Forbidden and Preforbidden States in the Multi-model Approach

Oulaid Kamach, Laurent Piétrac and Éric Niel
INSA de Lyon, Laboratoire d'Automatique Industrielle (LAI)
Bat. Antoine de St-Exupery
25, Av. Jean Capelle
http://www-lai.insa-lyon.fr
69621 Villeurbanne cedex. France

oulaid.kamach@insa-lyon.fr

***Abstract*— This paper deals with operating mode management of Discrete Event Systems (DES) and this contribution is based on Supervisory Control Theory (SCT). Our aim is to extend SCT by introducing a mechanism for managing different operating modes for the controlled system. An operating mode corresponds to a specific system structure (engagement or disengagement of different system components) and specified tasks. Mode management will consist in controlling switching between modes with a view to handling models of reasonable size. Our approach is a multi-model one and involves representing a complex system by a set of simple automata models, each of which describes the system in a given operating mode. The adopted approach assumes that only one attempted operating mode is activated at a time, whilst other modes must be deactivated. The switching problem may be defined as finding compatible states, when controlled system behavior switches from one operating mode to another. The major contribution of this paper is the avoidance of switching from states (called forbidden states) with ghost compatible states in the selected operating mode. These states are called ghost because their existence would potentially violate a defined selected mode specification.**

I. INTRODUCTION

Operating mode management for DES remains a challenging problem and is the subject of considerable research [1], [2], [3], [4], [8], [15]. Existing work on operating mode management for DES focuses on problems of characterisation and switching between modes [1], [2]. However these approaches are not based on any formal models and they possess neither any validation mechanism of possible alternations (enabling and validity of switching between modes) nor any validation mechanism of deadlock research. To overcome these drawbacks in the Dynamic Hybrid Systems context , most works suggest novel methodology for synthesizing switching controllers and define the synthesis problem as finding the condition, on which a controller should switch system behavior from one mode to another to avoid a set of bad states [3]. [15] presents a framework for designing stable control schemes for systems whose dynamic equations change as they evolve in several operating modes. An appealing alternative is switching control schemes. Here, a different controller is applied to each operating mode and the stability of the overall system is ensured through a suitable switching scheme. In the approach of [4], a supervised control structure integrating operating mode detection and an active accommodation loop is designed. Active control accommodation is based on indirect switching control because it depends on detection of the actual process model.

Based on SCT (initiated by Ramadge and Wonham [6]), the approaches proposed by [8] and [7] apply the macro-action concept; operating mode management is ensured by activation of only one mode at any one time. Conscious of the advantages offered by [8] and [7], we extend these approaches to take the following statements into account.

1) A process comprises several components and not all components are used in every operating mode.
2) Specifications defined for each model can be conflicting, when switching from one mode to another (unlike the approach [5] in which all objectives must be concurrently achieved) and this may cause system blocking.

We have introduced a framework for modeling and switching, which takes into account the above statements [13]. The models considered feature processes and specifications, and more specifically, components engaged in a given operating mode. The multi-model approach involves representing a complex system by several simple models (each process model is associated with a specification model in a given operating mode). Each model is a partial description of the system in a given operating mode. Initially, only one model is activated and the nominal operating mode is generally assumed. All other modes are deactivated. Common component engagements are possible in each considered mode and the concept of tracking is introduced. This means maintaining a trace of events that have occurred for the common components. We have therefore extended each considered process and specification model by adding a specific state called the inactive state. The set of the events making it possible to switch from one model (process and specification) to another is called the set of the switching events. The difficulty of such an approach resides both in the building of extended models, which characterise different operating modes and in defining a switching mechanism allowing us to track explicitly the behavior of each model. This switching mechanism, characterized by information channels, is based on a set of traces generated in the model previously deactivated, to determine a suitable starting or recovery state for the recently activated model. Our approach applies to the mechanism for switching between different process and specification models, which have been extended to determine their compatible connection states. Finding the states from which these models need to be activated, whilst ensuring adequacy between current process dynamics and control decisions, has solved the problem of the mechanism for switching between specification models. In this paper, we extend the approach fof [13], [14] by considering a problem of switching from states with potentially ghost compatible connection states in the selected operating mode.

In Section II, switching between modes is ensured by tracking model S_i/G_i to ensure compatibility between the current state and all previous mode changes.

Intuitively, a state q in a model S_i/G_i is said to be compatible with a state q' in a model S_j/G_j, if the set of the common components between the two modes i and j have the same activity in the two considered states and the controlled process behavior S_i/G_i (resp. S_j/G_j) corresponds to a defined desired language of mode i (resp. mode j).

Based on Kumar's algorithm [12], we thus develop an algorithm, which allows forbidden and preforbidden states to be avoided.

II. MULTI-MODEL APPROACH

A real system involves a set of nominal and degraded modes. We adopt the following notation to deal with this. The set of operating modes is denoted by $I = \{1, 2, \ldots, n\}$, where $n \in \mathbb{N}$ and $n \geq 2$. By convention, we assume initially that the activated mode is mode 1. For each operating mode i, we associate an automaton model $G_i = (Q_i, \Sigma_i, \delta_i, q_{i0}, Q_{im})$ coupled by its owen supervisor, the set Σ' of switching events is defined as follows: $\bigcup_{i,j,i\neq j}^{n}\{\alpha_{ij}\}$, where α_{ij} represents the event ensuring switching from mode i to mode j. These multiple switching events mean that several switchings are possible: switching from mode 1 to mode 2, switching from mode 1 to i, from mode 2 to k, etc. These switchings must induce a trace memorization step because of common component engagement. Let us consider a case in which switching takes place. from mode i to mode j, then from mode j to mode k. In this case, we have to memorize controlled process S_i/G_i history in mode i prior to initial switching, then controlled process S_j/G_j history in mode j prior to the second switching. All these history recordings are required to determine the starting states in each mode (*i.e.* in each state of process G and specification S engaged in that mode) to which switching leads. These recordings are performed by the information channel denoted by π_{ij} (Figure 1), where:

Definition 1

$$\textit{Let} \quad \pi_{ij} : \Sigma_i^* \longrightarrow \Sigma_j^* \quad \textit{such that} \quad \forall \sigma \in \Sigma_i \text{ and } \forall s \in \Sigma_i^* :$$

$$\pi_{ij}(\varepsilon) = \varepsilon$$

$$\pi_{ij}(s\sigma) = \begin{cases} \pi_{ij}(s)\sigma \text{ if } \sigma \in \Sigma_i \cap \Sigma_j \\ \pi_{ij}(s) \text{ if } \sigma \in \Sigma_i \setminus \Sigma_j \end{cases}$$

This projection function definition restricts neither alphabet Σ_i nor alphabet Σ_j. In the particular case in which $\Sigma_j \subseteq \Sigma_i$, this function corresponds to the canonical projection used conventionally in SCT [9], [10], [11]. This function "erases" effectively from a string s those events σ that are not included in the set of common events $\Sigma_i \cap \Sigma_j$. This allows the behavior of common components only to be tracked.

In S_j/G_j, projection π_{ij} is used to identify the output states of intersection components of S_i/G_i, when α_{ij} occurs.

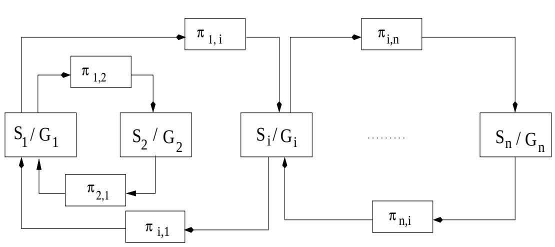

Fig. 1. Exchanges of necessary informations for management of modes

Formally, the set of mode n starting is given in the form (q, x), where q is the starting process model state that will be given by proposition 1. x is the starting specification model state that will be given by proposition 2. For more details, the reader could refer to [13].

Proposition 1

Let models G_1, G_2,...,G_n characterize the dynamic process in each operating mode.

1) *Determine a partial function C, defining possible $i - to - j$ switchings in C, if and only if there is a switching from S_i/G_i to S_j/G_j.*
2) *$I = \{1\}$. I represents the set of mode indices from which switching events will be considered events, starting from the initial mode.*
3) *While $I \neq \{\}$ do:*
    a) *$L = \{\}$. L is a temporary set allowing determination of modes indices from which switchings with the following step will be considered.*
    b) *For each $i \in I$: let L_i be the set of modes such that, for all j in L_i, the $i - to - j$ switching in C.*
        i) *For each S_j/G_j such that $j \in L_i$:*
            A) *Determine the set of starting states by applying: $\delta_{j,ext}(q_{jin}, \alpha_{ij}) = \delta_j(q_{j0}, \pi_{ij}(K_{qq'}))$[1] ($\forall s \in K_{qq'}$, $\alpha_{ij} \in follow(s)$[2]). This needs to be performed for all $K_{qq'}$ languages. There are several possible q and q' states.*
            B) *$C = C - \{i \to j\}$, $i \to j$ represent switching from mode i to mode j.*
        ii) *$L = (L \cup L_i) \cap dom(C)$[3]*
    c) *do $I = L$* ♦

The above proposition adopts formally the state from which the model G_j ($j \in \{1, 2, \ldots, n\}$) will be activated (the starting state). The following

[1] $K_{qq'}$ is the language containing all the sequences with starting state q of model S_i/G_i as origin state and a final state like the starting state q' of this model

[2] Denote by $follow(s)$ the set of events which follow the sequence of events s

[3] dom(C) represent the field of function C *i.e.*, the set of the indices i such that $i \to j$ belongs to C.

proposition establishes the switching mechanism between specification models by searching the states from which these models must be activated, whilst ensuring adequacy between current process dynamics and control decisions.

Adopting the following notations:

$\Sigma(q)$: represents the set of generated process events from state q

$\Sigma_a(x)$: represents the set of enabled events from specification state x.

$Re(x, S)$: are the specification states reachable from state x

$Re(q, G)$: are the process states accessible from state q.

Proposition 2

Let q_l, q_k, ..., q_n be the starting process G_i states.

1) *Determine for each starting state q_i, the desired language K_{q_i} elaborated from this state.*
   Do $H = X$. Initially H is the set of specification S_i states.
2) *For each q_i do:*
   a) *Calculate $\Sigma(q_i) \cap K_{q_i}$. This represents the set of process events generated from state q_i and belonging to desired language K_{q_i}.*
   b) *For each specification state $x \in H$ do:*
      i) *Calculate $\Sigma_a(x)$.*
      ii) *Calculate $\Sigma_a(x) \cap \Sigma(q_i)$. This is the set of process events generated from state q_i and enabled from specification state x.*
      iii) *If $\Sigma(q_i) \cap K_{q_i} \neq \Sigma_a(x) \cap \Sigma(q_i)$ then $H = H - \{x\}$. $H - \{x\}$ is the set H derived of all states x, which do not check the condition.*
      iv) *While $card(H)$ [4] $\neq 1$ do:*
         A) *Calculate $Re(x, S)$.*
         B) *Calculate $Re(q_i, G_i)$.*
         C) *If for all $x' \in Re(x, S)$ and for all $q' \in Re(q_i, G_i)$, there is an events sequence that checks $\delta_i(q_i, s) = q'$ and $\xi_i(x, s) = x'$, such that $s\Sigma(q_i) \cap K_{q_i} \neq s(\Sigma_a(x) \cap \Sigma(q_i))$ then $H = H - \{x\}$.*
      v) *State x checking that $card(H) = 1$ is consequently unique compatible starting state of specification model.* ⧫

[4] $card(H)$ represents the number of elements in H

A. Complete Definition Of S_{iext}/G_{iext}

The previously established propositions makes it possible to complete building the extended controlled process for each operating mode i. In the following, we define in formal terms wide models (S_{iext}/G_{iext}) for each operating mode i: the extended controlled process model for each operating mode $i \in I$ is given by automaton model S_{iext}/G_{iext} defined formally by: $S_{iext}/G_{iext} = (X_{iext} \times Q_{iext}, \Sigma_{iext}, \xi_{iext} \times \delta_{iext}, (x_{i0ext}, q_{i0ext}), X_{imext} \times Q_{imext})$ in which:

- $X_{iext} \times Q_{iext} = X_i \times Q_i \cup (x_{iin}, q_{iin})$,
- $\Sigma_{iext} = \Sigma_i \cup \Sigma'_i$ where Σ'_i is the set of events allowing to leaving or returning to mode i,
- $(x_{i0ext}, q_{i0ext}) = \begin{cases} (x_{i0}, q_{i0}) \text{ if } i = 1 \\ (x_{iin}, q_{iin}) \text{ if } i \neq 1 \end{cases}$
- $X_{imext} \times Q_{imext} = X_{im} \times Q_{im}$,
- extended transition function $\xi_{iext} \times \delta_{iext}$ is given as follows:
  1) $\forall(x, q) \in X_i \times Q_i$ and $\forall\sigma \in \Sigma_i$ if $\xi_i \times \delta_i((x, q), \sigma)$ exists (*i.e.* $\xi_i(x, \sigma)$ exists and $\delta_i(q, \sigma)$ exists) then: $\xi_{iext} \times \delta_{iext}((x, q), \sigma) := \xi_i \times \delta_i((x, q), \sigma)$
  2) all other transitions will be determined by using the proposition 1 and proposition 2.

III. FORBIDDEN COMPATIBLE STATES

In this section, we study the problem of switching from states in which compatible states in the selected mode are ghost (these state are called ghost, because their existence would potentially violate the defined selected mode specification). For the sake of brevity, a controlled process state will be denoted by y. To ensure better understanding and uphold intuitively the concept, only 2 modes will be considered in the following section. As denoted in the previous section, each operating mode is represented by a process model assigned with a specification model. We recall that our contribution above is an algorithm which generates a set of compatible connection states between modes. Specifically, we have shown that if we leave controlled process S_i/G_i from a state y, we must thereby activate the controlled process S_j/G_j from a state y', such that y' is compatible with y. However, the problem is what will happen when state y' is ghost in the controlled process S_j/G_j?.

To grasp our proposition, let us consider the following example.

⌜ *Example*

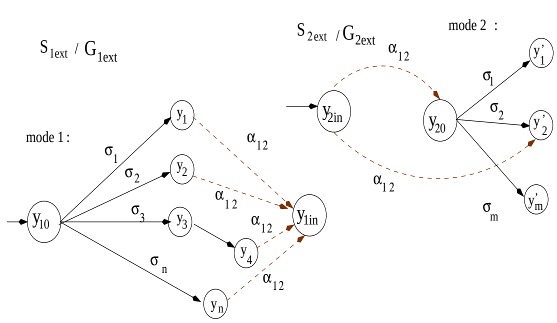

Fig. 2. example of application

Assuming that initially only mode 1 is activated, so from y_{10}, occurrence of event σ_1 leads S_1/G_1 to state y_1, in which switching event α_{12} is possible. Switching event α_{12} can occur in several states of model S_1/G_1: y_1, y_2, y_4 and y_n. When this event occurs, model S_{1ext}/G_{1ext} enters state y_{1in} (proposition 1 and 2). On the other hand, the set of compatible connection states of y_1 and y_2 in mode 2 are assumed to be y_{20} and y'_2 respectively. However, when switching event occur from state y_4 and y_n, their comaptible connection states in mode 2 do not exist, so y_4 and y_n are forbidden. In this example, we have illustrated only the problem of switching from states in mode 1, in which their compatible connection states in mode 2 are ghost. We can encounter the same problem on switching from mode 2 to mode 1. ⌟

Based on Kumar's algorithm [12], we suggest a methodology for ensuring switching between enabled compatible connection states. For each operating mode i, the strategy adopted can be informally described in proposition 3. However, we must firstly give the formal definition of forbidden and preforbidden states.

Definition 2

A state y is called a:

1) *Forbidden state if and only if:*
   - *the switching event can occur from y,*
   - *the compatible state of y is does not exist in the reachable selected mode.*
2) *Preforbiddden state if and only if:*
   - *the switching event cannot occur from y,*
   - *there is a sequence of uncontrollable events $s \in \Sigma_{ui}^*$, whose occurrence leads to a forbidden state.* ♦

Proposition 3

Step 1: *calculate controlled process S_i/G_i ($L(S_i/G_i)$ is assumed controllable with respect to G_i)*

Step 2: *identify all forbidden states $\mathcal{BS}$(mode i),*

Step 3: *identify all preforbidden states $\mathcal{PBS}$(mode i),*

Step 4: *delete from S_i/G_i all states in $\mathcal{BS}$(mode i) and $\mathcal{PBS}$(mode i) (also all transitions associated with these states),*

Step 5: *delete all states y of S_i/G_i from which there are no paths to y from the initial state of S_i/G_i.*

A controllable event leading to either a forbidden state or a preforbidden state can be directly disabled. On the other hand, in the case of an uncontrollable event leading to a forbidden state, we therefore disable the controllable event leading to the state, from which the sequence of uncontrollable events can occur. The language obtained in this way is controllable. There is therefore a supervisor achieving this language. The problem of calculating this supervisor has been omitted from this paper.

Remark 1

It should be remembered that this approach makes it possible to switch only between existing compatible states enabled in two operating modes. It does however restrict, in terms of permissivity, the controlled process behavior in these two operating modes. ♦

IV. EXAMPLE OF APPLICATION

The system is comprised of three machines, as shown in Figure 3. Initially, buffer B is empty and machine M_3 is performing other tasks outside the unit, but it intervenes when M_1 breaks down. Starting in state I_1, machine M_1 takes a workpiece (event b_1) (resp. event b_3) from an infinite bin, thereby moving to state W_1, machine M_1 may either complete its work cycle, returning to state I_1 (event e_1), or else break down (event f_1), moving to state D_1. It remains in D_1 until occurrence of repair event (r_1). M_2 operates similarly, but takes its workpiece from B and deposits it, when finished in an infinite output bin.

In this example, we assume that only M_1 can break down and that M_1 cannot recover its nominal use if M_3 is working. Figure 4 shows automaton models G_i of each machine M_i.

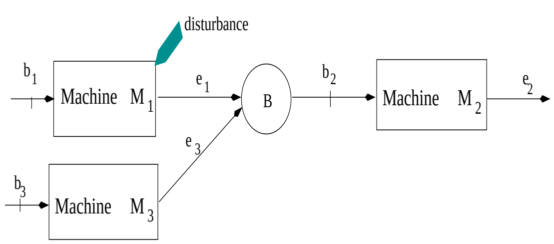

Fig. 3. Manufacturing system with three machines and Buffer

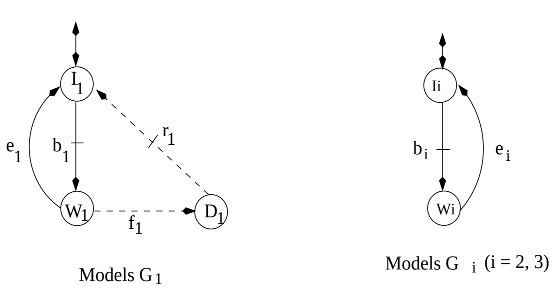

Fig. 4. Automaton models for machines M_i ($i \in \{1, 2, 3\}$)

In this example, the different operating modes considered are one degraded mode and one nominal mode.

In nominal mode (labeled n) the M_1 and M_2 machines are operating. The degraded mode (labeled d) corresponds to operating machine M_3 instead of machine M_1, which has failed, whilst machine M_2 is in operation.

The global alphabet is $\Sigma = \Sigma_n \cup \Sigma_d \cup \Sigma'$ where:

$$\Sigma_n = \{b_1, b_2, e_1, e_2\}, \Sigma_d = \{b_3, b_2, e_3, e_2\},$$
$$\Sigma' = \{f_1, r_1\}.$$

Σ_n represents alphabet in nominal mode, Σ_d represents alphabet in degraded mode and Σ' is the set of switching events. The designer has included different possible switchings. We assume that the system is initially in mode n, so occurrence of switching event f_1 will lead the system to mode d. In degraded mode, occurrence of switching event r_1 leads the system to the nominal mode. Figure 5 depicts the process models in nominal and degraded mode.

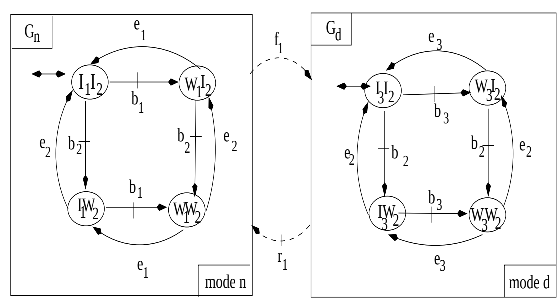

Fig. 5. nominal and degraded process model

For each process model, we now assign a corresponding specification model. The specifications state simply that B must be protected against underflow and overflow. In the nominal mode, the corresponding specification assumes that the buffer B capacity is 3 workpieces. In degraded mode, the corresponding specification assumes that the buffer B capacity is 1 workpiece.

Specification models for each operating mode are represented in Figure 6.

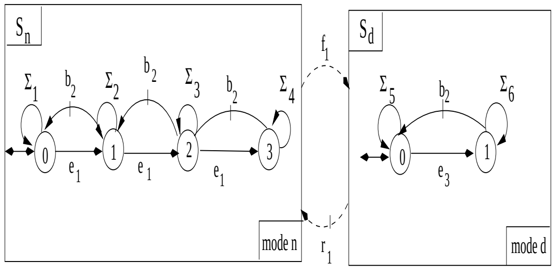

Fig. 6. nominal and degraded specification model

The selfloops Σ_1, Σ_2, Σ_3, Σ_4, Σ_5 and Σ_6, are $\Sigma_n - \{e_1, b_2\}$, $\Sigma_n - \{e_1, b_2\}$, $\Sigma_n - \{e_1, b_2\}$, $\Sigma_n - \{b_1, b_2\}$, $\Sigma_d - \{f_3, b_2\}$ and $\Sigma_d - \{b_3, b_2\}$ respectively.

Having set up process and specification models for each operating mode, we then obtain controlled process S_n/G_n (see Figure 7) and controlled process S_d/G_d (Figure 8).

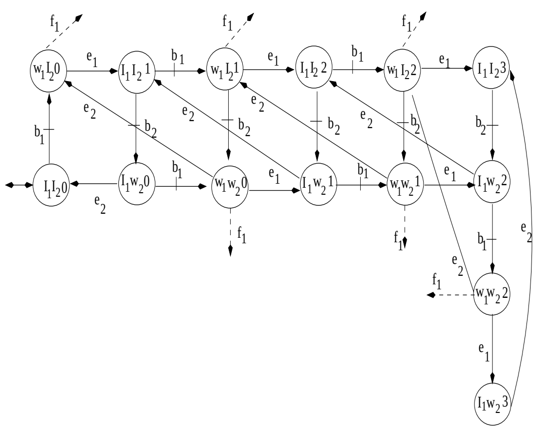

Fig. 7. Controlled process S_n/G_n in mode n

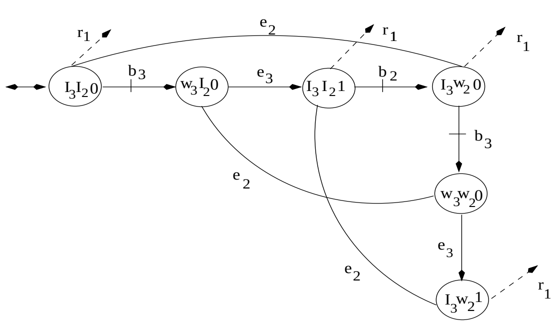

Fig. 8. Controlled process S_d/G_d in mode d

We note that, in the mode n, switching event f_1 can occur from states (W_1I_20), (W_1I_21), (W_1I_22), (W_1W_20), (W_1W_21) and (W_1W_22). By applying proposition 1 and 2, the set of compatible connection states, after switching from mode n to mode d, is shown in the following table.

States in mode n		**Compatible states in mode d**	
(W_1I_20)	legal state	(I_3I_20)	legal state
(W_1I_21)	legal state	(I_3I_21)	legal state
(W_1I_22)	forbidden state	(I_3I_22)	ghost state
(W_1W_20)	legal state	(I_3W_20)	legal state
(W_1W_21)	legal state	(A_3W_21)	legal state
(W_1W_22)	forbidden state	(I_3W_22)	ghost state

TABLE I

LEGAL, FORBIDDEN AND GHOST STATES FOR EVENT f_1

Applying proposition 3, we obtain the new controlled process model in mode n (see Figure 9).

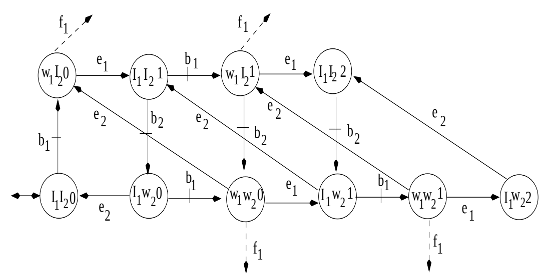

Fig. 9. New controlled process model S_n/G_n in mode n

Controlled process model S_d/G_d remains the same because the compatible connection states of the states in mode d, from which the switching event r_1 can occur are allowed in mode n.

V. CONCLUSION

This paper proposes a Supervisory Control Theory-based approach. We have presented a framework for managing switching of systems, whose dynamics change as they evolve in several operating modes. Our primary contribution is the introduction of a multi-model approach involving representation of a complex system by several simple models. Each model is a partial description of the system in a given operating mode. Initially, only one model is activated and the nominal operating mode is generally assumed. All other modes are effectively deactivated. Common components are possible in each considered mode and the concept of tracking is introduced. We have therefore extended each considered controlled process model and defined a switching mechanism, which makes it possible to track explicitly the behavior of each process model. This switching mechanism is characterised by information channels. In other words, we have shown that switching between modes is only between compatible states. We have shown also that there is a subset Q of states in mode i (resp. Q' in mode j) from which the switching event can occur and that their compatible connection states in mode j (resp. in mode i) are ghost. We have therefore proposed an algorithm permitting avoidance of both this subset of so-called forbidden states and of the set of so-called preforbidden states of mode i (resp. of mode j), from which the ocurrence of the uncontrollable event sequence leads to a forbidden state of Q (resp. of Q').

References

[1] Adepa, Guide d'Étude des Modes de Marches et d'Arrêts (GEMMA), 1981.

[2] N. Hamani and N. Dangoumau and E. Craye, *A formal approach for reactive mode handling*, IEEE international conference on Systems, Man and Cybernetics, SMC04, 2004.

[3] E. Asarin and O. Bournez and T. Dang and O. Maler and A. Pnueli, "Effective Synthesis of Switching Controllers for Linear System", *Proceedings of IEEE*, vol. 88, 2000.

[4] P. Charbonnaud and F. Rotella and S. Médar, "Process Operating mode Monitoring Process: Switching Online the Right Controller", *IEEE Transactions on Control Systems Technology*, vol. 31, 2002.

[5] S. Hashtrudi and R.H. Kwong and W.M. Wonham, "Fault Diagnosis in Discrete-Event Systems: Incorporating Timing Information", *IEEE Transactions on Automatic Control*, vol. 50, 2005.

[6] P.J.G. Ramadage and W. M. Wonham, "Control of discrete-event systems", *IEEE transaction on automatic control*, vol. 77, 1989.

[7] S. Chafik and E. Niel, "Hierarchical-decentralized solution of supervisory control", *3rd International Symposium on Mathematical Modeling, 3rd MATHMOD, Wien-Austria*, 2000.

[8] M. Nourelfath and E. Niel, "Modular supervisory control of an experimental automated manufacturing system", *Control Engineering Practice-Journal of IFAC*, vol. 12, 2004.

[9] F. Lin and W.M. Wonham, "Decentralized supervisory control and coordination of discrete event systems with partial observations", *IEEE transactions on automatic control*, vol. 44, 1990.

[10] K. Rudie and W. M. Wonham, "Think globally, act locally: Decentralized supervisory control", *IEEE transactions on automatic control*, vol. 37, 1992.

[11] K. Wong and J.G. Thistle and R. Malhame and H. Hoang, "Supervisory Control of distributed Systems: conflict resolution", *Discrete Event Dynamic Systems: Theory and applications*, vol. 10, 2000.

[12] R. Kumar, "Supervisory synthesis techniques for discrete event dynamical systems", *phd thesis, University of Texas, USA*, 1991.

[13] O. Kamach and L. Pietrac and E. Niel, "Generalisation of Discrete Event System multi-modelling", 11^{th} *IFAC Symposium of Information Control Problems in Manufacturing (INCOM'04), Salvador-Bahia, Brazil*, 2004.

[14] O. Kamach and L. Pietrac and E. Niel, "Supervisory Uniqueness for Operating Mode Systems", 16^{th} *IFAC World Congress. In Prague, Czech Republic*, 2005.

[15] M. Zefran and J. Burdick, "Design of switching controllers for systems with changing dynamics", *37th Conference on Decision and Control (CDC)*, 1998.