Proceedings of the 17th World Congress
The International Federation of Automatic Control
Seoul, Korea, July 6-11, 2008

A contribution to the validation of operating mode switching: application to satellite

S.I. Gonzalez Berlanga*, E. Niel*, B. Zouari, J.P. Blanquart*****

**Université de Lyon, Laboratoire Ampère, Institut National des Sciences Appliquées de Lyon, France (saul.gonzalez@insa-lyon.fr, eric.niel@insa-lyon.fr)*
***LIP2, Faculté des Sciences et Techniques de Tunis, Tunisie (belhassen.zourai@fst.mu.tn)*
****Astrium EADS, Toulouse, France (jean-paul.blanquart@astrium.eads.net)*

Abstract: We propose a methodology for modeling systems with different operating modes using Nested Petri Nets (NPNs) based on Valk's approach, where each token can be also considered as one Petri Net. NPNs provide a powerful tool for concurrent modeling and introduce interesting properties such as synchronization at a hierarchical level. In order to manage operating modes of critical and complex systems these properties are used to define and link component behaviors to the global system, through synchronized transitions. In order to formally verify these properties, CTL formulae will be used, translated from a logical table of technical specifications. The formulae allow a formal validation of the model and an examination of its coherency when the system switches to a new operating mode, under the influence of exceptional events. This verification is possible namely by using logic programming tools for the simulation and model checking. It is illustrated through a case study concerning a satellite's control unit.

Keywords: Discrete event systems modeling and control; Automata, Petri Nets and other tools; Verification; Switching stability and control; Model validation in design methods; Design of fault tolerant/reliable systems.

1. INTRODUCTION

From a dependability standpoint, satellite is a particular embedded system for which maintenance is not feasible (no access for repair). In such a way the more the satellite will be autonomous, the higher the performance. Reconfigurations by exploiting redundancies are the ultimate actions which allow the mission to be continued before changing a mode. Reconfiguration means that due to some external or internal reasons – sensor/actuator breakdown, low energy- the currently active components are no more able to ensure the attempted mode without referring to its redundancy. Depending on the severity the failure events, if the operating mode could not be maintained, the mission must be interrupted and restarted in another one. According to the operability abilities, two types of switching mode are allowed; the first one is dedicated to a request while the second one is subject to the failure recovery management, so called FDIR[1] procedure. In such situations depending on the control architecture, failure recovery amounts to let the system itself manage the malfunctioning.
This involved problematic is typically relevant to Discrete Event System (DES), taking into account that only one mode is activated at a time and that switch is issued from an event occurrence.
Switching mode is a convenient approach to manage such operational mode changing. Therefore one must respect – none exhaustively- the following constraints:
1. The correctness of the switch event
2. The identification of the current state for the outgoing mode
3. The coherency of the incoming mode
4. The time intervals in which the mode switching is possible,
5. Stability, accessibility, transient mode……

All these constraints can normally be declared in the technical operating mode requirements. If it is the case, it is not always expressed so that the verification should simply be done. The first two constraints are related to diagnosis and malfunctioning knowledge, the third one is related to control abilities, the fourth constraint is dedicated to performance finally the last constraints handles qualitative properties relevant to qualitative approaches.

With a complementary standpoint to control, qualitative properties have to verify that the technical requirements are always respected – on the adopted model -. This design step belongs to the last engineering investigation before implementation and is cost dependent. It focuses on possible lacks of incoherence between requirements and switch needs. Model-checking is now a well-established technique to achieve this efficiently, major problem result in interpreting the validation of the requirements into formulae. Even if it is time consuming, formal verification is extremely important and needed for dependability assessment. One more time, expertise is crucial at that point, only those who own the switch knowledge are able to express the correctness of the attempted properties.
In this paper, we investigate a methodological approach to deal with qualitative verification of technical operating mode requirements. The goal is to propose a global methodology

[1] Fault Detection Isolation Recovery

including concise modelling and adequate formulae generation for model-checking. Tools are also presented and applied in a safe satellite control architecture context. Section II defines the operating mode management and introduces the formulae expression. Section III presents the purposed methodology and tools. Section IV presents the "Pleiades" satellite as an example system, to illustrate the concepts and notions. Section V is related to modelling and section VI to formulae generation. Finally, in Section VII conclusions of this work are drawn.

2. OPERATING MODE MANAGEMENT AND PROPERTIES

Operating mode management for DES remains a challenging problem and is subject of considerable researchs (Zefran *et al*, 1998, Hamani *et al*, 2004; Asarin *et al*, 2000; Nourelfath *et al*, 2004). Existing work on operating mode management for DES focuses on problems of characterization and switching between modes (Asarin *et al*, 2000; Nourelfath *et al,* 2004). However, these approaches possess neither any validation mechanism of possible alternations nor any validation mechanism of deadlock research.

2.1 Organic definition of an operating mode

Operating mode M_k is described as a stable configuration, in which n components C_i are activated to fulfill an attempted task T_k through interaction $L_{i,j}$ with a component C_j.
Formally, the organic model for the k^{th} mode is $M_k = (C_j, L_{i,j}, T_k)$ for $i\ j = 1, \ldots, n$; n: total number of components and $k = 1, \ldots, m$; m: total number of modes. Considered components are physical or logical entities acting on other entities through $L_{i,j}$ (Kammach, 2005).
The activation level of each component j is identified by its charge $c_{j,y}$ and its redundancy typology $r_{j,q}$. Operating rate $c_{j,y}$ will be considered for a component j at three levels (minimum, medium, maximum) for a first assumption. Operating rate $c_{j,y}$ can be constant or may vary for a given mode M_i, depending on the involved task T_k. Redundancy index q characterizes the fault tolerance ability of a component C_j. It will be considered respectively as none (no redundancy ability), active (2 identical components activated, "warm redundancy"), passive (1 component in action, the second identical component will be activated when the first one will fail "cold redundancy"), vote (considered here as a 2-out-of- 3 structure).
A component will be then characterized under the assumption of its charge and redundancy ability as an entity $C_i = (r_{j,q}, c_{j,y})$ for $i = 1, \ldots, n$; $q \in \{none, act, pass, vote\}$, $y \in \{min, med, max\}$. When redundancy is discussed then the component will be described by its aggregated elements.
A configuration insures a mission T_k to be executed by activating components at a given operating rate $c_{j,y}$ and redundancy ability $r_{j,q}$. This means that a configuration is an architecture standpoint and could induce a dynamic behavior depending on the redundancy abilities of the component.
Then mode switching is allowed if and only if the redundancy structure is no more able to tolerate the faulty state **AND** there is a sufficient degree of freedom in component control terms. In order to achieve this functioning surely, technical requirements would have to be proved.

2.2 Formal verification of technical requirements

The most used verification method is certainly the test generation which consist in producing scenarios at the input of the system and observing its outputs. The problem is to be sure that all attempted component behaviors have been tested (coverage). This is nowadays not feasible and reminds an opened-problem. Other methods as *theorem proving* or *model-checking*, insure the verification exhaustively but are limited relating to the model size. This is due to the fact that these methods need to handle the global state space of the system (Berard, 2007).

Model-checking deals with the following question: given a system and a temporal logic property, does the system satisfying that property? The methodology is based on modelling steps resulting to formal representation of the system and of the properties respectively. Unfortunately even though the modelling steps are easy to follow by engineers as they can directly translate their knowledge *i.e.* properties formalization is difficult because the technical requirements are mostly expressed in an informal way. A lot of solvers are available; the most used are UPPAAL or SMV.

3. METHODOLGY AND TOOLS

A methodology for modeling systems is described with a hierarchical and multi-component approach, using Nested Petri Nets (NPN). A methodology to build CTL formulae is also presented for formal verification of technical requirements.

3.1 Nested Petri Nets

NPN is an extension of Petri Nets, introduced by Valk (1998) with a proposition of two levels Petri Nets formalism also known as Object Petri Nets.

Figure 1 presents a NPN constituted of one System Net (SN; top level) and one Object Net (ON; low level). It's easy to understand that SN manages autonomous components ON.

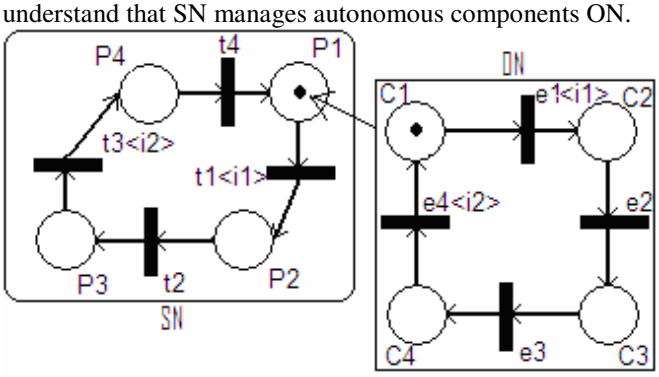

Fig. 1. Nested Petri Net

In place P1 of SN, one token encompasses a net itself where the dynamical behaviour is modelled by ON. For each transition of SN a set of labelled transitions exists related to some transitions of ON. For example if t1 fires from SN, the transition e1 of ON will be fired too, since the label <i1> is associated on both transitions. When transition t2 will fire (more than one token in place P2) two different behaviours can occur:

1. Transition t2 in SN fires and ON doesn't have any change (token in C2 reminds).
2. Transition t2 in SN and transition e2 in ON fire then place P3 of SN and C3 of ON will get one token respectively.

The second case represents autonomy between SN and ON which describes different dynamics.
Numerous formalisms have been proposed for NPN (Valk, 1998; Lomazova *et al*., 1998), more recently an extension of Valk's approach has been proposed by Leuschel (2004) mainly limited to one ON.

Definition 1. (Object Petri Net). An Object Petri net (OPN) is a quadruple $(SN,\{ON_i\}_{i\in I},I,\rho)$, the ON_i are object nets, I is a finite indexing set and ρ is a synchronization relation.

Definition 2. (System Net). A System Net (modeled by a NPN) is a tuple SN=(Σ,P,T,F,C,V,E) where the following hold:

(i) Σ is the set of types or colors with a subtype relation $\subseteq$ that is reflexive and transitive.
(ii) P is the set of SN places and T is the set of SN transitions such that $P\cap T=\phi$.
(iii) $F\subseteq(P\times T)\cup(T\times P)$ is the flow relation, also called the set of arcs.
(iv) $C:P\to\Sigma$ is a total function, called the typing function or coloring function of the system places.
(v) V is the set of variable symbols and to every $v\in V$ there is associated a $type(v)\in\Sigma$.
(vi) $E:F\to\langle V\rangle$ is the arc labeling function.
(vii) the set of variables on the incoming arcs of transition t is denoted by $(i.e,V_t=\{E((p,t))\mid(p,t)\in F\})$ and, for every variable v on one outgoing arc it is required that $v\in V_t$ holds.

Definition 3 (Object Net). An Object Net ON = (Q,B,G,W) is a P/T net, where:

(i) Q is a set of places and B is a set of transitions, with $Q\cap B=\phi$.
(ii) $G\subseteq(Q\times B)\cup(B\times Q)$ is the flow relation.
(iii) $W:Q\to\mathrm{N}$ is the arc-weight function.

Definition 4 (Synchronization Relation). Let SN= (Σ,P,T,F,C,V,E) be a system net and let $\{ON_i\}i\in I$ be a set of object nets $ON_i=(Q_i,B_i,G_i,W_i)$ such that T and all B_i are disjoints. Let $\tilde{T}=T\cup\bigcup_{i\in I}B_i$ denotes the set of all transitions. Then a synchronization relation is a tree-like relation $\rho\in\tilde{T}\times\tilde{T}$, such that its reflexive and transitive closure ρ^* is asymmetric and $(t',t)\in\rho\wedge(t'',t)\in\rho\Rightarrow t'=t''$.

3.2 Suggested model for the management of operating modes.

The system within its different operating modes will be modelled by one OPN. The different operating modes will be represented by the SN and each component's behaviour by one ON.

3.2.1 Construction of SN

Every operating mode M_i of the system will be represented by one place in SN. Considering Leuchel's approach, this proposal restricts SN to a state graph -just one mode once-, implying a restriction of *V*. In other words:

Let's call $\boldsymbol{M}$ the finite set of operating modes then for each $M_j\in M\Rightarrow\exists!\ p\in P$, for *j = (1…n)* n: the total number of operating modes.

The switching from an operating mode to another one will be ensured by T under the following condition:

That is to say $(M_j,M_k)_{(j\neq k)}\in M^2$ then from $M_j\to M_k$ $\exists!$ $t\in T$, in the strict direction of the implication.

Each transition of SN allows synchronization between elective components, insuring the right reconfiguration for a specific operating mode. This synchronization will be done by ρ.

Activation or deactivation of a component is supported by the arc labeling *E* function during the transition fires. The marking of SN represents the configuration of the system *i.e*. one place can consume from one up to *n* token for the configuration with *n* components at a time. A switch (firing of a synchronization transition) implies that all these token will be consumed from the input place and some of them (or new ones) placed in the output place representing the new configuration.

3.2.2 Construction of the ON

The modeling of each component is represented by one ON using two transition types:

External transitions: they are synchronized with those in SN. Their firing depends of ρ (synchronization relation). We refer after words:

$$Ejk \text{ with } \begin{cases} j: component's\ item \\ k: transition's\ item \end{cases} \quad \text{(E means external)}$$

Internal Transitions: they represent the component dynamics and are used to model the possible random events. Those transitions belong to the set of autonomy transitions of OPN. We refer after words:

Ijl with $\begin{cases} j : component's\ item \\ l : transition's\ item \end{cases}$ (I means internal)

In this work, these transitions are controlled by standard firing conditions of an ordinary PN.

3.3 Formal verification of technical requirements

In order to formally verify properties as described in section 1, a model-checking using CTL formula is elaborated. The approach consists in building a Boolean formulation starting from the technical requirements of the system, to translate it in CTL requests describing the properties to verify.

The formulae CTL will be adapted to the OPN context. The computed nodes are the different places P of SN and Q of ON. These nodes will correspond to one marking relevant to the properties to verify. By hypothesis the technical requirements, the set of the operating modes and switches are given.

Table 1: Technical Requirements

	Mode 1	Mode 2	……	Mode n
Comp. 1 (C1)	1	0	0	**1**
⋮	0	1	0	**1**
Comp. k (Ck)	**0**	**0**	1	**0**

Table 1 presents the different operating modes (column) and components (line). Boolean value "1" means that the "J" component is under operation, the opposite case for "0".

In order to validate the coherency of the operating mode, the first stage consists on translating the table into mathematical equations using Boolean's operators:

$$\text{Table 1} \Leftrightarrow \begin{cases} if\ Mode1 \Leftrightarrow (C_1 \wedge \neg C_2 \ldots \wedge \neg C_n) \\ if\ Mode2 \Leftrightarrow (\neg C_1 \wedge \ldots \wedge C_j \wedge \ldots \neg C_n) \\ \quad \vdots \\ if\ Mode\,k \Leftrightarrow (C_1 \wedge C_2 \wedge \ldots \wedge \neg C_n). \end{cases} \quad (1)$$

Verifying this set of equations concludes on the correct reachability of all operating modes. This property will be expressed by CTL formulae using ***EF*** operator, under the following hypothesis:

a) Each place of SN is called P_i *with* $i = (1...n)$.

b) $\exists! c_j \in Q$ with Q the set of places of ON representing the component in its operational state and $j = (1...n)$.

c) The marking of any ON will inform on the operational state of the component.

Then for each operating mode, the following CTL formula is verified:

$$EF(p_1 \wedge (c_1 \wedge \neg c_2 \wedge \ldots \wedge \neg c_n)) \quad (2)$$

If this formula is true, it guarantees that after a finite number of transitions the correct configuration for this operating mode is well reached. More formulae will be further presented associated to the satellite example.

4. STUDY CASE: PLEIADES SATELLITE

In this section the general structure of the Pleiades Satellite will be presented and more specific the control's unit from the hardware of Pleiades Satellite.

4.1 General description of Pleiades Satellite

The Platform and the Payload are managed by a central processing unit. According to the satellite's mission the payload change using different instruments. The equipments referenced to energy, thermal control and an attitude and orbit control measurement system are in the Platform. The study case is the management of the equipments of the platform between the different OMs of the Pleiades Satellite.

4.2 Different operating modes of Pleiades Satellite

In order to guarantee the properties described in section 1, the Pleiades Satellite has seven different operating modes {INIT (IT), STANDBY (SY), LAUNCH (LH), SAFE (SE), PSAFE (PE), SUSPEND (SD) and OPERATIONAL (OL)}. The switches between modes are given by the designer. An extract of the state graph with the premised operating modes is shown in figure 2.

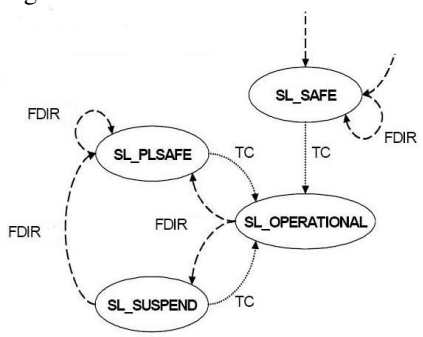

Fig. 2. Extract of System Modes. *Copyright © 2007 EADS Astrium.*

The switch between each mode and mode's reconfiguration are made by a request as Telecommand (TC) from earth or by the strategy called FDIR.

4.3 Technical requirements

Table 2 gives an extract of the technical requirements for some components {Processor Module (PM), Input/Output Thermal (IOT), Telemetry, Telecommand and Reconfiguration board (TTR)}. In reason of dependability the components are in warm redundancy (TTR) or cold redundancy (PM and IOT) under nominal mode (element A) or degraded mode (element B).

Table 2. Technical requirements by mode. *Copyright © 2007 EADS Astrium.*

	IT	SY	LH	SE	PE	SD	OL
PM (A or B)	1	1	1	1	1	1	1
IOT (A or B)	0	1	1	1	1	1	1
TTR (A and B)	1	1	1	1	1	1	1

Then for one mode, different configurations can exist due to the component's redundancy. Respecting section 2.1, mode M_{IT} = {*(PM, TTR)*, $L_{PM\text{-}TTR}$, *INIT*}, and mode M_{LH} = {*(PM, IOT, TTR)*, $L_{PM\text{-}TTR}$, $L_{PM\text{-}IOT}$, *LAUNCH*}.

4.4 Hardware of the Pleiades Satellite

An extract of the hardware of the Pleiades Satellite is shown (Fig. 3). The different components in the OBMU (On Board Management Unit) are in cold redundancy except TTR under warm redundancy.

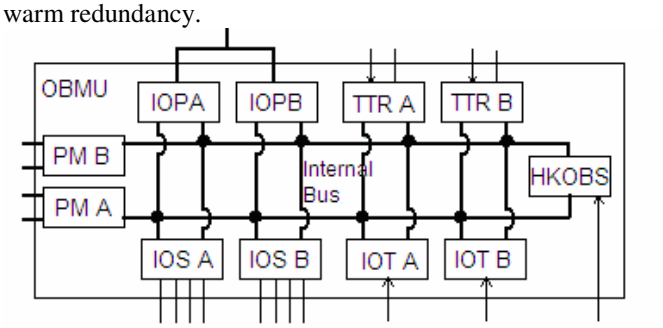

Fig. 3. Hardware of the Pleiades Satellite. *Copyright © 2007 EADS Astrium.*

In order to verify and validate that the system follow the technical requirements, the suggested model for the management of operating modes in section 3, will be used into the hardware's Pleiades Satellite.

5. MODELLING

The model of operating modes management for the "Pleiades" satellite is given by SN. ON depicts the component's dynamic (activation, redundancy characteristics). Following the requirements, the SN model is built starting from the state graph provided by the designers (Fig. 4). {P1: STAND BY, P2: INIT, P3: LAUNCH, P4: SAFE, P5: OPERATIONAL, P6: PLSAF, P7: SUSPEND}.

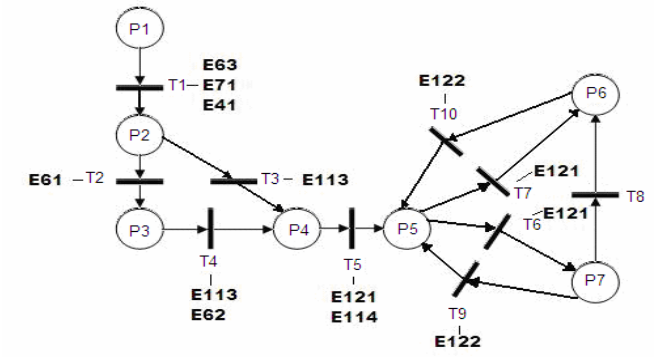

Fig. 4: SN modeling the operation modes of the "Pleiades" satellite

Initial marking corresponds to P1, it defines the standby operating mode. The number of tokens in this place will be equal to the number of components which are activated in this mode.

A component with redundancy will be modeled by one ON including dynamic's activation (A or B in the case of a double redundancy). In order to remain concise, limited number of components (IOT, TTR) is represented. It illustrates warm and cold redundancy for the "Pleiades" satellite. The dynamic's behavior depends on the redundancy's type.
IOT component runs in cold redundancy (element IOT_A and element IOT_B) with its maximal charge when operates, so it is defined (ref. section 2.1) as (*IOT, passive, max*).

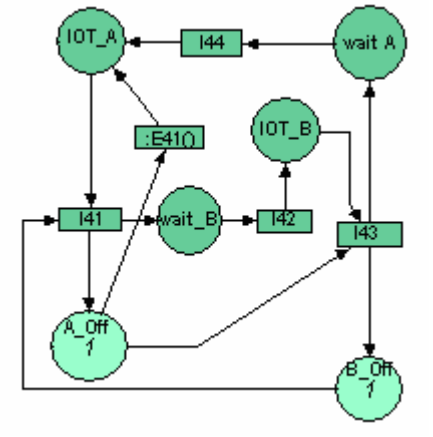

Fig. 5. ON for one IOT component with cold redundancy, expressed with RENEW

The ON model of IOT (Fig. 5) represents cold redundancy and is composed of five transitions (one external transition (E41), four internal (I41, I42, I43, I44)). The switch to an operating mode where IOT is active, will fire the transition "E41" in synchronization with the adequate transition of SN. The firing of internal transitions will depends of the internal behavior of IOT. If "IOT_A" fails then "I41" will be fired and "IOT_B" will be activated when its time of activation is accomplished.

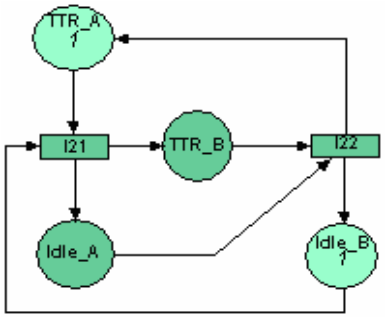

Fig. 6. ON of TTR in warm redundancy

TTR component runs in warm redundancy (2 identical elements TTR_A, TTR_B), so referred to the technical

requirements: (*TTR, active, TTR_A$_{max}$, TTR_B$_{min}$*). The initial marking from figure 6 is given by places "TTR_A" and "Idle_B" point at a TTR A active and a TTR_B with its operating rate minimum. The transition I21 fires when "TTR_A" fails. The transition I22 will be fired when "TTR_A" failure is recovered.

These examples show how SN manages the ON and how the component's dynamic is taken into account.

The synchronization relations (Fig. 4) between SN and the six components - modeled by their own ON constitute the "Pleiades" satellite architecture - are associated to the corresponding transitions. The SN's marking insures the right configuration for each operating mode.

6. PROPERTIES

Before to use model-checking which is the easier approach to check the model property per property, (model proving would be more convenient because it seems to be more exhaustive), the problem is to translate some technical requirements in mathematics formula. At this time, just some simple properties have been checked. The following list is made up of some conditions to check for critical systems relevant to reliability:

1. Mutual exclusion between different operating modes,
2. Mutual exclusion between two elements in cold redundancy,
3. Simultaneous inclusion of two elements in warm redundancy,
4. Prohibition of passage by certain states "X" to reach a desired state,
5. Obligatory passage through a state "X" to reach another

These conditions can be verified by the following requests in CTL:

condition 1:

$$\begin{cases} \neg EF(P_1 \wedge (P_2 \vee P_3 \vee \cdots \vee P_n)) \\ \neg EF(P_2 \wedge (P_1 \vee P_3 \vee \cdots \vee P_n)) \\ \vdots \\ \neg EF(P_n \wedge (P_1 \vee P_2 \vee \cdots \vee P_{n-1})) \end{cases} \quad (3)$$

condition 2: $\neg EF(E_A \wedge E_B)$ (4)

condition 3: $EG(E_A \wedge E_B)$ (5)

condition 4: $\neg EF(P_X \wedge EF(P_1 \wedge P_2))$ (6)

condition 5: $\neg EF((\neg P_X)\, U\, (P_2 \wedge M_2))$ (7)

7. CONCLUSIONS

A two steps methodology is proposed in this paper in order to validate the technical requirements of operating modes management for DES. The first one is dedicated to NPN modeling in reason of its power to encapsulate low dynamic level. Synchronization transitions are used to switch from one mode to another one. Each place of the top level represents a component configuration including own redundancy abilities. At this stage, some conventional properties related to Petri Nets may be done.

The second step is devoted to the model checking having in mind to validate technical properties in terms of switch and mode coherencies. The major advantage is to expose a set of concise tools for operating mode strategy assessments. Further works including temporal intervals and stochastic behaviors would enrich the contribution and look forwards to some dependability application such that with AltaRica.

Some characteristics describing a component, such as charge and redundancy, which are just empirical in this paper, can be used in further works for dependability assessment.

All the methodology steps have been developed using RENEW (logical program mainly designed to simulate NPN); ProB a logical program where the deadlocks and the reachability graph is automatically generated; XSB a logical program that execute XTL (executable temporal logic) to check CTL formulae in the satellite's example (Leuschel, 2004).

REFERENCES

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, pp. 1011-1025.

B. Berard. Model-checking temporisé, Ecole d'été temps réel, ERT 2007, Nantes, pp.97-109, 2007.

N. Hamani, and N. Dangoumau, and Craye, E. (2004) "A formal approach for reactive mode handling", IEEE, SMC04, pp. 4306-4311, The Hague, Netherlands.

O. Kammach, and L. Pietrac, and E. Niel. Multi-Model approach to Discrete Events Systems: Application to operating mode management, Mathematics and Computers in Simulation, vol. 70, n°5-6, p. 394-407, 2005.

M. Leuschel and B. Farwer.(2004) Model Checking Object Petri Nets in Prolog, PPDP'04, August 24–26, 2004, Verona, Italy.

I. A. Lomazova. Nested Petri nets — a formalism for specification of multi-agent distributed systems. Concurrency Specification and Programming (CSP'99), Proceedings, pages 127–140.

M. Nourelfath, and E. Niel, Modular supervisory control of an experimental automated manufacturing system, Control Engineering Practice, vol. 12, n°2, pp. 205-216, 2004.

R. Valk. Petri nets as token objects. An introduction to elementary object nets. Applications and Theory of Petri Nets 1998. Proceedings, volume 1420, pages 1–25. Springer-Verlag, 1998.

M. Zefran, and J. Burdick. Design of switching controllers for systems with changing dynamics, 37th CDC, pp. 2113-2118, Phoenix, Arizona, USA.

B. Zouari and R. Frefrita and E. Niel. A colored Petri net approach for the management of operating modes in discrete event systems, IFAC MCPL, Sibiu, Romania September 27-30-2007.