# Specification and Verification of the Model of Component and the Model of Function

**Nadia Hamani**[1], **Nathalie Dangoumau**[2], **and Etienne Craye**[2]

[1]Laboratoire Universitaire de Recherche en Production Automatisée (LURPA)

61, avenue du Président Wilson, 94235 Cachan cedex, France, nadia.hamani@lurpa.ens-cachan.fr

[2]Laboratoire d'Automatique, Génie Informatique et Signal (LAGIS),

Ecole Centrale de Lille, BP.48 59651 Villeneuve d'Ascq cedex, France

**Abstract:** Due to increasing complexity and competitiveness, automated production systems make, nowadays, higher demands on powerful techniques used to guarantee the requirements of performance and safety. Modeling production systems according to their operating modes allows obtaining a simplified view of these systems. The main difficulty is thus to insure mode changing while guaranteeing the system compatibility and coherence. The solution to this problem relate to modeling methods as well as verification tools. In this paper, we extend our modeling approach of *mode handling* by introducing adequate verification methods so that the design process will be carried correctly. The Model of Component (MoC) and the Model of Function (MoF) are the basic models of our modeling approach. Thus, we formalize some properties of the MoC and the MoF and we present the corresponding verification methods. We illustrate these methods through an application example. A computer aided tool for specification and verification is developed to illustrate our approach.

**Keywords:** Automated Production Systems, control system, supervision, mode handling, modeling, graph theory, verification.

**Dr. Nadia Hamani** received a Ph.D in Automatic control for Manufacturing and Discrete Events systems from Ecole Centrale de Lille, France, in 2005. She is currently in a post-doctoral position. Her research interests include supervision, monitoring, and control of discrete events systems.

**Dr. Nathalie Dangoumau** is an Associate Professor of Computer Sciences and Discrete Events Systems (D.E.S.) at the Ecole Centrale de Lille (France). She obtained a Ph.D. in Automatic control for Manufacturing and Discrete Events systems in 2000. Her research is focused on reconfiguration and modes management for Flexible Manufacturing Systems (FMS).

**Pr. Étienne Craye** was born in Roubaix (France) in 1961; he obtained in 1984 the Engineer Diploma of the "Institut Industriel du Nord" (French "Grande Ecole") and the same year his Master Degree in Computer Sciences. He obtained a Ph.D in Automatic control for Manufacturing and Discrete Events systems in 1989 and his "Habilitation à Diriger des Recherches" in 1994. Since 1995 he has been Professor at the Ecole Centrale de Lille and in the same time the Director of this institution. Pr. Craye is currently working on Monitoring and Supervision of Fault Tolerant Systems. Specially, reconfiguration and working mode management are today studied by taken into account on one-hand failures and on the other hand the flexibilities of the system architecture. The objective is to be able to go on with the production and not to reconsider the objectives.

## 1. Introduction

For the requirements of the Automated Production Systems (APS) control functions, it is necessary to know all operating modes and states of a system and its subsystems and also their changing conditions. However, due to increasing complexity of APS, some problems appear during state or mode changing if safety constraints are not taken into account in the specification/modeling stages.

Dependable modeling methods must be developed to deal with an increasing complexity and safety requirements. Several approaches are proposed in the literature [9][13]. These approaches focus on modeling, without dealing with the verification of the proposed models. However, these models should be verified at the early stages of the design process. A greater interest is allocated in our research team to verification methods, which guarantee that the developing models of APS respects some properties. That is why in this work we extend our modeling approach of *mode handling* [8][12] by developing formal verification tools so that the design process will be performed correctly.

We proposed in [8] a modeling method of the behavior of production systems from the point of view of their modes. The system is modeled using a Functional Graph (FG) of Exploitation obtained according to an analysis approach based on production goals. We should determine the main goals for which the system was designed and the sub goals that allow to reach them; each one of the sub goals can be decomposed in the same manner until obtaining the initial goals. These are the leaves of the graph; they are related to the resources which perform the initial goals. The behavior of the resources and the functions is specified in order to know constantly their states. For each function or resource, several concurrent families of modes are determined according to a multipoint of view method. The obtained models representing the behavior of the resources and the functions are respectively called Model of Component (MoC) and Model of Function (MoF). They are the basic models of this design approach.

As explained above, adequate verification methods should be developed for our modeling method. In this paper we introduce some properties of the MoC and the MoF and the corresponding verification methods using graph theory.

In the following section we present the main characteristics of our modeling approach for *mode handling* and the design process of the APS behavioral model. We extend this process by integrating appropriate verification methods. Section 3 presents some properties of the MoC and the MoF and the corresponding verification methods. In section 4, we illustrate our propositions through an application example. Finally, in section 5, we discuss some future prospects.

## 2. A Modeling Approach for Mode Handling

The modeling methods and the specification formalisms which characterize the existing approaches are compared in [13]. The main conclusions of this comparative study show that a functional modeling approach [17] is very convenient for *mode handling*. Indeed, functional approaches [6][7][16] are well suited for efficient reconfiguration procedures instead of structural approaches [4][15]. In structural approaches, the decomposition is based on structural relationships between the resources of the system. Reconfiguration actions are taken on the resources, the workstations and the cells. Functional approaches are concerned with the services delivered by the system rather than the resources. The decomposition is based on functional relationships between subsystems. Such relationships enable the implementation of automated reconfiguration procedures. Our modeling method [8] is based on a functional decomposition. The specification of the functional subsystems and their behavior is presented in the following**.**

According to [8], our modeling method of an APS for *mode handling* is carried out in two steps as shown in Fig. 1:

1) Building the hierarchical structure of the behavioral model starting from the structure of the FG [18] and the production goals;
2) Specification of the behavior of the subsystems: the behavior of each function and each component identified in the previous step is specified.

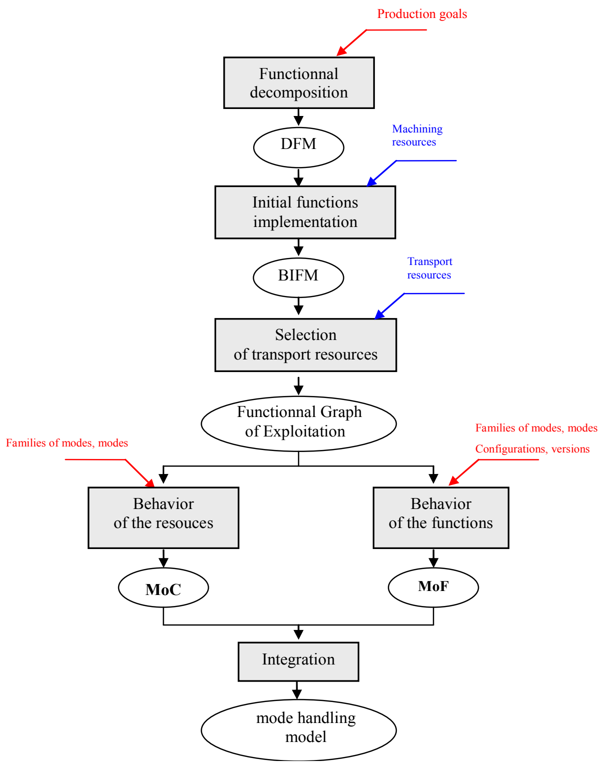


**Figure1.** The design process of *mode handling* model.

Before describing these two steps in the following paragraphs, we remind that the FG [18] is a directed graph whose nodes represent the functions and the edges represent the functional dependencies. The FG represents the causality of the system through functions. Thus, the FG is used for failure diagnosis within the monitoring function of our control system [18].

Two intermediary models are used to build the FG: the Design Functional Model (DFM) and the Basic Implementation Functional Model (BIFM). Based on the information provided in the functional requirements, the DFM represents the main functions of the system. The designer defines then some levels of intermediate functions before the identification of initial functions. The determination of the BIFM consists in specifying the initial functions taking into account the selected resources. The initial functions of the DFM are implemented by the operations which are performed by the resources. The FG is obtained starting from the BIFM when the complete selection of machining and transport resources is established.

These stages of building the FG are used in [8] to determine the hierarchical structure of the model dedicated to *mode handling*. In the functional decomposition (see Fig. 1), the production goals are taken into account for the determination of the functions. These goals are obtained from the control models [10]. In the following we present the process of obtaining the model dedicated to *mode handling*.

## 2.1 Building the hierarchical structure

The modeling method proposed in [8] is a top-down approach based on production goals. The proposed model consists in defining and characterizing the system from a functional point of view (according to its goals). The main goals correspond to the goals for which the system was designed. It is necessary also to characterize the sub goals. Each one of the sub goals can also be decomposed (Fig. 2).

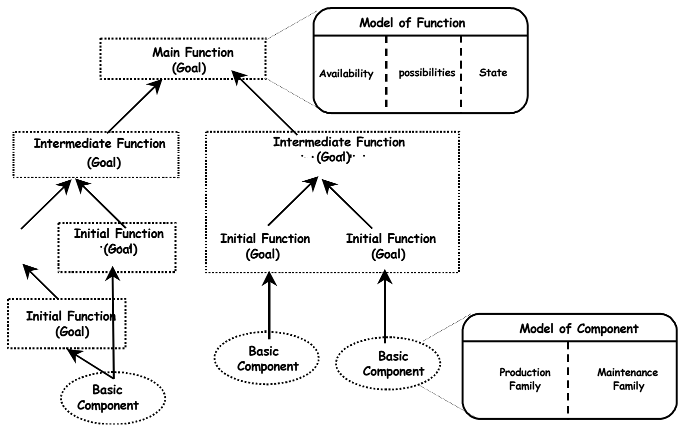


**Figure 2.** The model dedicated to *mode handling* [8].

The arrows represented on Fig. 2 between each intermediate level represent the existing causality links among the various levels. The source of each arrow corresponds to an element necessary to obtain that of the higher level. The states of initial goals (at the lower level) depend on the elementary components which are necessary to achieve them. It is then necessary to determine the state of each elementary component in order to determine the state of the initial goals.

Each goal (or function) is then like a system or an autonomous subsystem corresponding to a given abstraction level. The characterization of this level allows determining the state of the systems or the subsystems of higher and lower level. Each level is associated with a MoF (Fig. 2) and each elementary component is represented by a MoC.

## 2.2 Specification of the behavior of the subsystems

Each function and each elementary component is respectively associated with a MoF and a MoC. The behavior represented by these models is characterized by a set of concurrent state-transition graphs (the set is called a family of modes and the graphs are called modes [8]). Each state of a graph characterizes the production system or a subsystem at a given moment. The transitions allow changing a state to another. These various modes represent the states of the same component; there are consequently relations between them. These relations (also called constraints) can lead to incoherent situations. To avoid such situations, it is necessary to know the incompatible states. The incompatibilities and constraints are taken into account in the design process by the addition of some specifications (called mechanisms) to respect them.

*1) Specification of the modes:* This specification process is based on the point of view concept, which allows characterizing a resource according to the observer's criteria (Fig. 3). For example from the exploitation point of view, the system is characterized according to two points of view: production and maintenance. These points of view can be characterized by other points of view. Inspired from the *GEMMA*[1] [1], the method determines a set of families of modes and generic modes representing the behavior of a production system [8].

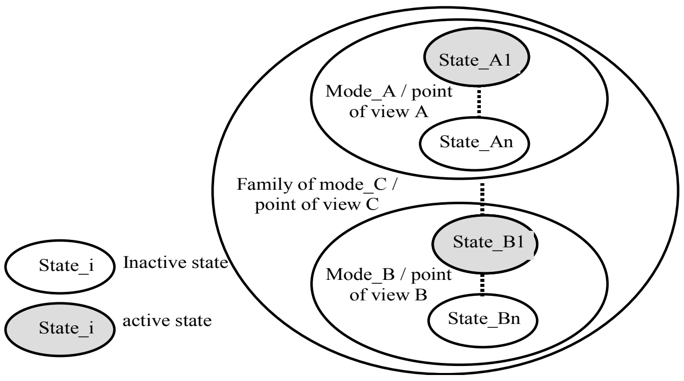


**Figure 3**. Partition of the states according to the point of view approach [8].

*2) Specification of the Models of Component:* The specification of the MoC (Fig. 4) follows three steps:

***1st step- Specification of the static part:*** This step consists in listing the states that can take the system. The modes which characterize an elementary component are the same ones as those which characterize a production system.

***2nd step- Specification of the dynamic part:*** This step consists in determining at first the initial states of the model then the change-of-state conditions. A matrix form called Matrix of the Change-of-state Conditions (MCC) is used to represent all the possibilities within a mode.

***3rd step- Study of the coherence of the modes:*** This study begins with the development of the Matrix of Coherence of the Modes (MCM) (also called Matrix of Compatibility) which characterizes the incompatibilities of the states. When two states are incompatible their simultaneous activation is not possible and this should be taken into account in the specifications. We distinguish forbidden states and transient states. The forbidden states correspond to situations that should not occur. So any switching to these states must be forbidden. The transient states are states from which we can only go through without staying (the switching is considered as instantaneous). The activation of such states causes then a switching to compatible states. In order to guarantee the coherence of the MoC, a specification solution called mechanism is proposed for each case (see subsection 3.2.2). Then it is necessary to report the specifications resulting from the mechanisms on the final MoC.

Following the previous steps, the final MoC is obtained. The example represented in Fig. 4 shows two families of modes: production and maintenance. The family of production modes includes:

- Shutdown mode (PS): turned off (PS-to), initial (PS-i);
- Working mode (PW): automatic (PW-a);
- Functioning mode (PF): normal (PF-n), degraded (PF-d);

[1] *GEMMA* (the French acronym of Guide d'Etude des Modes de Marches et d'Arrêts) is a method of operating modes specification.

- Production mode (PP): preparation (PP-p), effective (PP-e).

A meaningless state is represented in each mode.

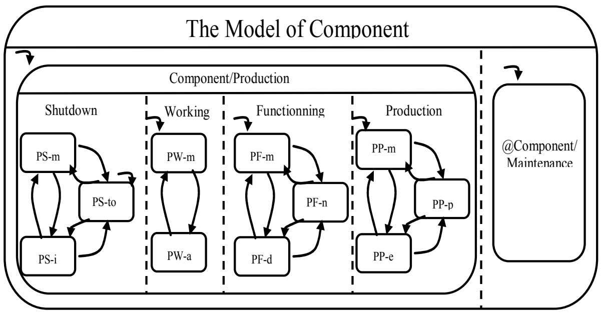


**Figure 4.** Example of a Model of Component [8].

*3) Specification of the Models of Function:* A function is characterized by its availability, its using context (the configurations and their versions) and the states (behavior) of the corresponding subsystem according to its modes. The states of the subsystem are obtained in the same manner as for the MoC. Fig. 5 shows an example of a MoF.

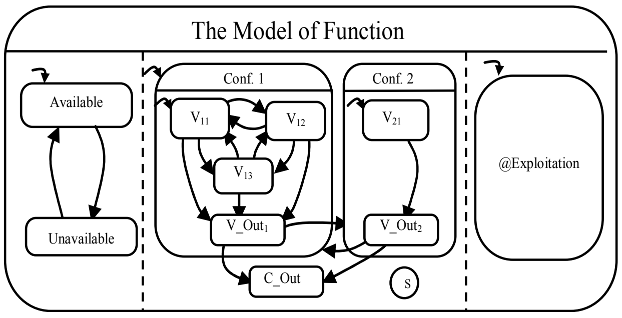


**Figure 5.** Example of a Model of Function [8].

For clarity reasons, the change-of-state conditions are not represented in Fig. 4 and Fig. 5.

*4) Integration rules*: Some rules are defined in order to handle the interactions between the models designed in the previous stages. These rules depend on the kind of the relationships that characterize the functional modeling (necessity, alternative), the considered point of view (production, maintenance) and the current context (configuration and version) [8].

## 3. From Specification to Verification

The specification of the behavior of the resources and the functional subsystems is carried out using the MoC and the MoF. The following concepts are used to represent both the MoC and the MoF [8]:

- The static part: families of modes, modes, states (meaning state, meaningless state);
- The dynamic part: initial state, change-of-state conditions, events;
- Mode coherence: incompatibility, switch and forbidden mechanisms.

We use the mathematical model of state-transition graphs to formalize the MoC (resp. the MoF). The specification and the corresponding verification process are presented in the following.

### 3.1 Specification of the modes using state-transition graphs

As shown the MoC (resp. MoF) represent the behavior of an elementary component (resp. a functional subsystem) from the point of view of its modes. The specified modes can be represented using directed state-transition graphs (Fig. 6). The states correspond to the tops of the graph and the transitions correspond to the edges. Each transition is labeled and the label corresponds to change-of-state condition.

Therefore, handling the modes and consequently the states of an entity is carried out using labeled transition systems, each one represents a mode.

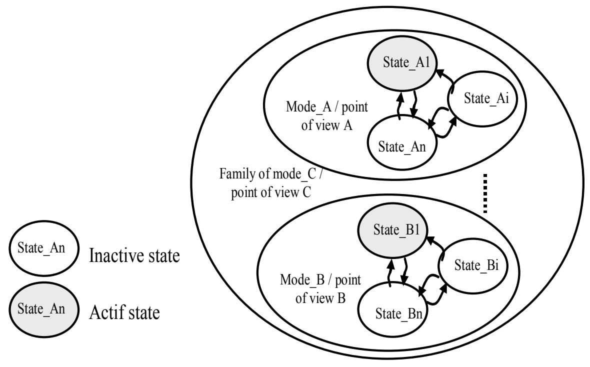


**Figure 6.** Specification of the modes using state-transition graphs.

***Definition 1.*** The states of an entity (or a system) in a mode are represented by a directed state-transition graph. Each top of the graph represents a state in a mode and the label corresponds to the change-of-state condition.

The formal definition of a labeled state-transition system is given below [3].

***Definition 2.*** A labeled state-transition system on an alphabet A is a quintuplet [S, T, $\alpha$, $\beta$, c] where:

- [S, T, $\alpha$, $\beta$, c] is a directed graph,
- S is a set of states (instead of nodes),
- T is a set of transitions (instead of edges),
- $\alpha$ and $\beta$ are two applications of T in S that to each edge, associate its origin $\alpha(t)$ and its goal $\beta$ (t).
- c is an application of T in A: c(t) is called the label of the transition t.

In our model, the alphabet A corresponds to the combinations of events that cause mode changing of an entity.

The change-of-state conditions in a mode are listed in a Boolean matrix known as incidence matrix of the graph [11]. There are as many matrices as modes taken into account.

The definition given below presents some concepts related to graph theory [2][11]. We need these concepts later.

***Definition 3.*** We call incidence matrix of a graph G = [S, T, $\alpha$, $\beta$, c] the matrix W(G) such as:

$$W(G) = w_{ij} \text{ with } \begin{cases} w_{ij} = 1 & \text{if } (i, j) \in T \\ w_{ij} = 0 & \text{otherwise} \end{cases}$$

$(i, j) \in T$ corresponds to the directed transition (of top i towards the top j).
For verification, we use the calculations below related to the incidence matrix:

- The out-degree of top i, noted $d^+$ (i) is the arithmetic sum of the edges number outgoing of top i.
- The in-degree of top i, noted $d^-$ (i) is the arithmetic sum of the edges number entering the top i.
- A graph transitive closure is represented using the matrix:

$$\hat{W}(G) = W^{[0]} + W^{[1]} + W^{[2]} + \ldots + W^{[k]} \ldots, \text{ where } \begin{cases} W^{[0]} = I \text{ (Boolean matrix unit)} \\ W^{[1]} = W \end{cases}$$

With k > n-1 (n is the order of the incidence matrix W). In practice, we stop as soon as: $W^{[k+1]} = W^{[k]}$.
We use a practical calculation of transitive closure with the Boolean theorem of the binominal [2], which

is written as follows: $[I + W]^{[k]} = I + W + W^{[2]} + \dots + W^{[k]}$

The terms to be calculated are:

$[I + W]^{[2]}$, $[I + W]^{[4]}$… until: $[I+W]^{2^{P+1}} = [I+W]^{2^{P}}$

- Some properties need to use change-of-state conditions in the incidence matrix (Table 1). In this case, we will have:

$$w_{ij} = \begin{cases} c_{ij} & \text{si } (i, j) \in T \end{cases}$$

(i, j) $\in$ T corresponds to the directed transition (of top i towards the top j).

**Table 1.** The matrix of change-of-state conditions

| States | $e_1$ | $e_j$ | $e_n$ |
|---|---|---|---|
| $e_1$ | | | |
| | | | |
| $e_i$ | | 0 or $c_{ei,ej}$ | |
| | | | |
| | | | |
| $e_n$ | | | |

The constraints between the modes belonging to the same family are represented using a matrix form as shown in the following.

**Table 2.** Matrix of Coherence of the Modes

| Modes | | S | | | | W | | F | | | | P | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | States | PS-m | PS-to | PS-i | PS-d | PW-m | PW-a | PF-m | PF-n | PF-d | PF-o | PP-m | PP-p | PP-e |
| S | PS-m | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| | PS-to | | 1 | 0 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| | PS-i | | | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| | PS-d | | | | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 |
| W | PW-m | | | | | 1 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 |
| | PW-a | | | | | | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| F | PF-m | | | | | | | 1 | 0 | 0 | 0 | 1 | 0 | 0 |
| | PF-n | | | | | | | | 1 | 0 | 0 | 0 | 1 | 1 |
| | PF-d | | | | | | | | | 1 | 0 | 0 | 1 | 1 |
| | PF-o | | | | | | | | | | 1 | 0 | 1 | 1 |
| P | PP-m | | | | | | | | | | | 1 | 0 | 0 |
| | PP-p | | | | | | | | | | | | 1 | 0 |
| | PP-e | | | | | | | | | | | | | 1 |

***Definition 4***. The constraints are taken into account through compatibility relations between states. They are specified in the MCM [8] in which the lines and the columns represent the states. $\mathrm{MCM}\binom{m_i, m_j}{e_{ik}, e_{jl}}$ represents the compatibility of the states $e_{ik}$ and $e_{jl}$ (with $e_{ik} \in m_i, e_{jl} \in m_j$). This matrix (Table 2) is built as follows :

$\mathrm{MCM}\binom{m_i, m_j}{e_{ik}, e_{jl}} = 1$ if the states $e_{ik}$ and $e_{jl}$ are compatible,

$\mathrm{MCM}\binom{m_i, m_j}{e_{ik}, e_{jl}} = 0$ if not.

For example, the states PS-m and PW-a belonging respectively to PS and PW modes, are compatible whereas the states PS-m and PF-m belonging respectively to PS and PF modes are incompatible.

## 3.2 Properties and verification

The specification of the behavior of a MoC or a MoF is based on the 'point of view' concept [8]. Thus, the identified families of modes and modes for an entity depend on the considered points of view. A top-down specification is used. We start with the identification of the families of modes, the modes and the states. The verification is a bottom-up process as explained in the following.

The state-transition graphs representing the modes belonging to the same family (definition 1) are constrained. Thus, the expression of properties and their verifications are carried out in three steps.

1) We consider first the state-transition graphs that represent each mode belonging to the same family: some properties being verified are introduced and the corresponding verification methods are presented. This stage concerns the state-transition graphs specified for each mode and taken independently from each other.
2) After considering the modes separately, we introduce in a second stage the properties related to the modes belonging to the same family.
3) A MoC (resp. a MoF) is characterized by a set of families of modes. It is assumed that the families of modes are not constrained so we introduce the properties related to these models.

*1) The modes:* In this stage, we verify the structural properties of the state-transition graph representing a mode. The verification of these properties uses mainly the incidence matrix of the graph defined above (definition 3). Some properties will be presented in the following.

***Property 1.*** *Deadlock-freeness*

Every state within a mode must be deadlock-free.

***Verification:*** We have to verify that whatever the top i of the graph, its out-degree $d^+(i)$ is not null.

$$\forall G = [S, T, \alpha, \beta, c], \forall i \in S, d^+(i) \neq 0$$

☞ In the incidence matrix, for a given top, the corresponding line should not contain only zeros. So there is at least an edge whose initial extremity is the top i.

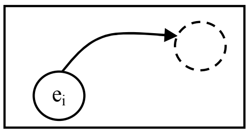


**Figure 7.** Deadlock-free state $e_i$.

**Property 2.1.** *Reachability*

Every state within a mode must be reachable.

**Verification:** We have to verify that whatever the top i of the graph, its in-degree $d^-(i)$ is not null.

$$\forall G = [S, T, \alpha, \beta, c], \forall i \in S, d^-(i) \neq 0$$

☞ In the incidence matrix, for a given top, the corresponding column should not contain only zeros. So there is at least an edge whose final extremity is the top i.

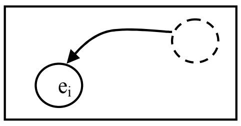


**Figure 8.** Reachable state $e_i$.

**Property 2.2.** *Mutual reachability*

Every state within a mode must be reachable starting from any other state in the same mode.

**Verification:** We have to verify that the graph is strongly connected by calculating its transitive closure.

$\forall G = [S, T, \alpha, \beta, c]$, if there is a path between any pair of distinct tops of G, the graph is strongly related.

☞ All the elements of the transitive closure matrix of a strongly connected graph are equal to 1.

**Note:** the previous property is not mandatory in all design cases [12].

***Property 3.*** *Determinism*

Every mode must be deterministic.

***Verification:*** We have to verify that for any top i of which the degree is strictly higher than 1, the labels (the logical expressions) of the transitions cannot be simultaneously true. It is supposed here that two uncorrelated events cannot be simultaneous.

For verification, we use change-of-state conditions according to the incidence matrix (definition 3).

☞ In this matrix, for the transitions whose top is i, the change-of-state conditions should not be simultaneously true.

*2) The families of modes:* The structural properties listed above have to be checked for each mode. The incompatibilities of the modes belonging to the same family are taken into account within the specification of the state-transition graphs (the change-of-state conditions). We consider the modes within their family and we propose the corresponding properties. At first we introduce the following definition.

***Definition 5.*** At a given moment, the states, which do not belong to the same mode, are simultaneously active in a family of modes.

This is guaranteed by the principles of modeling: the concurrency of the modes belonging to the same family and the presence of an initial state for each mode.

The following property adds a condition on this definition.

***Property 4.*** *Compatibility*

Each state should be simultaneously active at least with another state of each one of the other modes in the same family.

***Verification:*** We have to verify that any state of any mode is compatible with at least one state of the other modes of the same family.

$\forall\, e_{ki} \in m_k$, $\forall\, l$, $\exists\, j$ such as $e_{lj} \in M_l\ (k \neq l)$ is compatible with $e_{ki}$ where $e_{ki}$ and $e_{lj}$ are two states of two distinct modes $m_k$ and $m_l\ (k \neq l)$ respectively and belonging to the same family.

☞ In the MCM, for two distinct modes, the line corresponding to each top i should not contain only zeros.

Now we consider the constraints that exist between the states of the modes belonging to the same family in the MoC (resp. the MoF). The constraints are represented by the incompatibilities, which are taken into account in the specification stage. The incompatibilities are provided by the designer and specified in the MCM (definition 4). Thus, we need the list of incompatible states as well as the kind of the mechanisms used to specify the incompatibilities (forbidden and switch mechanisms).

Given two incompatible states $A_1$ and $B_1$; when using the mechanisms, three solutions can be used to forbid the simultaneous activation of these two states [8]:

1) When the forbidden mechanism is used, the addition of constraints '*and not $A_1$*' and '*and not $B_1$*' do not enable switching to $A_1$ (respectively to $B_1$) if the state $B_1$ (respectively $A_1$) is already active (Fig. 9).
2) When the switching mechanism is used, if the state $B_1$ (respectively $A_1$) is active, the condition $a_{21}$ (respectively $b_{21}$) which allows the activation of the state $A_1$ (respectively $B_1$) provokes a change-of-state to $B_2$ (respectively $A_2$) thanks to '*/$i_2$*' (respectively '*/$i_1$*') (Fig. 10).
3) The forbidden and switching mechanisms are used, which is a combination of the two previous cases (Fig. 11).

Let us note here that the choice of these three solutions is the task of the designer.

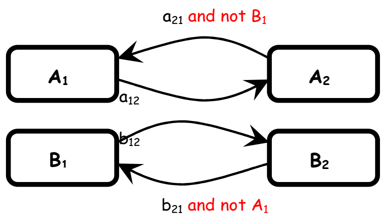


**Figure 9.** The forbidden mechanism [8].

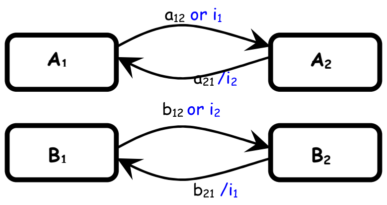


**Figure 10.** The switch mechanism [8].

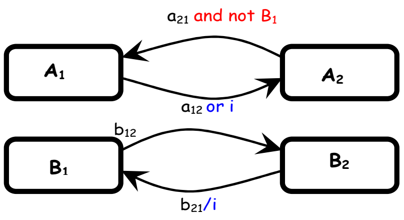


**Figure 11.** The forbidden and switch mechanisms [8].

These solutions are illustrated on the example of two constrained modes. The method implementing these partial specifications on the set of concurrent modes of the same family is detailed in [8]. It is based on the asynchronous product of the graphs representing the modes (inspired from [5]). Then it is necessary to use forbidden or switch mechanisms for all the transitions entering the incompatible composed states. After dissociation, new change-of-state conditions are obtained.

The properties being verified on the MoC (resp. le MoF) depend on the kind of the mechanism, more precisely on the change-of-state conditions. Thus, for each state of each incompatible states pair, a forbidden or a switching solution is implemented.

The following properties concern these two kinds of mechanisms. To this aim, we look at the change-of-state conditions of the transitions entering at and/or outgoing of these states according to the mechanism. The properties correspond to the specifications of Fig. 11. We chose this case because it is specified with both solutions.

***Property 5.*** *Reachability - forbidden mechanism*

When the forbidden mechanism is used, if one of the states is active the other state remains unreachable (and reciprocally if the forbidden solution is used in the two directions).

***Implementation*:** $A_1$ and $B_1$ are two incompatible states, the designer chooses to forbid the activation of $A_1$ when $B_1$ is already active (a forbidden mechanism). Thus, if $B_1$ is active, the activation condition of $A_1$ '$a_{21}$ and not $B_1$' is invalid. If the forbidden solution is used for the two cases (Fig. 9), we will see that the activation conditions of $A_1$ and $B_1$ respectively '$a_{21}$ and not $B_1$' and '$b_{21}$ and not $A_1$' are never valid simultaneously ($a_{21}$ and $b_{21}$ are uncorrelated so never simultaneously true).

***Verification*:** We have to verify that for any forbidden states pair, if one state is active, the change-of-state conditions associated with the entering transitions of the other state are invalid (and reciprocally).

☞ In the incidence matrix, if one of the two forbidden states is active, the column corresponding to the other state should include only transitions that are invalid (and reciprocally if the forbidden mechanism is used in the two directions).

***Property 6.*** *Reachability-switching mechanism*

When the switching mechanism is used, if one of the states is active, the activation of the other state provokes instantaneously a switching of the first state to a compatible state (and reciprocally if switching is caused in the two directions).

***Implementation:*** For the incompatible states $A_1$ and $B_1$, the designer chooses a switching mechanism: when $A_1$ is active, the activation of $B_1$ is allowed but it provokes instantaneously switching $A_1$ to a compatible state with $B_1$, which is $A_2$. Thus, if the activation condition of $B_1$ '$b_{21}$ / i' is true, the one which allows the change-of-state of $A_1$ (i.e. '$a_{12}$ or i') is also true. $A_1$ and $B_1$ are called transient states.

***Verification*:** We have to verify that at least one of the outgoing change-of-state conditions from the supposed active state becomes instantaneously valid at the moment of the activation of the other state.

☞ In the incidence matrix, if one of the two transient states is active, the column corresponding to the other state must include a transition, which instantaneously provokes the switching of the state presumed active. The line corresponding to this last state contains a transition, which instantaneously becomes valid and switches the state to a new compatible one.

*3) The MoC (resp. the MoF):* According to the point of view concept, a MoC (resp. MoF) is characterized by a set of families of modes. We assume that there is no constraint between those families. Thus the integration of the families of modes in the MoC (resp. the MoF) does not require the addition of particular specifications. As a result the properties verified on the families of modes are unchanged.

***Remark.*** In the same manner, the proposed analysis and verification process can be applied to the specifications related to the alternatives and availability parts of the MoF. Indeed, the properties related to the graphs representing the modes (subsection 3.2.1) remain valid for the graphs representing the alternatives and the availability.

We summarize in the following paragraph the verification process of the MoC (resp. the MoF). These models fulfilling some properties can then be integrated for the design of the mode handler (cf. subsection 2.2.4).

## 3.3 The analysis and verification process

Fig. 12 summarizes the process detailed in the previous paragraph. Thus, upwards we start with representing each mode determined in the specification stage. The constrained modes are then organized in families of modes. The constraints are taken into account through the incompatibilities between states listed in the MCM. The families of modes are then integrated to obtain the MoC (resp. MoF).

At this stage, we should verify that the graphs representing the modes are correctly specified (i.e. properties of the modes). Then it is necessary to check the coherence of each family of modes (i.e. properties related to the coherence of the families of modes).

If one of the properties related to the modes or the families of modes is not verified it is necessary to reconsider the method of implementing the incompatibilities or the earlier steps i.e. the structural and functional decomposition of the APS and the determination of the modes.

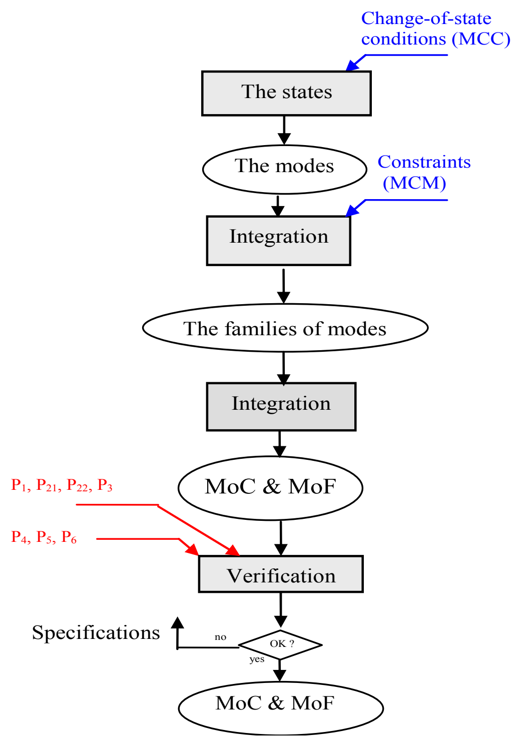


**Figure 12.** Modeling and verification of the MoC and the MoF

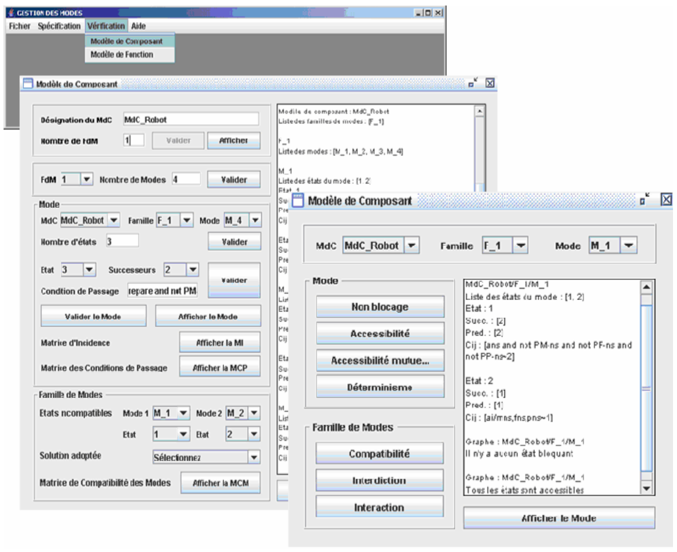
**Table 3.** The properties of the MoC (resp. of MoF)

| Properties | | Designation | Correspondence with the properties of state-transition graphs |
|---|---|---|---|
| Modes | $P_1$ | deadlock freeness | the out-degree $d^+(i)$ of each top i of the graph is not null |
| | $P_{21}$ | reachability | the in-degree $d^-$ (i) of each top i of the graph is not null |
| | $P_{22}$ | mutual reachability | the graph is strongly connected |
| | $P_3$ | determinism | for each top of the graph which has more than one outgoing edge, the labels of the transitions cannot be simultaneously true |
| Families of modes | $P_4$ | compatibility | in the MCM, for two distinct modes, the line corresponding to each top i should not contain only zeros |
| | $P_5$ | Reachability-forbidden mechanism | if one of the states is active, the other state is unreachable |
| | $P_6$ | Reachability-switching mechanism | if one of the states is active, it should be switched from the moment of the activation of the other state |

The obtained MoC (resp. MoF) respect the required properties and can be re-used for the design of the mode handler. This verification process allows an early correction of specification errors; otherwise they lead to a considerable cost of correction of the final model.

Table 3 presents the proposed properties. We give their significance and the corresponding mathematical properties of state-transition graphs.

Building a MoC (resp. a MoF) using our modeling method seems not trivial. We need to define many state-transition graphs and the obtained model may be quite complicated. Within the modeling process, the properties presented in this paper should also be respected. That is why an assistance tool implementing all the steps from specification to verification is developed using Java (with *Jbuilder* 9) [14]. Fig 13 represents the main user interfaces of the tool.



Désignation du MdC: MdC_Robot
Nombre de FdM: 1
FdM: 1 Nombre de Modes: 4

Mode
MdC: MdC_Robot Famille: F_1 Mode: M_4
Nombre d'états: 3
Etat: 3 Successeurs: 2
Condition de Passage: repare and not PM
Matrice d'Incidence
Matrice des Conditions de Passage

Famille de Modes
Etats incompatibles Mode 1: M_1 Mode 2: M_2
Etat: 1 Etat: 2
Solution adoptée: Sélectionnez
Matrice de Compatibilité des Modes

```
Modèle de composant : MdC_Robot
Liste des familles de modes : [F_1]

F_1
Liste des modes : [M_1, M_2, M_3, M_4]

M_1
Liste des états du mode : [1, 2]
```

MdC: MdC_Robot Famille: F_1 Mode: M_1

```
MdC_Robot/F_1/M_1
Liste des états du mode : [1, 2]
Etat : 1
Succ. : [2]
Pred. : [2]
Cij : [ans and not PM-ns and not PF-ns and not PP-ns~2]

Etat : 2
Succ. : [1]
Pred. : [1]
Cij : [ai/mns,fns,pns~1]

Graphe : MdC_Robot/F_1/M_1
Il n'y a aucun état bloquant

Graphe : MdC_Robot/F_1/M_1
Tous les états sont accessibles
```

**Figure 13.** User interface of the design aided tool.

The application of this method to a MoC of a robot taken as an illustration example is presented in the following paragraph. The notation proposed in [8] is used for representing the states, the modes and the families of modes.

## 4. Illustration

Consider a loading/unloading robot of a flexible manufacturing cell (Fig. 14) to illustrate our approach. The MoC of this example is represented in Fig. 15. It has two families of modes: the Production family and the Maintenance family; only the first family is represented. It includes four modes, the Shutdown mode (PS), the Working mode (PW), the Production mode (PP) and the Functioning mode (PF). Let us take the Functioning mode example that has three states: the meaningless state (PF-m), the normal state (PF-n) and the out of order state (PF-o).

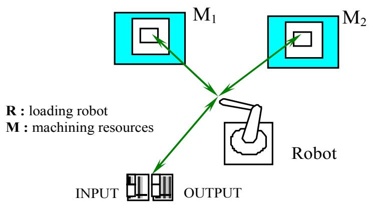


**Figure 14.** Illustration example.

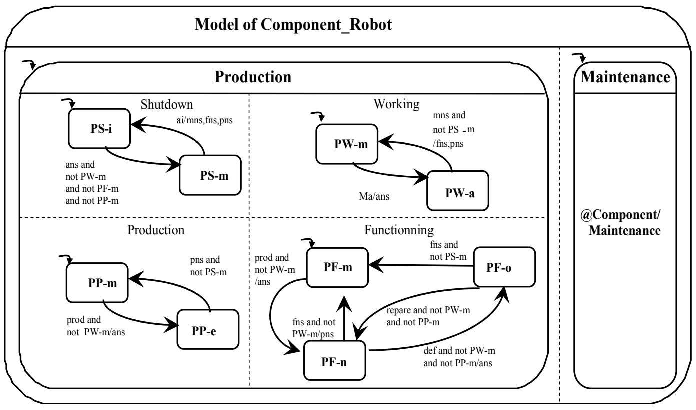


**Figure 15.** The production family modes of the MoC_Robot.

Taken the modes separately, only the verifications made for PF mode are presented. The other modes of MoC are verified in the same manner. We present then the verifications related to the modes belonging to the same family.

- The incidence matrix associated with PF mode is given in the table below:

**Table 4.** PF mode incidence matrix

| States | PF-m | PF-n | PF-o |
|---|---|---|---|
| PF-m | 0 | 1 | 0 |
| PF-n | 1 | 0 | 1 |
| PF-o | 1 | 1 | 0 |

- Let us calculate the out-degrees and the in-degrees of all the states:

  For the out-degrees, we find: $d^+$(PF-m)=1, $d^+$(PF-n)=2, $d^+$(PF-o)=2.

  For the in-degrees, we find: $d^-$ (PF-m)= 2, $d^-$ (PF-n)=2, $d^-$ (PF-o)=1.

- The transitive closure matrix of the graph is obtained using the binomial theorem [2]:

We calculate $[I+M]^{[2]}$ and $[I+M]^{[4]}$ and we stop because $[I+M]^{[4]} = [I+M]^{[2]}$ = matrix with ones (1) everywhere. We obtain:

$$\hat{M} = I + M + M^{[2]} = [I + M]^{[2]} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

- The incidence matrix of PF mode is completed with the change-of-state conditions, we obtain the following matrix:

**Table 5.** Change-of-state conditions of PF mode

| States | PF-m | PF-n | PF-o |
|---|---|---|---|
| PF-m | 0 | prod and not PW-m/ans | 0 |
| PF-n | fns and not PW-m | 0 | def and not PW-m and not PP-m/ans |
| PF-o | fns and not PS-m | repare and not PW-m and not PP-m/ans | 0 |

- Table 8 represents the MCM of the MoC_Robot, which we use to verify property 4. Properties 5 and 6 use the incidence matrices completed with change-of-state conditions.

For PF mode, ***properties 1, 2.1*** and ***2.2*** are proved according to the previous calculations: the out-degrees as well as the in-degrees are not null. Moreover, all the elements of the transitive closure matrix are equal to 1.

***Property 3*** is proved because if there is more than one change-of-state condition in the line corresponding to any state, they are never simultaneously true. For example, the transitions that start from PF-o, have the change-of-state conditions '*fns and not PS-m*' and '*repare and not PW-m and not PP-ms*' that are not true at the same time because the events '*fns*' and '*repare*' are not correlated.

***Property 4*** is proved because for two distinct modes, the line corresponding to each top does not contain only zeros.

According to MCM (Table 8) the MoC of the robot has 12 incompatible state pairs. The designer gives specification solutions (to forbid and/or to switch). Let us take an example; the incompatible states pair (PW-m, PP-e). To respect this incompatibility in the specifications, the designer chooses both forbidding (see the change-of-state condition between PP-m and PP-e: '*prod and not PW-m/ans*') and switching (see the change-of-state condition between PW-a and PW-m: '*mns and not PS-m/fns, pns*') mechanisms.

For PW mode, ***property 5*** is proved because if PW-m is active the condition '*prod and not PW-m/ans*' which allows PP-e activation in PP mode is invalid. PP-e is unreachable as long as PW-m is active because it is conditioned by '*not PW-m*'.

**Table 6.** Change-of-state conditions of PP mode

| States | PP-m | PP-e |
|---|---|---|
| PP-m | 0 | prod and not PW-m/ans |
| PP-e | pns and not PS-m | 0 |

For PP mode, ***property 6*** is proved because if PP-e is active, the condition '*mns and not PS-m/fns,pns*' which allows PW-m activation in PW mode causes instantaneously PP-e switching to PP-m (Table 6). So, if the condition, which enables PW-m activation, is true, the one that switches PP-e (i.e. '*pns and not PS-m*') to PP-m is also true. PP-m is compatible with PW-m. PW-m and PP-e are called transient states.

**Table 7.** Change-of-state conditions of PW mode

| States | PW-m | PW-a |
|---|---|---|
| PW-m | 0 | Ma/ans |
| PW-a | mns and not PS-m/fns, pns | 0 |

Thus, all the properties are proved for the production modes family of the MoC_Robot. Note that if one of the properties is not proved it is necessary to reconsider the specification stage as needed.

## 5. Conclusion

For safety requirements of APS *mode handling*, the models must be provided with adequate verification methods and tools. In this paper, we extend the modeling approach proposed in [8] by introducing adequate verification methods. We proposed some properties for the specification of the MoC and the MoF as well as the corresponding verification methods so that the design process will be carried out correctly. The properties are independent of the context and should be respected whatever the modeled system is. The approach is based on graph theory. We illustrated it through an example of a loading/unloading robot. A computer aided tool was developed to help the designer for the specification and the verification of the models dedicated to *mode handling*.

This study will be continued to verify other properties within the design process. The verification approach can be extended to deal with the model representing the behavior of the APS.

**Table 8.** MCM of the Production Family of MoC_Robot

| Modes | | S | | W | | F | | | P | |
|---|---|---|---|---|---|---|---|---|---|---|
| | States | PS-m | PS-i | PW-m | PW-a | PF-m | PF-n | PF-o | PP-m | PP-e |
| S | PS-m | 1 | 0 | 0 | 1 | 0 | 1 | 1 | 0 | 1 |
| | PS-i | | 1 | 1 | 1 | 1 | 0 | 0 | 1 | 0 |
| W | PW-m | | | 1 | 0 | 1 | 0 | 0 | 1 | 0 |
| | PW-a | | | | 1 | 1 | 1 | 1 | 1 | 1 |
| F | PF-m | | | | | 1 | 0 | 0 | 1 | 0 |
| | PF-n | | | | | | 1 | 0 | 0 | 1 |
| | PF-o | | | | | | | 1 | 0 | 1 |
| P | PP-m | | | | | | | | 1 | 0 |
| | PP-e | | | | | | | | | 1 |

## REFERENCES

1. ADEPA, Le GEMMA, **Guide d'Etude des Modes de Marches et d'Arrêts**, Production Automatisée, ADEPA, 1981.
2. ALJ, A. and FAURE R., **Eléments de la théorie des graphes, Guide de la recherche opérationnelle**, T.1. Les fondements, MASSON (France), 1986. Ch. 2.
3. ARNOLD, A., GUESSARIA I., **Mathématiques pour l'informatique**, MASSON (France), 1993.
4. AUSFELDER, C., MAIK J.P., KERMAD L., GENTINA J.C., DELFIEU D., MOISAND R. and SAHRAOUI A.E.K., **Integration of operating modes in the control of flexible manufacturing systems combining synchronous and asynchronous approaches**, Proc. of IEEE Int. conf. on Systems Man and Cybernetics (SMC 93), Le Touquet, France, 1993.
5. BILAND, P., **Modélisation des Modes de Marche d'un Système Automatisé de Production**, Ph.D. dissertation, Ecole de Centrale de Nantes, Nantes, France, 1994.
6. BILAND, P., DEPLANCHE A.M. and ELLOY J.P., **A Model for Operating Modes of Computer Integrated Manufacturing Systems**, Proc. of the European Workshop on Integrated Manufacturing Systems Engineering (IMSE 94), Grenoble, France, 1994.
7. BOIS, S., CRAYE E. and GENTINA J.C., **Manager of Working Modes**, CIM Integ. Design. in Manufacturing 21(3), 1992.
8. DANGOUMAU, N., **Contribution à la Gestion des Modes des Systèmes Automatisés de Production**, Ph.D. dissertation, Université des Sciences et Technologies de Lille, Lille, France, 2000.
9. DANGOUMAU, N., DUMERY J.J., FAURE J.M. and CRAYE E., **Prise en compte des modes de marche dans le pilotage**, in Fondements du pilotage des systèmes de production, HERMES (France), IC2, 2002, Ch. 5, pp. 155-180.
10. DANGOUAMAU, N., **Utilisation du Graphe Fonctionnel pour la gestion des modes d'un SAP**, presented at the DES research team seminar, LAGIS, juin 2003.
11. GONDRAN, M. and MINOUX M., **Graphes et Algorithmes**, Collection de la Direction des Etudes et Recherches d'Electricité de France, EYROLLES (France), 1995.
12. HAMANI, N., DANGOUMAU N. and CRAYE E., **Design Analysis and Verification of the Model of Component**, Proc. of the 10th Int. Multi-Conf. on Advanced Computer Systems (ACS 03), Miedzyzdroje, Poland, 2003.
13. HAMANI, N., DANGOUMAU N. and CRAYE E., **A Comparative Study of Mode Handling Approaches**, Proc. of the 35th Int. Conf. on Computers & Industrial Engineering (CIE 05), Istanbul, Turkey, 2005.
14. HAMANI, N., DANGOUMAU N. and CRAYE E., **A Computer Aided Tool for Specification and Verification of the MoC and the MoF**, Proc. of the Int. Joint Conferences on Computer, Information - International Conference on Industrial Electronics, Technology & Automation, and Systems Sciences, and Engineering (CISSE-IETA'06), USA, 2006.
15. KERMAD, L., CRAYE E., BOUREY J.P. and GENTINA J.C., **The Exploitation Modes of Flexible Manufacturing Systems**, Proc. of IEEE Int. Conf. on Systems Man and Cybernetics (SMC 94), San Francisco, USA, 1994.
16. PARAYRE, T., SALLEZ Y. and SOENEN R., **Model of the State of Operation of Automated Systems**, Proc. of the 8th Int. PROLAMAT conference (PROLAMAT 92), Tokyo, Japan, 1992.
17. TOGUYENI, A.K.A., ELKHATTABI S. and CRAYE E., **Functional and/or Structural Approach for the Supervision of Flexible Manufacturing Systems**, Proc. of IMACS-IEEE Multi-Conf. on Computational Engineering in Systems Applications (CESA 96), Lille, France, 1996.
18. TOGUYENI, A.K.A., **Contribution à la tolérance aux fautes des Systèmes Flexibles de Production Manufacturière**, H.D.R. dissertation, Université des Sciences et Technologies de Lille, Lille, France, 2001.