An Hybrid Method for the Validation of Real-Time Systems

Laurent Kaiser, Françoise Simonot-Lion

AN HYBRID METHOD FOR THE VALIDATION OF REAL-TIME SYSTEMS

Laurent Kaiser, Françoise Simonot-Lion

LORIA - ENSEM (CNRS-UMR 7503)
2, avenue de la Forêt de Haye, F54516 Vandoeuvre-lès-Nancy, France
Tel. +33 3 83 59 55 79
Fax. +33 3 83 59 56 62
{kaiser,simonot}@loria.fr

Abstract: We present a method for the temporal properties verification of complex systems. Its interest lies in a coordinated use of both exhaustive analyze and simulation techniques. The model is based on a unique formalism, called TIOSM (Timed Input output State Machine) which is a specialization of Timed Automata. The exhaustive analysis is done by model checking and concerns only partial models that represent critical parts of the whole system. The results obtained on these partial models are inserted in the global model that can be then simulated. In the paper, we give rules for defining partial model, analyzing them and integrating the results in the global model, that can therefore simulated.

Keywords: validation, timed automata, real-time, simulation, model-checking, complexity.

1. INTRODUCTION

The level of abstraction or the accuracy of a model is inverse ratio to its capacity to be treated in a bounded time. Two ways can be used for a model analysis:

- by exhaustive exploration of the model,
- by simulation of the model.

This last approach allows getting results quickly but the pertinence of these results is linked to the scenario and the simulation duration. In the first approach, results are deterministic but the combinatory explosion is a bar to exhaustive method application.

Nevertheless, exhaustive exploration of model can be required for critical parts of systems while for the other parts, validation can be done thanks to simulation techniques. For example, for specification of the METEOR subway control, exhaustive analysis of formal models is used only for parts involved in safety properties as doors closing and opening, emergency stop, … (Bustany, et *al*., 1996; Dehbonei, et *al*., 1995).

In order to take profit of the advantages of two approaches, we propose in this paper an hybrid method based on a unique representation of the complete system and which combines simulation with exhaustive analysis restricted to critical parts. The underlying formalism is a sub-class of Timed Automata (Alur and Dill, 1994; Alur *et al.*, 1995), called TIOSM, Timed Input Output State Machine (Koné et *al.*, 1995).

This paper will present successively in section 2 the TIOSM formalism and, in section 3, an overview of the hybrid validation method. The so-called “partial model” representing the critical part is defined in section 4. How to analyze each partial model and to build an abstraction of it (named a “black box”) is the purpose of section 5 and section 6. The integration of these “black boxes” in the complete model and the simulation process is shown in section 7. Finally we conclude in section 8.

2. TIMED INPUT OUTPUT STATE MACHINES

In Timed automata (Alur and Dill, 1994) each transition can be characterized by several clocks constraints; the form of each constraint is a non-bounded time interval. In TIOSM formalism, the form of transition firing time constraints is a bounded interval. Moreover, this formalism allows specifying the application as a set of communicating state-machines. In a Timed Input Output State Machine, two main attributes can act on the transition firing:

the reception/emission of messages and, timed constraints expressed on different clocks.

Formally, a TIOSM is defined by a tuple $T=(S,L,C,s_0,\varepsilon)$ (Koné et *al.*, 1995):

- S is a non empty finite set of states,
- L is a non empty finite set of messages,
- C is a non empty finite set of clocks,
- $s_0 \in S$ is the initial state of T,
- ε is a non empty finite set of transitions.

Any $t \in \varepsilon$ has the form $t=(s,\mu,D,Z,d)$ where:

- $s \in S$ is the origin state of t,
- $\mu = (\{!,?\}xLx\{\tau\}$; τ is an internal action, !a, ?a indicates an output or an input of message a, where $a \in L$,
- D is a non empty, finite list of temporal properties having the form $c\in[m,M]$ ($c\in C$, m, $M\in Q_+$),
- Z is a finite set of clocks to be reset after t firing,
- d is the target state of t.

In Fig.1, an example of transition characterization is illustrated.

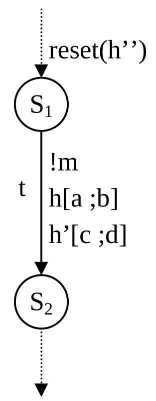

Fig. 1. Example of a TIOSM T

Definition: *several firing time of a transition t were defined:*

- $\theta(t,c)$ is the firing time of t with respect of constraints on the clock c (where $(c,[a;b])\in D(t)$)
- $\theta(t)$ is the firing time of t with respect of constraints on each clock c: $\theta(t) = \cap\theta(t,c)$ / $\forall(c,[a;b]) \in D(t)$

In order to avoid no-determinism, we add two hypotheses:

- every transition $t=(s,\mu,D,Z,d)$ may only send or receive a message (there is no internal action: $\mu \ \# \ \tau$)
- every pair of transitions (t,t') with s(t) =s(t') respect $\mu(t) \# \mu(t')$. In the same state, two transitions can't use the same message.

The firing transition policy for the transition characterized in Fig.1 is illustrated in Fig. 2 (in the document, we note the firing time of *t* by $\theta(t)$).

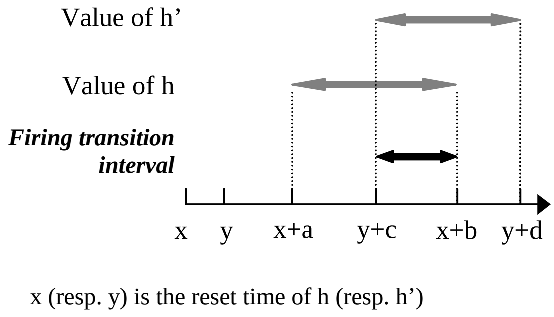

Fig. 2: Firing interval of t for TIOSM T

Some previous works are based on this formalism. In particular, the construction of an accessibility graph similar to the class graph obtained for Timed Petri Nets (Merlin and Farber, 1976; Berthomieux and Diaz, 1991) was specified in (Kaiser, 1996). An extension of the "on the fly" approach, introduced in (Fernandez et al., 1996), was developed for real-time properties verification in (Kaiser et *al.*, 2000a); finally, an adaptive tester, minimizing the number of inconclusive verdict, was proposed by (Kaiser et *al.*, 2000b). (Laurençot and Castanet, 1997) proposed a canonical tester construction used for real systems. A demonstration of equivalence between TIOSM and Time Petri nets was demonstrated in (Haar, *et al.*, 2000).

3. HYBRID VALIDATION METHODOLOGY

As introduced previously, we use for this methodology both exhaustive analysis for parts of the system and simulation for the whole system. For this purpose we must define four kinds of model:

- *Complete model* is the initial model of the whole system in terms of TIOSM.
- *Partial model* is a part of the *complete model* assuming to specify a critical part of the behavior of the whole system (TIOSM formalism)
- A *black box* is the abstraction of the *partial model* obtained after its validation;
- The result of this integration of all the *black boxes* in the *complete model* is called *derived model*.

The hybrid validation method manipulates these models along six steps. Two main stages are necessary. The first one concerns the validation of critical parts while the second one concerns validation of the whole system.

Fig.3 illustrates the three first ones whose purpose is to build and validate the *partial models*:

- Modeling of the whole system in TIOSM formalism (*complete model*)
- Identification of critical part and extraction of *partial models* from the *complete model*; each result is a particular TIOSM
- Exhaustive analysis of each *partial model*; this step consists in validation of *partial model* by application of model checking techniques

The next steps (see Fig.4) require that every *partial model* be validated; the role of this step is to build *black boxes* and *derived model* and to validate this *derived model* by simulation:

- For each *partial model*, specification of the corresponding *black box*
- Integration of each *black box* in the *complete model* in order to obtain the *derived model*
- Finally, simulation of the *derived model*.

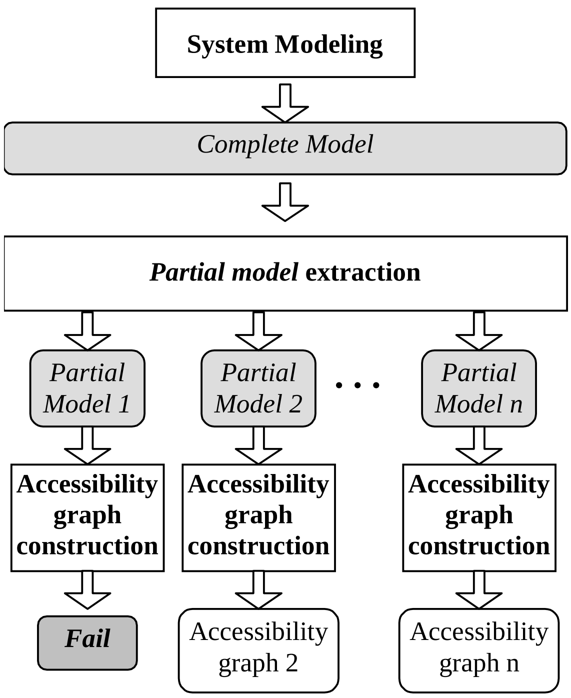

Fig. 3. *Complete* and *partial models* construction

4. PARTIAL MODEL DEFINITION RULES

The identification of critical part in the whole system and subsequently, in the *complete model* is an activity that needs both a know-how in TIOSM formalism and a good knowledge of the modeled system. Nevertheless, we specified a set of rules that guides the model designer to specify its *partial model*. These rules express that the defined *partial models* are "independent" of its environment in the whole system. Then we include temporal information in the obtained *partial models*.

4.1. Independence properties

Let T_S be the TIOSM modeling the whole system and T_{Si} ($i \in [1;n]$) be the TIOSMs modeling the *partial models*. The "independence" of T_{Si} is expressed by the following rules (these rules are illustrated on Fig.5):

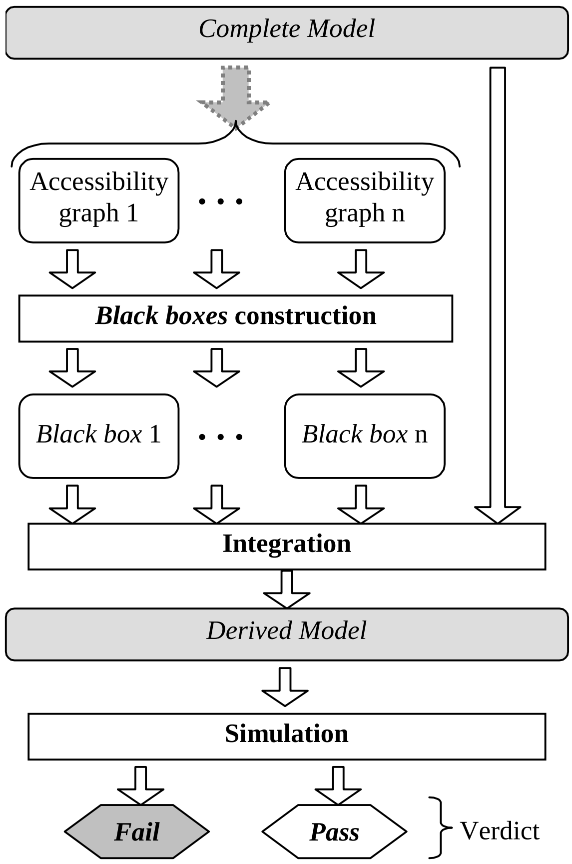

Fig. 4. *Black boxes* and *derived model* construction

- Each *partial model* T_{Si} must have one and only one "entry" state s (for example, in Fig. 5, S_2 is the "entry" state of *partial model* T_{S1} and S_3 is the "entry" state of *partial model* T_{S2}). It is noted enter(T_{Si}).
- Each *partial model* must have at least one "exit" state. We introduce the set exit(T_{Si}). For example, in Fig. 5, we have

  exit(T_{S1})={S_6}

  exit(T_{S2})= {S_6, S_7, S_8}.
- An "entry" state s or an "exit" state s' of a *partial model* T_{Si} can belong to another partial model T_{Sj} only if:

  $s \in$ exit(T_{Sj}) or $s' \in$ exit(T_{Sj})

- Origin or target of a transition t(s) in the *complete model* can be a "entry" state or an "exit" state of a *partial model* only if:
  $s = enter(T_{Si})$ or
  $s \in exit(T_{Si})$.

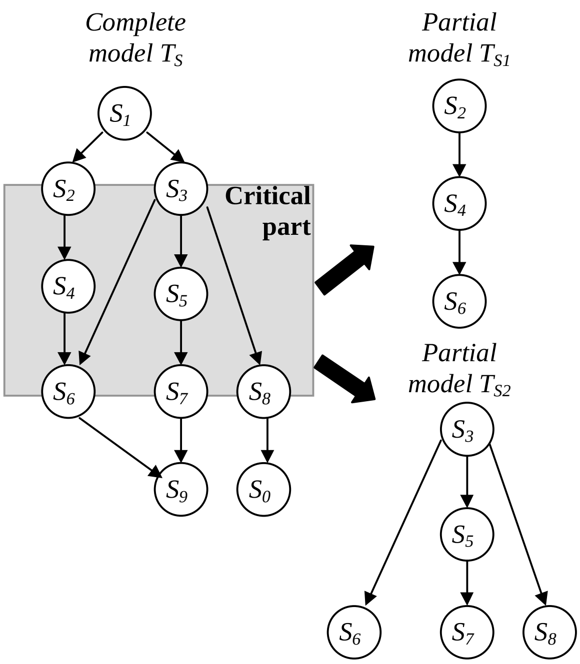

Fig. 5. Example of *partial model* definition.

4.2. Temporal attributes of partial models

Some clocks are not used in a *partial model* (not reset nor used in any constraint). Nevertheless, for the *complete model* simulation, any *partial model*, must know the value of these clocks when reaching their "enter" state. These clocks are added to the set *C* of the *partial model*. This will allow us to build inequalities systems taking into account their values.

5. ACCESSIBILITY GRAPH

Each TIOSM T_{Si} is exhaustively analyzed. We developed an algorithm based on the resolution of inequalities systems thanks to SIMPLEXE method. An accessibility graph, similar to the state class graph proposed by (Berthomieu et *al.*, 1991) for Timed Petri Nets model (Merlin et *al.*, 1976), is computed.

Each node N_k (a state class) of this graph is described by:

- a state denoted $st(N_k) \in S(T_{Si})$
- an inequalities system denoted $Q(N_k)$; this system contains information about the firing time of all transitions that will be fired later.

An edge e_k, between an origin and an extremity nodes (denoted $origin(e_k)$ and $extremity(e_k)$), represents a transition $tr(e_k) \in \varepsilon(T_{Si})$ whose firing is possible from the state $st(origine(e_k))$.

Below, we present the method and illustrate it by an example (see Fig. 6 for the TIOSMs describing the whole system, Fig. 7 for the *partial model* corresponding to a critical part of this system and Fig. 8 for the accessibility graph).

5.1. Initial inequalities system

Note, that as T_{Si} is treated independently, the instant at which, its initial state will be marked is not known. This instant depends of the trajectory that the whole system followed before reaching this state. N_{init}, the initial node, is defined by the state $s_0=enter(T_{Si})$ and, for each clock $h\in C(T_{Si})$, by an equation $h=h_0$ (where h_0 is a variable representing the value of h when reaching s_0). In the case studied in Fig.6, 3 clocks are used in T_{Si}, so the initial system is $Q_{init}=\{x=x_0, y=y_0, z=z_0\}$.

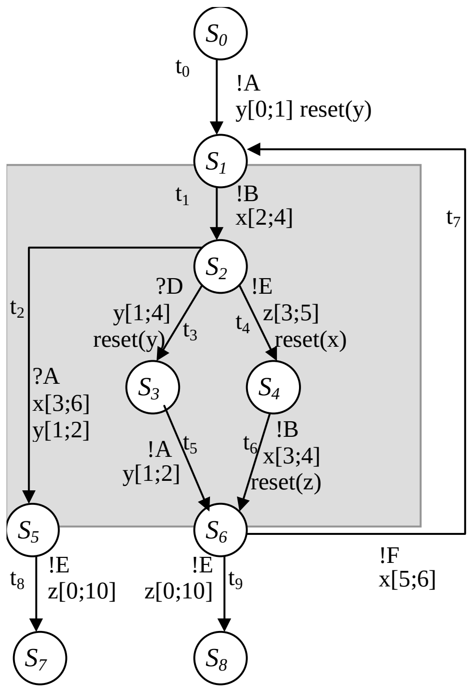

Fig. 6. *Complete model* T_S

5.2. Computation of nodes

The construction of the graph is done from the initial node N_{init}:

- $st(N_{init})=enter(T_{Si})$
- $Q(N_{init})=Q_{init}$

Then for each node N_j, the process is similar: let e_i the edge whose extremity is N_j and consider $tr(e_i)$ and $st(N_j)$.

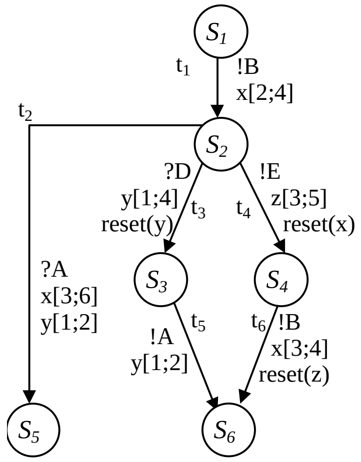

Fig. 7. *Partial model T_{Si}*

- In a first step, the different clocks $h \in C(T_{Si})$, are updated.
  - If *h* is reset after $tr(e_i)$ firing, then the equation become $h=0$.
  - If not, the value of *h* is defined by adding $\theta(tr(e_i))$ to the previous value of *h* before $tr(e_i)$ firing; this previous value is obtained from the inequalities system associated to $origine(e_i)$.

  For example, on Fig. 8, $Q(N_2)$ contains the equation $x=x_0+\theta(t_1)$ where $\theta(t_1)$ is the firing time of t_1, … and $Q(N_4)$ contains the equation x=0, because clock x is reset while firing t_4.
- Then, for each transition t_k whose origin is $st(N_j)$, we consider its clock constraints: for each constraint expressed on a clock *h*, $h \in [a,b]$, the inequality $a-h<\theta(t_k)<b-h$ is added at the system.

  For example, on Fig.8, the inequalities $3-x<\theta(t_2)<6-x$, $0-y<\theta(t_2)<4-y$, … are added to $Q(N_2)$). So, we obtain the inequalities system characterizing the node.

  In order to determine if a transition t_k can be fired from $st(N_j)$, we evaluate if the inequalities system admits at least one solution. This is done thanks to SIMPLEXE method. Furthermore, we compute the minimum and the maximum of t_k firing time.

  Note that if there is no solution, we conclude that this transition will never be fired.

If the construction of the accessibility graph shows that any exit state can be reached, it demonstrates a “livelock” in the specification. So, the critical part modeled by T_{Si} is not correct.

If not, this critical part is supposed to be correct (no deadlock, no “livelock”). Note that this step is important because the validation of each partial model is a necessary condition for the validation of the whole system.

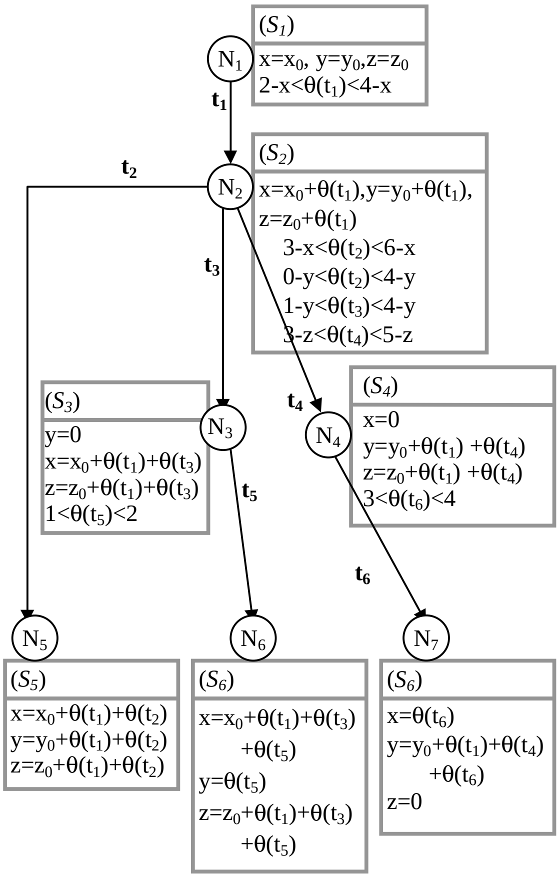

Fig. 8. Accessibility graph of *partial model T_{Si}*

6. BLACK BOXES CONSTRUCTION

This construction is illustrated in Fig. 9 that represents the *black box* for the *partial model* of Fig. 7. It is done only when every partial model of T_S is validated. If one at least is not validated, we conclude that the system modeled by T_S is not correct.

From each validated *partial model*, an abstraction of T_{Si} is done. It reflects temporal properties of T_{Si}. In fact, this abstraction, called *black box* specifies how the time can be passed from the arrival of the *complete system* at the $enter(T_{Si})$ to its arrival to a state belonging to $exit(T_{Si})$.

This *black box* has one and only one initial state $S_{init}=st(N_{init})$ (S_1, in the presented example) and one terminal state S_k for each terminal node N_k characterized by $(st(N_k),Q(N_k))$. In the same example, nodes N_5, $N_{6,1}$, $N_{6,2}$ give 3 terminal states S_5, $S_{6,1}$, $S_{6,2}$. Notice that in this example two nodes refer the same state S_6 in T_{Si}; so, we create two different states in the *black box*.

An inequalities system, Q'_k, is created and associated to each “black box” terminal state S_k that corresponds to a terminal node N_k;

Q'_k contains:

- $Q(N_k)$
- $Q_{init,k}$: all the inequalities that are associated to the transitions labeling edges on the path $\{N_{init}, N_k\}$ found in the accessibility graph.

For example, in Fig. 9, Q'_5 is defined by $\{Q(N_5), Q_{1,5}\}$.

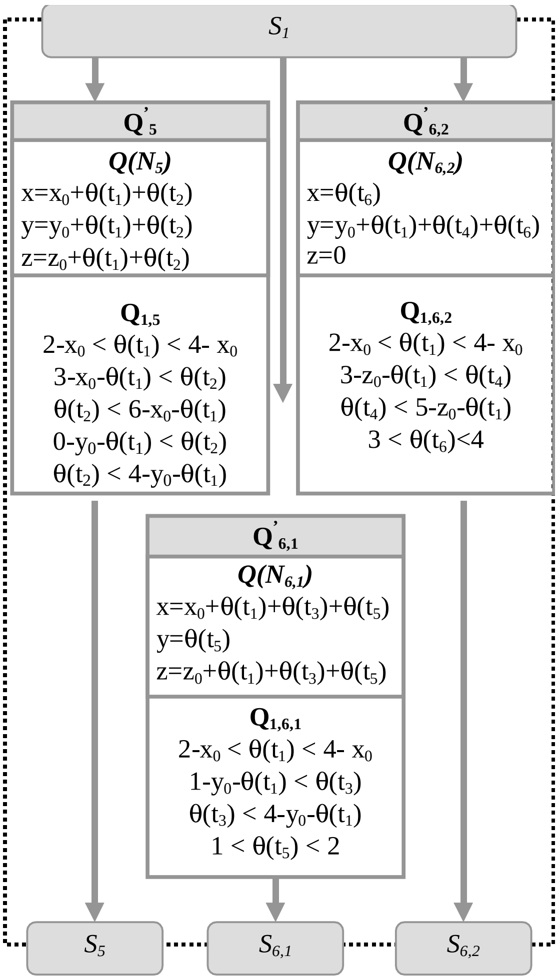

Fig. 9. Black box of partial model T_{Si}

7. DERIVED MODEL

7.1. Generation of derived model

The derived model that corresponds to the example (Fig. 6) is presented in Fig. 10.

The integration in a *complete model* of a "*black box*" corresponding to T_{Si} follows several steps:

- suppression in the *complete model* of each state and transition belonging to T_{Si}
- adding the "*black box*" to the obtained model

* each transition t leading, in the *complete model* to enter(T_{Si}), is linked to S_{init}. For example, in Fig. 10, t_0 and t_7 are connected to S_1

* each transition t, whose origin state s belongs to exit(T_{Si}), are linked to each *black box* terminal state associated to s. In Fig. 6, t_9 is connected both to $S_{6,1}$ and to $S_{6,2}$, while t_8 is connected only to S_5.

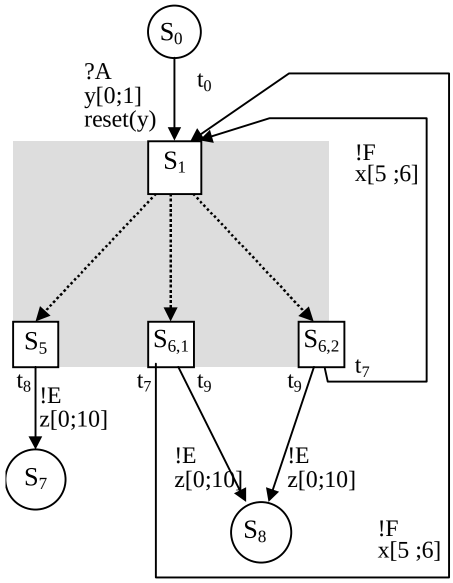

Fig. 10. The derived model of example (Fig.6)

7.2. Derived model simulation

Along the simulation process, we verify that, when reaching an initial state of a "*black box*", the value of the different clocks allows an exit of the "black box". The simulation algorithm progress from state to state and the calculus of transition firing time is based on the following rules:

- At initialization, all the clocks used in the *complete model* are reset,
- Let S be the current state at a simulation step; the objective of the simulation is then to determine if there is at least a reachable state and, if yes, at which time this state can be reached. Two cases can be identified:
  - S is not an entry state of any partial model; a transition t, from those that can be fired is chosen according to a given strategy (depth first, random choice, …) and an instant for this transition firing is determined (minimum, maximum, mean, random value); every clocks $h \in C(T_S)$ are then updated or reset if they belong to Z(t).
  - S is an entry state for a *partial model*; every clock variable representing partial model entry time must be updated (in example illustrated by Fig. 5 and 6, these variables are x_0, y_0 and z_0).

Thanks to the SIMPLEXE method applied to the inequalities systems associated to the *black box*, it is possible to find which exit states are reachable and when they can be reached.

Similarly to the precedent case, the simulation algorithm choose a reachable exit states and its reaching time among the possibilities given by SIMPLEXE method.

This mechanism stops in two cases.

- When, from a current state, any transition can be fired or any *partial model* exit state can be reached; if the system is not blocked in one of its terminal state, we identify a livelock and we prove that the modeled system is not correct.
- When the simulation duration is reached; in this case we conclude that the system is correct according to the simulation scenario.

Two simulations of system presented Fig. 5 are illustrated in Fig. 11 and Fig.12.

In the first case, for the tried scenario, there is no problem; the system reaches S_7 that is one of its terminal state. Note that, as we simulate the system, it is said to be correct according to this particular scenario.

In the second case, the system is blocked in S_1. So we conclude that the system is not correct.

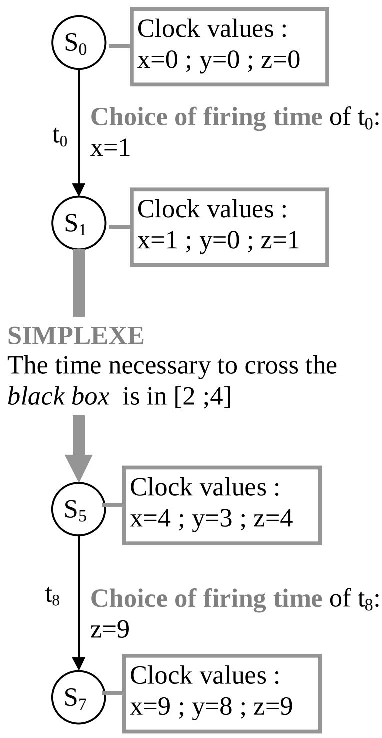

Fig. 11. *Complete system* is correct for this scenario

8. CONCLUSIONS AND FUTURE TRENDS

The hybrid method presented in this document shows how to combine simulation and exhaustive analysis. A unique formalism that is specialization of Timed Automata supports both the activities. The exhaustive analysis is done by model checking and concerns only the critical parts of a large system. So it limits the dimension of the accessibility graph. The results obtained on these partial models are inserted in the global model that can be then simulated.

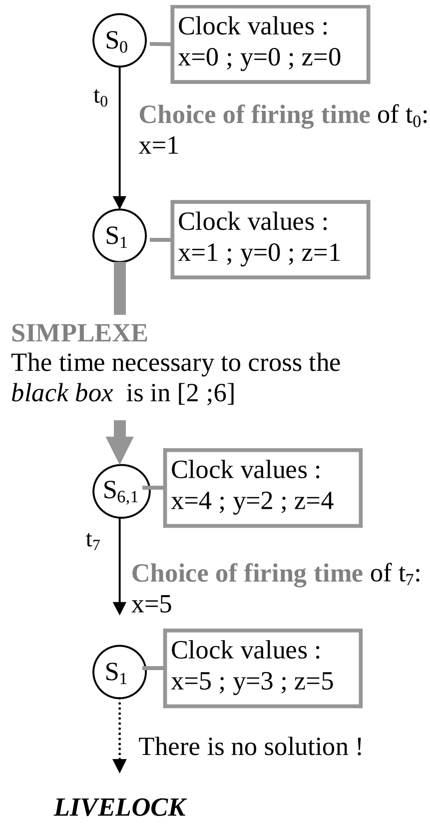

Fig. 12. *Complete system* is not correct

This method is implemented in a tool prototype Xtiosm (Santos Marques R., 2001). We are studying how to generalize the presented concepts in order to develop a library of re-usable model of partial system that are exhaustively “pre-analyzed”. This will allow the design of a whole model by “connecting” or “composing” several partial models modeled as “black boxes”.

9. REFERENCES

Alur R., D. A. Dill (1994). *The theory of Timed Automata*. In: Theorical Computer Science, vol.126, N°2.

Berthomieu, B., M. Diaz (1991). *Modeling and Verification of Time Dependant Systems Using*

Petri Nets. In: IEEE Transactions of Software Engineering, 17(3) :259 :273.

Dehbonei, B., F. Mejia (1995). *Formal Development of Safety-Critical Software Systems in Railway Signalling*. In : Applications of Formal Methods, Series in Computer Science, pp. 227-252, Prentice Hall International.

Desforges, P., F. Bustany, P. Behm (1996). The use of the B method for the realization of security software of the SAET Meteor. In: *General Review of the French Railway*. N°6. Dunod.

Fernandez, J-C., C. Jard, T. Jéron, C. Viho (1996). Using on-the-fly verification techniques for the generation of test suites. In : *CAV'96*, LNCS 1102, Springer Verlag.

Haar, S., L. Kaiser, F. Simonot-Lion, J. Toussaint (2000). *On Equivalence between Timed State Machines and Time Petri Nets*, Technical Report INRIA RR-4049.

Kaiser, L., (1996). Interopérabilité temporelle à l'aide d'automates temporisés à entrés-sorties, Master report, Nancy, France.

Kaiser, L., F. Simonot-Lion, O. Kone (2000a). Verification method of interoperability for real time systems. *In : S4th IFAC International Symposium on Intelligent Components and Instruments for ControlApplications - SICICA'2000*, Buenos Aires, Argentine

Kaiser, L., F. Simonot-Lion (2000b). Adaptative timed tests for temporal interoperability verification. *In 3rd IEEE International Workshop on Factory Communication Systems - WFCS2000, Porto, Portugal.*

Koné, O., R. Castanet (1995). Conformance with Time Extensions, Technical Report, Université de Bordeaux.

Merlin, P.M., D.J. Farber (1976). *Recoverability of Communication Protocols – Implications of a Theorical Study*. In: IEEE Transactions on Communications, 24(9) :1036-103.

Santos Marques R. (2001). Formal test generation - Xtiosm tool. *In: Internship Report, Lisboa University, Portugal*.