Identification of compatible states in switching mode

Asmae El ghadouali
Laboratory LIST,
FSTT, Abdelmalek Essaâdi University,
Campus Ziaten, BP: 416, Tangier, Morocco;

Oulaid Kamach
Laboratory LTI,
ENSAT, Abdelmalek Essaâdi University,
Campus Ziaten, BP: 1818, Tangier, Morocco.

Benaissa Amami
Laboratory LIST,
FSTT, Abdelmalek Essaâdi University,
Campus Ziaten, BP: 416, Tangier, Morocco;

ABSTRACT

Based on operating mode management, this paper introduces a new framework for studying dynamics of Discrete Event Systems (DES). Studied system presents several operating modes due to the state space explosion problem. To cure this problem, we propose a multi-model approach where each model describes a system in a given operating mode. We assume that only one attempted operating mode is activated at a time, whilst other modes must be inactivated. In order to ensure the alternation between these operating modes, we propose a formal approach using linear algebra. The commutation problem can be defined as compatibility problem when the behavior of physical system switches from an operating mode to another. The compatibility problem is treated as the consistency of current states when a mode generates an event activating the other mode. For this purpose, we introduce the notion of a compatible state in the switching mode.

General Terms:

Natural Language Processing, Adaptive Control

Keywords:

discrete event system, operating mode management, multi-model, safe commutation.ifx

1. INTRODUCTION

An usual way in industry to design discrete events systems (DES) consists of using the mode decomposition method to reduce the complexity of processes. Several works on SED have attempted to use mode management to design a complex system [1, 9, 8]. Some studies have focused on the automaton use for representing modes [5, 4] in bearing a reasonable size, while other researches deal with the problem of using several models [7, 3]. However, these approaches are not based on any formal models: they possess neither any validation mechanism of possible alternations (enabling and validity of switching between modes).
A physical system can be represented by different operating modes. Adjustment and maintenance of modes are examples of operating modes and will also be necessary to a production system. We are interested in modeling these operating modes by applying a multi-model approach [7], which involves designing a process model for each operating mode. In order to ensure the alternation between these operating modes, we assume that only one attempted operating mode is activated at a time, whilst other modes must be inactivated. Switching from mode to another one is equivalent to enabling and disabling the current mode. Changing mode and process structure raises problems such as to detect model switching and to maintain model tracking. The compatibility problem is treated as the consistency of current states in switching mode [6, 2] when a mode generates an event activating the other mode. The problem of commutation and compatibility between all designed models is formalized by the proposed framework.
This paper is organized as follows: Section 2 covers the selected design multi-model terminology of DES, where the new notions and extended models are introduced. Commutation formalism between designed process models is also briefly recalled in this section. Section 3 introduces the compatibility problem between different models. Section 4 comprises an illustrative example and study conclusions are presented in section 5.

2. OPERATING MODE MANAGEMENT

2.1 Definition of operating modes

A real system presents generally several operating modes. Our work tackles a multi-model approach that involves representing complex system by several simple models; each model describes a system in a given operating mode. This approach supposes that the system can be engaged only in one operating mode at a time, called active mode. The commutation between these operating modes takes place when a particular event occurs, called commutation event. Thus, the activation and inactivation of an operating mode take place by the occurrence of a commutation event. This event allows the switching from the mode in which a system performs perfectly in its task, known as nominal mode, to a mode for continuing a task in spite of a failure, known as degraded mode. A physical system involves a set of nominal and degraded modes. Indeed, each system admits that only one mode of normal functioning (nominal mode); by contrast, it can have several failure modes (degraded mode). In this approach, the nominal mode will always be considered the first selected mode.

DEFINITION 1. *Let $I = \{1, 2, ..., n\}$ a set of operating modes, where $n \in \mathbb{N}$ and $n \geq 2$. For each operating mode i, we associate to an automata model $G_i = (Q_i, \Sigma_i, \delta_i, q_{i,0}, Q_{i,m})$ where:*

—Q_i is the set of states of mode i,
—Σ_i is the alphabet of symbols,
—$\delta_i : Q_i \times \Sigma_i \rightarrow Q_i$ is the partial transition function.
—$q_{i,0}$ is the initial state in the mode i, $q_{i,0} \in Q_i$,
—$Q_{i,m}$ is the subset of marker states, $Q_{i,m} \subseteq Q_i$.

For any state $q \in Q_i$ and any event $\sigma \in \Sigma_i$, we write $\delta_i(q, \sigma)!$ (resp. $\delta_i(q, \sigma)\neg!$) if $\delta_i(q, \sigma)$ is defined (resp. isn't defined). This definition of δ_i can be extended to a partial function for $Q_i \times \Sigma_i^* \rightarrow Q_i$, such that $\forall s \in \Sigma_i^*$ and $\forall \sigma \in \Sigma_i$, $\delta_i(q, s\sigma) = \delta_i(\delta_i(q, s), \sigma)$ and $\forall q \in Q_i$, $\delta_i(q, \varepsilon) = q$.

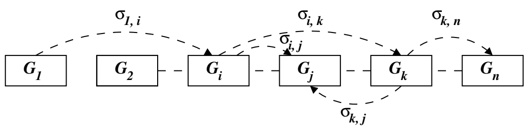

Fig. 1. Commutation between different models.

The set Σ_i^* contains all possible finite strings (i.e., sequence) over Σ_i and the null string ϵ. The language generated by G_i, denoted by $L(G_i)$, is also called the closed behavior of G_i: $L(G_i) := \{s \in \Sigma_i^* | \, \delta_i(q_{i,0}, s)!\}$.
The global set of events Σ_{global} of a system is given by the union of all alphabets Σ_i of elementary automata models G_i increasing by the set of commutation events Σ_c. Furthermore, the set of commutation events is disjoint of the different set of models: $\Sigma_c \cap \Sigma_i = \emptyset$ (for $i \in I$). Although, $\Sigma_i \cap \Sigma_j$ can't empty (common components between modes).

2.2 Commutation between Different Models

Considering several operating modes, our generalized approach is meant to define the n operating modes and the m possible commutation. The set Σ_c of commutation events is defined as $\cup_{i,j,i\neq j}^{n}\{\sigma_{i,j}\}$ where $\sigma_{i,j}$ presents the event ensuring the switching between mode i to mode j. For a given commutation, we specify that for a given mode several switching events can be considered. Moreover, from a mode i another commutation event $\sigma_{i,k}$ can lead to a mode k. This commutation mechanism is illustrated by Fig. 1. However, in each switching mode k, we must correctly determine the starting state after the commutation from the model G_i. To do this, we first have to extend the model G_k (resp. G_i) by adding respectively inactive state $q_{in,k}$ (resp. $q_{in,i}$) to states set of the model G_k (resp. G_i) based on concept of significative state suggested by Dangoumau [1]. The occurrence of commutation event $\sigma_{i,k}$ will lead model G_i to its inactive state $q_{in,i}$ and the process model G_k will be activated from its inactive state $q_{in,k}$. The activated process model at a given time is, thus, the only model from which the current state is different to the inactive state. Reciprocally, the state of all inactive models is their inactive state. So the use of this inactive state ensures that only one mode is active in a time.

DEFINITION 2. *The extended model for each operating mode $i \in I$ is given by automata model $G_{i,ext} = (Q_{i,ext}, \Sigma_{i,ext}, \delta_{i,ext}, q_{i,0,ext}, Q_{i,m,ext})$ in which:*

—$Q_{i,ext} = Q_i \cup \{q_{i,in}\}$*: extended set states with an inactive state,*

—$\Sigma_{i,ext} = \Sigma_i \cup \Sigma_c^i$*: extended alphabet with the set of commutation events Σ_c^i enabling to leave or to return to the mode i;*

—$q_{i,0,ext} = \begin{cases} q_{i,0} & \text{if } i = 1\text{: Initially, } G_1 \text{ is in its initial state} \\ q_{i,in} & \text{if } i \neq 1 \text{ : models } G_i \text{ are assumed desactivated} \end{cases}$

—$Q_{i,m,ext} = Q_{i,m}$ *($q_{in,i}$ will never be marked state),*

—*The extended transition function $\delta_{i,ext}$ is defined as follows:*

—$\forall q \in Q_i$ *and* $\forall \sigma \in \Sigma_i$ *if* $\delta_i(q,\sigma)!$ *then* $\delta_{i,ext}(q,\sigma) := \delta_i(q,\sigma)$*: this extended function is the same as transition function if we consider only non extended alphabet Σ_i,*

—$\forall q \in Q_i$ *from which commutation event $\sigma_{i,j} \in \Sigma_c^i$ can occur, then $\delta_{i,ext}(q,\sigma_{i,j}) = q_{i,in}$: extended transition function allows model G_i to be deactivated if the commutation event occurs.*

With the regard to process, the main aim of operating mode management is to define the starting state of model G_j ($q_{j,start} := \delta_{j,ext}(q_{j,in}, \sigma_{i,j})$, with $((i,j) \in I^2$ and $i \neq j)$) and, in turn, the return state of process model G_i ($q_{i,start} := \delta_{i,ext}(q_{i,in}, \sigma_{j,i})$). The notion of starting state will be defined later.

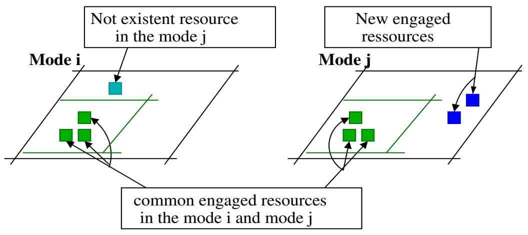
Not existent resource in the mode j
New engaged ressources
Mode i
Mode j
common engaged resources in the mode i and mode j

Fig. 2. Common resources of the modes i and j.

This approach makes it difficult to build the extended models and to ensure the switching mechanism. Also, in the context of the multi-model approach, our research brings together a formal concept of these notions: inactive state and switching mechanism. The latter can be defined as a compatibility problem when its behavior switches from an operating mode to another. Proposing a notion of compatible state allows to determine the starting or return state in the switching mode.

3. COMPATIBILITY BETWEEN TWO MODELS

Multi-model approach (see Fig. 1) is ideally suited for implementation of operating mode management and inter-mode phase switching in the DES. Each mode corresponds to a particular type of resources. Thus, this system don't require all resources in each operating mode. For instance, Fig. 2 shows that there are common resources engaged in two operating modes and some resources doesn't contribute to the production in mode i, but they intervene when a commutation from mode i to mode j occurs. During this engagement, the structure and the task of the system are fixed. We assume that the process model can change its structure when switching from operating mode to another has engaged new resources.
Considering a typology as that there is still a subset of common resources between two modes (see Fig. 2). It will be necessary to follow the evolution of these resources to correctly determine the states from which a functional connection will be allowed. We consider, thus, the tracking trace generated in inactive mode, which leads to a commutation event. The tracking implementation allows us to determine the connection state, i.e., adequate state in the novel mode where a commutation event occurs. This current state can be compatible state: starting or return adequate in the newly activated model. This has facilitated our work to determine the compatible state in the active mode without loss of information related in this changing mode. The compatible state is described by its activity sets of engagement resources in active mode. To determine this compatible state, we have to treat the information related to activities set in each current state.

DEFINITION 3. *A state q_i in the mode i is the cartesian product of all activities resources*[1] *engaged in this mode.*
For example, we assume that two resources $R1$ and $R2$ are engaged in the mode i respectively with (a_1^1, a_1^2), (a_2^1, a_2^2).
A state q_i in mode i is written:

$$q_i = (a_1^l, a_2^k) \;\; with \; l,k \in \{1,2\} \qquad (1)$$

Considering the evolution of common resources requires knowing all past history of the first mode selected. Or, the tracking mechanism allows to keep a record of the events leading to the one of these common resources to a switching mode. Therefore, we use the information given on accessible states, in inactive mode G_i, after the occurrence of traces generated by the occurrence of a commutation event. Each trace is an activation sequence S_i of the active mode G_j. Indeed, it is not a question of

[1]Each resource, engaged in the mode i, has a unique activity in a given state q_i.

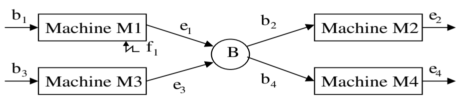

Fig. 3. Production system with four machines and intermediate stock.

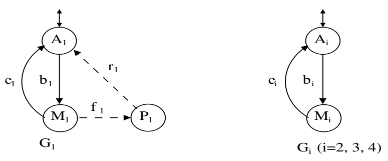

Fig. 4. Automata models of machines $Mi (i \in \{1, 2, 3, 4\})$.

applying our proposed approach to all existing sequences Σ_i^*. For that, we limit on the traces belonging to the language $L(G_i, \Sigma_c^i)$ with as origin state $q_{i,0}$ and as last event the commutation event $\sigma \in \Sigma_c^i$. This language is represented as follows:

$$L(G_i, \Sigma_c^i) = \{s \in L(G_i) \cap \Sigma_i^* |\, \delta_{i,ext}(q_{i,0}, s\sigma)!, \sigma \in \Sigma_c^i\} \quad (2)$$

The activities of a given state q_j in the active mode j are divided into two types of resources: common resources engaged in both modes i and j, and resources that don't exist in mode i. The compatible state, in the active mode j, have the same activities of common resources that the final state in inactive mode i. To keep the according information related to final states in the inactive mode G_i, we are based on the activation sequence belonging to the language $L(G_i, \Sigma_c^i)$.

DEFINITION 4. *A state q_i in the mode i is compatible with the state q_j in the mode j, when the common resources engaged in these two states have the same activities.*
Formally, let A_{q_i}, A_{q_j} respectively be the activities set in the two states q_i and q_j, with:

$$A_{q_i} = \{a_i^1, a_i^2, ..., a_i^n\}\ and\ A_{q_j} = \{a_j^1, a_j^2, ..., a_j^m\} \quad (3)$$

Let $A_{i,j}$ be the activities set of common resources engaged between the mode i and the mode j.
The state q_i is compatible with q_j iff:

$$A_{q_i} \cap A_{i,j} = A_{q_j} \cap A_{i,j} \quad (4)$$

4. APPLICATION

Consider the manufacturing system illustrated in Fig. 3; the system comprises four machines $M1$, $M2$, $M3$ and $M4$, and equipped with one buffer B. The machines are used to process apart and the buffer is used as storage between the machines with a maximal capacity of 1.
Initially, buffer B is empty and machine $M3$ is performing another task outside the unit, but it intervenes when $M1$ breaks down. With event b_1 (resp. b_3), $M1$ (resp. $M3$) picks up a workpiece from an infinite bin and places it in buffer B after completing its work (event e_1, resp. e_3). $M2$ (resp. $M4$) operates similarly, but takes its workpiece from B (event b_2 (resp. b_4)) and place it in an infinite output bin when it has finished its task (event e_2 (resp. e_4)). It is assumed that only $M1$ can break down (event f_1) and be repaired (event r_1) (as shown in Fig. 4).
Two operating modes are designed for the overall system: a nominal mode (G_n), in which $M1$, $M2$ and $M4$ produce and a degraded mode (G_d), in which $M3$ replaces $M1$. These two models are built up from models of $M1$, $M2$, $M3$ and $M4$, but they exclude f_1 and r_1 events (see Fig. 5). When f_1 occurs, the system switches to the degraded mode described by the model G_d. Occurrence of r_1 allows G_d to switch to G_n.

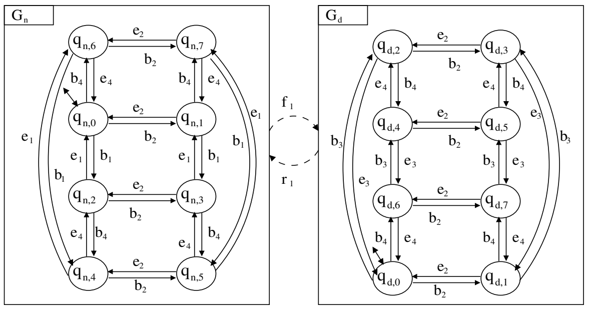

Fig. 5. Different possible commutations.

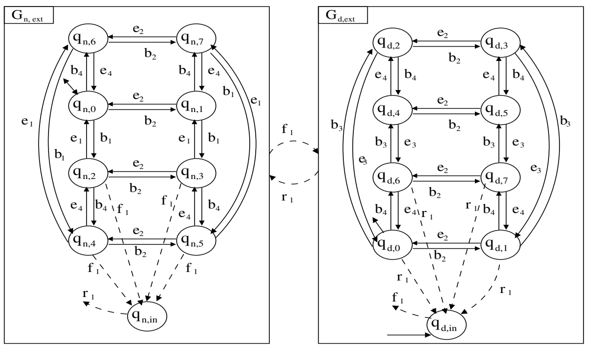

Fig. 6. Temporary extended automata of different operating models.

In this example, the set of events Σ_{global} can be partitioned into 3 sets: $\Sigma_n = \{b_1, e_1, b_2, e_2, b_4, e_4\}$ the nominal mode set, $\Sigma_c = \{f_1, r_1\}$ commutation events set and $\Sigma_d = \{b_2, e_2, b_3, e_3, b_4, e_4\}$ degraded mode set. The set of activities of resources, engaged in each operating mode, is given as follows: $A_n = \{A_1, M_1, A_2, M_2, A_4, M_4\}$ in the nominal mode and $A_d = \{A_2, M_2, A_3, M_3, A_4, M_4\}$ in the degraded mode.
By hypothesis, the event f_1 can be occurred from states in the nominal mode where the machines $M1$ is in normal operation. Before continuing, we assume, henceforth, that the event return r_1 can occur only from states of a degraded mode when the machine $M3$ has finished its task. Since there are common resources (machines $M2$ and $M4$) engaged between nominal and degraded mode. Then, the starting states in the degraded model are related to traces generated in the nominal model and leading to states where the commutation event can occur. During a change of operating mode, the state of common resources remains fixed. The inactive state $q_{n,in}$ (resp. $q_{d,in}$) of nominal model G_n (resp. degraded model G_d) is illustrated in the extended model $G_{n,ext}$ (resp. $G_{d,ext}$) (see Fig. 6).
In this example, we consider the commutation between nominal mode and degraded mode with the activation sequence $S_n = b_1\, b_2\, e_2\, b_4\, b_2$ (with $S_n \in L(G_n, f_1)$). The final state in nominal mode (previously activated), after the occurrence of S_n, is the state $q_{n,5}$ with activities $\{A_1, M_2, M_4\}$. The attainable state, after the occurrence of activation sequence S_n, is the state $q_{d,7}$. Thus, this state is compatible with state $q_{d,7}$ because both machines $M2$ and $M4$ (common resources) are found in work piece $\{M_2, M_4\}$.
Now, we have established the switching mechanism between nominal (resp. degraded) mode to degraded (resp. nominal) mode. We apply the same principle to all traces in language $L(G_n, f_1)$ of the nominal mode immediately before the breakdown event f_1. The same for language $L(G_d, r_1)$ of the degraded mode followed by the event r_1. Thus, we obtain extended model for each operating mode (degraded or nominal) in Fig. 7. These established models are nondeterministic. In fact, an event

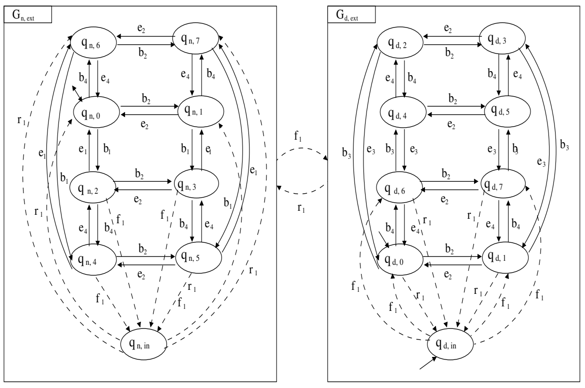

Fig. 7. Extended automata of different operating models.

such as the failure f_1 leads from the inactive state to several different states in the degraded model. To raise the indeterministic, we can apply existing algorithms to render extended models deterministic.

5. CONCLUSION

This paper deals with the compatibility of states when a switching between the operating modes takes place. Our approach introduces a definition of compatible states and gives a solution to resolve the management alternation problem of different operating modes. The compatibility problem is treated as the consistency of current states when a mode generates an event activating the other mode. Our current research is attempting to treat the diagnosis of unobservable events, applying the multi-model approach, in a physical system.

6. REFERENCES

[1] N. Dangoumau, A. Toguyéni, and E. Craye. Functional and behavioral modeling for dependability in automated production systems. *Journal of engineering manufacture*, 216:389–405, 2002.

[2] L. Piétrac G. Faraut and E. Niel. Identification of incompatible states in mode switching. In *ETFA IEEE conference*, pages 121–128, 2008.

[3] L. Piétrac G. Faraut and E. Niel. Formal approach to multimodal control design: Application to mode switching. *IEEE Transactions on Industrial Informatics*, 5:443–453, 2009.

[4] T. Gautier J-P. Talpin, C. Brunette and A.Gamatie. Polychronous mode automata. In *6th ACM & IEEE International conference on Embedded software*, pages 83–92, 2006.

[5] F. Maraninchi and Y. Rmond. Mode-automata: a new domain-specic construct for the development of safe critical systems. *Science of Computer Programming*, 1:219–254, 2003.

[6] L. Piétrac O. Kamach and E. Niel. Repulsive/ attractive discrete state space sets for switching management. *Studies in Informatics and Control Journal (SIC)*, 16:83–96, 2007.

[7] L. Pitrac O. Kamach and E. Niel. Multi-model approach to discrete events systems: application to operating mode management. *Journal of Mathematics and Computers in Simulation*, 70:394–407, 2005.

[8] F. Rotellaand P. Charbonnaud and S. Mauoar. Process operating mode monitoring process: switching online the right controller. *IEEE Transactions on Control Systems Technology*, 31, 2002.

[9] M. Zefran and J. Burdick. Design of switching controllers for systems with changing dynamics. In *37th Conference on Decision and control*, pages 2113–2118, 1998.