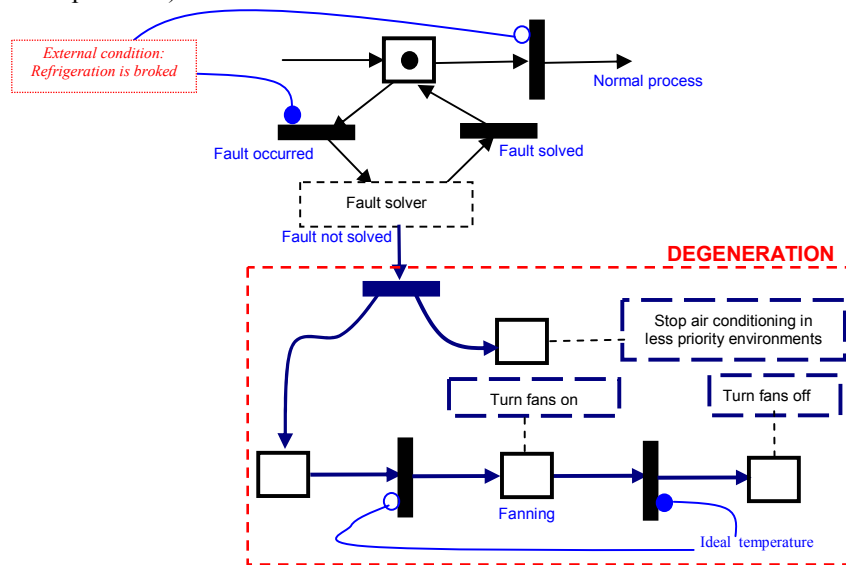


Figure 8. MFG model with degeneration in the refrigeration system.

Table 2. Technical specification of critical point supervisory.

Subsystem	Atributes	Services/Operations
HVAC	<p><i>_CriticalBombState</i>: <i>boolean</i> – indicate the necessity of the degeneration or not.</p> <p><i>_CriticalAirState</i>: <i>boolean</i> – indicate the necessity of the degeneration or not.</p>	<p><i>boolean getCriticalBombState()</i> – get the state of specific sensor for a critical point.</p> <p><i>boolean getCriticalAirState()</i> – get the state of specific sensor for a critical point.</p>
Transportation	<p><i>_CriticalEnergyTranspState</i>: <i>boolean</i> – indicate the necessity of the degeneration or not</p>	<p><i>boolean getCriticalEnergyTranspState()</i> – get the state of specific sensor for a critical point.</p>



Table 3. Technical specification of reconfiguration.

Subsystem	Attributes	Services/Operations
HVAC	<p><i>_idAirSegments[]</i>: integer – store the identifiers of air-conditioning segments.</p> <p><i>_idFans[]</i>: integer – store the identifiers of fans.</p>	<p><i>disableAirSegments(idSegmentsAr[]: integer)</i> – send signals to disable less priority segments.</p> <p><i>enableFans((idFans[]: integer)</i> – send signal to enable fans.</p>
transportation	<p><i>_idElevators[]</i>: integer – store the identifier of elevators.</p> <p><i>_idStairsRolling[]</i>: integer – store the identifiers of stairs rolling.</p> <p><i>_idStairsDoors[]</i>: integer – store the identifiers of stairs doors.</p>	<p><i>disableElevators(idElevators[]: integer)</i> – send signal to disable elevators.</p> <p><i>enableStairsDoors((idStairsDoors[]: integer)</i> – send signal to enable stairs doors.</p>

#### Technical specification of reconfiguration component

This component is associated with the control component. Thus, it sends information to the ‘actuator’ component in the control component. In this way, the corresponding degeneration is carried through activating the corresponding performance components (see Table 3).

## 5. CONCLUSIONS

Initially was presented the use of ISO/IEC 9126 standards in the control software design with degeneration requirements and the control system architecture for the discrete event systems with degeneration module included.

The degeneration method was presented through a sequence of tasks necessary to include the degeneration in productive systems. The results was the generation of software artifacts (static and dynamic models and technical specification of components) that will assist the software design for control system in intelligent building. Thus, the use of degeneration requirements in the proposed method permit to increase the system faults tolerance degree.

This paper received financial support from the following entities: FAPESP, CNPq and CAPES. The authors thank specially the Program TIDIA/Kyatera support.

## 6. REFERENCES

1. Arkin, H.; Paciuk, M. “Service System Integration in Intelligent Buildings”. Proc. of IB/IC Intelligent Buildings Congress. Tel-Aviv, Israel. pp. 19-30, 1995.
2. Arakaki, J. “Técnicas de Degeração no projeto do controle de Sistemas Produtivos”. Doctoral thesis, University of São Paulo. São Paulo-SP, 2004.
3. Bastidas Gustin, G. D.; et all. “Modelling integration of systems in intelligent buildings through mechatronics approach”. In: 8th Mechatronics Forum International Conference, 2002, Enschede. Mechatronics 2002 : proceedings. Enschede: Drebber Institute for Mechatronics, 2002.
4. Bastidas Gustin, G.D.; et all. “Open distributed supervisory system design using Petri nets”. In: IEEE International Symposium on Industrial Electronics, Rio de Janeiro, 2003. ISIE/2003. Piscataway : IEEE. v. 2, p. 712-717, 2003.
5. Flax, B. M. “Intelligent Buildings”. IEEE Communications Magazine, p. 24-27, 1991.

6. Hasegawa, K.; et all. "Enhanced production flow schema for modeling the complex sharing system". In: 3rd International Conference on Information Technology for Balanced Automation Systems in Manufacturing, Prague, 1998. Intelligent systems for manufacturing: multi-agent systems and virtual and organizations: proceedings of the BASYS'98. Boston : Kluwer Academic, p. 335-344, 1998.
7. ISO/IEC 9126. "Information Technology – Software Product Quality". 1998.
8. Miyagi, P. E.; Hasegawa, K.; Takahashi, K. "A Programming Language for Discrete Event Production Systems Based on Production Flow Schema and Mark Flow Graph". Transactions of The Society of Instrument and Control Engineers, Tokyo, v. 24, n. 2, p. 183-190, 1988.
9. Miyagi, P. E. "Controle Programável – fundamentos do controle de sistema a eventos discretos". Editora Edgard Blücher, São Paulo, 1996.
10. Miyagi, P. E.; et all. "Petri Net Approach for Modelling System Integration in Intelligent Buildings". Journal of The Brazilian Society of Mechanical Sciences, Rio de Janeiro, v. 23, n. 4, p. 341-350, 2002.
11. Villani, E.; Miyagi, P. E. "A Hybrid Petri Net Modeling Approach for HVAC Systems in Intelligent Buildings". Controle & Automação, a Journal of SBA, Campinas, v. 15, n. 2, p. 151-163, 2004.
12. Villani, E.; et all. "A Petri net-based object-oriented approach for the modelling of hybrid productive systems". Nonlinear Analysis, v. 62, n. 8, p. 1394-1418, 2005.