Modeling of Real-time Process Control

By
Mustafa-Sami M. Mustafa *

ABSTRACT

This paper discusses the successive steps used for modeling a real-time process control system. The term "process control" means the control of a process by a "computer". The "real time process" is distinguished from other types of processes by two basic features
- the fast response to sudden external events is an essential criterion, and
- the order in which the external events are occuring is the main critical factor [3]

These two private characteristics impose a set of conditions on the system response. The simultaneous occurence of events, must not result a random response. A predetermined action is necessarily decided.

Evidently, a successful system design starts by depicting the functional specifications into a suitable functional model. It is'nt enough to have a clear model, but it must be also free of problems. The possible problems can be detected and avoided by analysing the resulting model. The modeling tool which has a set of powerful analysis rules, is the Petri net. The Petri net structure proved effecient use in system modeling.

The chosen application,on the modeling procedure presented in the paper, is a double runway control system. A hierarchical construction of the model is illustrated with detailed explaination. This modeling procedure can be similarly applied to any other system.

Key terms: Real-time processing, Concurrent processing, System modeling, Petri nets

1. Definition of Real-time system

The common feature of a real-time process control is the feedback [1]. A continuous flow of data is transfered from the controlled process to the controller, which is in fact no more than a computer. The computer must respond to these data with an actual desired action to be followed by the process. In the real-time system, it's evident that a fast response is a basic requirement. A short time period is available to compute and to follow the desired action. The action is decided according to the values of the recently arrived data items. The correct response must be computed as a function of three components

a) the current state of the process,
b) the occured external events, and
c) the desired behaviour of the process.

This indicates that the real-time system needs more amount of control over timing considerations [2]. The control, under such

* Ass.Professor of Computer Engineering, M.T.C., Cairo, Egypt

time constraints, may only be realized by a fast microcomputer. To ensure no waste of time, the computer must be informed by the current state of the process during the shortest possible time period. This state must be also memorized in order to be used during all processing phase related to the current state. An image of the process reflected in the computer archticture [1], achieves three objectives

- the computer is informed by the current state of the process in zero time
- the switching to the next expected state is performed by the computer on the image
- the next state is checked on the image to detect any possible unexpected problem and avoid it

The desired response computed for each new state depends on the functional parts and the variables which are concernrd with the current configuration of the system. Therefore, the system activity can be subdivided into a set of smaller seperate activities. Each one of these activities can be realized by a processing module. The processing module may be executed only when the current state triggers it.

The block diagram illustrating the structure of the real time process control sytem is on figure 1. There is external link for data transfer, and internal link for controlling the execution of processing modules. The received data contains a set of input variables which is partitioned into two subsets. A subset supplied to the image part, representing the arrived events and it´s usually of logic type. Another subset for the computation of desired actions. The latter subset may be logic or numerical,and it´s supplied to the processing modules part. The image part controls the execution of different processing modules through the internal link.

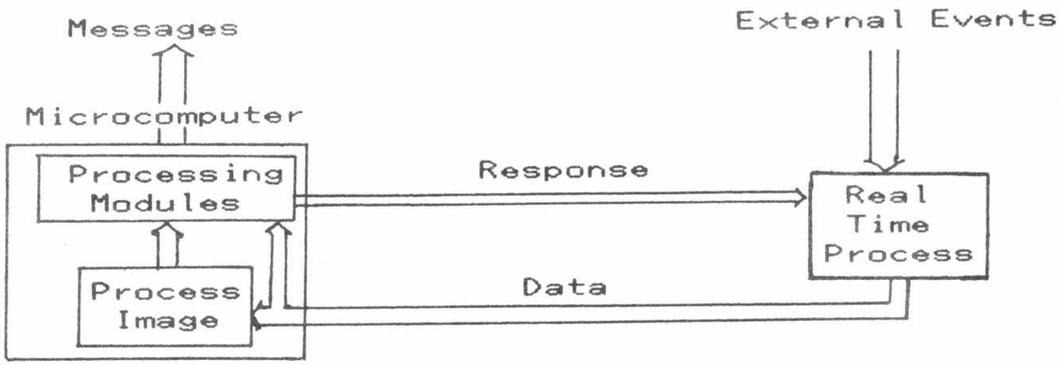


fig.1 Real-time process control system

The messages are a special type of actions generated to the operator for announcing an emergency state. The emergency state is out of normal operating states, which occurs oftenly if an accidental event arrives, or when no more processing can be accomplished (blocking state).

2. Defintion of Petri net as a modeling tool

Petri net is defined in many references, [as in 7,8,9], in various ways and with different symbols. However they all mean the same thing. The definitions stated in the paper are based on all definitions found in these references with a unified set of symboles. The stated definitions are selected so as to achieve the purpose of the paper, that is the modeling. Firstly, three basic definitions on structure, graph and execution of Petri net are given. A fourth definition is dedicated for the interpreted Petri net. Figures 3, 5, and 6 make the definitions more clear.

Definition 1: Petri net structure

" A Petri net structure is a four tuple C=(P,T,I,O) where P={ P1,P2,.....,Pn } is a finite set of places; $n > 0$, and T={ T1,T2,.....,Tm } is a finite set of transitions; $m > 0$. The two sets are disjoint $P \wedge T = \emptyset$. Mappings from transitions to bags of places defines the input function I: $T \longrightarrow P$, and the output function O: $T \longrightarrow P$"

Definition 2: Petri net graph

" A Petri net graph G is a bipartite directed multigraph, G=(V,A), where V={ V1,V2,.....,Vr } is a set of vertices and A={ A1,A2,...,Aq } is a bag of directed arcs, Ap=(Vi,Vj) Vi,Vj C V. The set V can be partitioned into two disjoint sets P (places) and T (transitions), such that $P \vee T = V$ and $P \wedge T = \emptyset$. For each Ap=(Vi,Vj) either $Vi \in P$ and $Vj \in T$, or $Vi \in T$ and $Vj \in P$ "

The Petri net graphs are essential for model construction and analysis, while its execution necissates special notation, that's marking of a Petri net. Marked Petri net is represented by assigning tokens to places. The marking function is defined as mapping from the set of places P to a set of nonnegative integers N, $u: P \longrightarrow N$. A marked Petri net structure is defined by M =(C,u)=(P,T,I,O,u). A place on the Petri net graph is represented by a circle, a transition by a line segment and a token by a dot. The rules for execution of a marked Petri net is given by the following definition.

Definition 3: Execution of Petri net

" A transition $Ti \in T$ in a marked Petri net can be enabled if for all $Pj \in P$, $u(Pj) > \#(Pj , I(Ti))$. This transition may be fired and resulting new marking u´ defined by

$$u'(Pj) = u(Pj) - \#(Pj , I(Ti)) + \#(Pj , O(Ti)) \text{ "}$$

The structure of a Petri net, as it has been shown, can be useful for modeling a sequence of events, which are controlled by preconditions. An event can be presented by a transition and the precondition by a place. An event can be activated when the transition representing it is enabled.The set of all conditions necessary for starting the execution of an event are modeled by the preceding places. A marked place means that the represented condition is true. This may arrive due to

i= a preceding event is terminated

ii= a requested resource is available

Note : It's important to distinguish between an external event as defined by the real time process, and an event (it's in fact an internal event) defined by a transition of a

Petri net model. This will be more comprehensive in the following section.

The firing of any enabled transition changes the marking of the net, and reconfigurate its state. The given definitions can be used to construct an abstract model which is useful for analysis, but not enough for putting the obtained Petri net to work [6]. A more practical representation is by an interpreted Petri net. The complete interpretation is performed by adding the details about the real operating environements of modeled system. These details are basicaly

- the condition imposed by the occurence of external events on the firing of a transition, and
- the precised activity to be accompanied with the firing of a transition

This leads to the following definition

Definition 4: Interpreted Petri net

" In an interpreted Petri net, a transition Ti ∈ T may be labeled by (Ci ; Ai), where Ci is a logic function of external events occurence, and Ai is a precised activity accompanying the firing of the transition Ti. An enabled transition Ti can be fired if and only if Ci is true. An activity Ai can start if and only if Ti is fired"

It should be noted that any transition Ti of an iterpreted Petri net is not necessarily labled by both Ci and Ai. Instead, either it can be labeled by only one of them, or not labeled at all. This depends upon the actual role of the transition in the model.

3. Modeling Procedure

A constructed model of the process image must constitute all alternative states exactly as in the real process. It must also be able to reflect the routing between the states in the sequence which is corresponding to the real operating environements of the process. This routing sequence depends upon the actual arrived events. Therefore the transition from a current state to a new state depends on two conditions

- the new state must be one of the possible successors to the current state
- the arrived external events select this new state

Also, a set of activities may be switched by the current state to produce the required response. Since the required response is dependent on the simultaneous arrived external events, then the latter must specify which activity to be swithed on. These facts form together the basic structure of the required model, which may be demonstrated by figure 2.

It should be noticed that the selected next state is the new current state. Therefore, the natural sequence necissates that the switching of the selected activity precedes the next state which initiates a new sequence.

It's evident that the structure of the interpreted Petri is adequate for modeling the described system. The state can be represented by a set of marked places. The switching from a state to another can be modeled by the labeled transitions, where the label may be formed by a logic condition dependent on the external events, and the corresponding activity to be switched (as stated in the previous section). The illustrated basic structure in figure 2, can be represented by the sample Petri net of figure 3. The model is not incluing the details of the processing modules, but it's obvious that each module

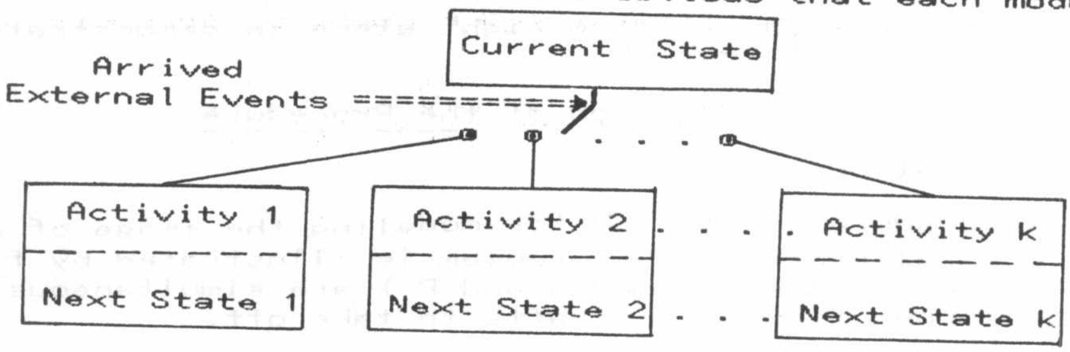


fig.2 The Basic Structure

must correspond to the operations acted by one activity. This is more attendant during the implementation phase. However, a model resulting by this method, is suitable for the top-down design procedure [4].

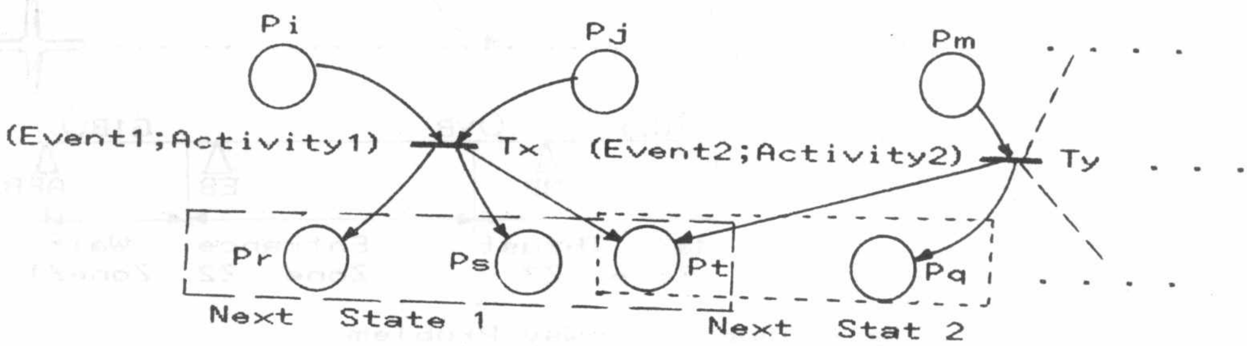


fig.3 Sample Petri net of Basic structure

The hierarchical procedure for modeling an image of a given real time process is described as follows

a. From the given specifications, determine the set of partial processes which can operate seperately
b. Determine the initial state of each partial process
c. Determine the subset of external events which can change the states of each partial process and the accompanied activity with each change
d. Construct a submodel for each partial process using suitable Petri net. Interpret the transitions with the corresponding entities obtained by the step (c)
e. Determine the required interconnection between the different partial processes. Realize this interconnection on submodels by the necessary places and interpreted transitions
f. Repeat step (e) until all described functional requirements are adopted on the model

g. Check the existence of any emergency state which may prevent the normal processing. For such a state add an activity for announcing the emergency state (e.g. alarm or message)
h. The complete needed model is obtained

The application of these eight steps is demonstrated by an example in the next section.

4. Application of the Procedure

4.1 Specifications

The procedure is applied for modeling the image of a double runway control process. The problem is illustrated by figure 4, in which two runways (named A and B) are simultaneously used by a continuous flow of aircrafts to take-off.

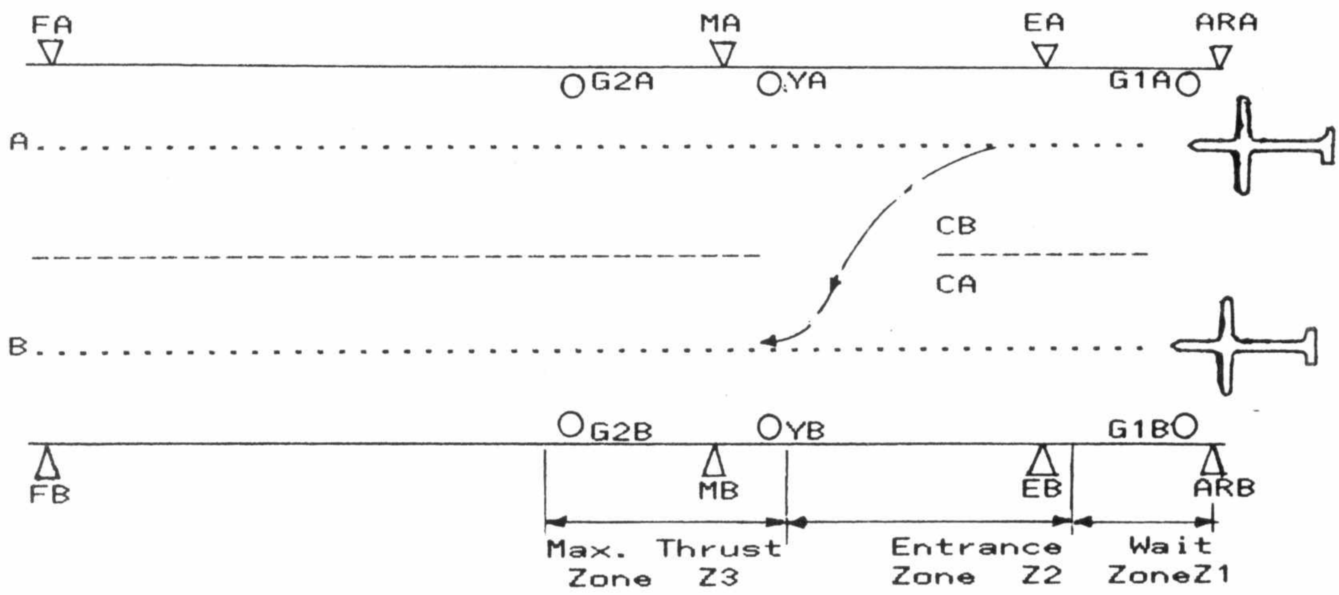


fig.4 Double Runway Problem

The arrival of an aircraft is sensed by a sensor ARA (or ARB) placed in the waiting zone Z1. If the entrance zone Z2 is free, the green indicator G1A (G1B) is set and the aircraft is permitted to pass to Z2, otherwise it must wait in Z1. The entrance of an aircraft to Z2 is sensed by a sensor EA(EB) and the green indicator G1A (G1B) is reset. The aircraft can enter to the maximum thrust zone Z3 only if it's free and the runway is in normal use, otherwise a yellow indicator YA (YB) is set and the aircraft waits in Z2. The entrance to Z3 is sensed by a sensor MA (MB). If the runway is blocked for any reason, a blocking signal BLA (BLB) is generated, and YA (or YB) is set. When the runway is available a green indicator G2A (G2B) is set and the aircraft can perform the take-off. The runway is free when the aircraft takes-off, which is sensed by the sensor FA (or FB) placed at the end of the runway.

In order to allow the take-off operation to continue, if a runway is blocked, the aircraft can use the other runway under two conditions

i- neither this alternative runway is blocked
ii- nor an aircraft is waiting at its entrance

When these two conditions are true, then an indicator lamp (CB or CA) is set, to permit the aircraft to transfer to the other runway (B or A).

4.2 Construction of submodels

The presented process constitutes two seperate and symetric partial processes. Each partial process is related to a runway, then they are named "process A" and "process B". The different possible states are

S0: Z1 is ready to receive an aircraft and Z2 is available
S1: an aircraft passes through Z2 and Z1 becomes unavailable
S2: an aircraft enters Z3, Z1 becomes available again and Z3 becomes unavailable
S3: an aircraft has taken-off and Z3 becomes available again

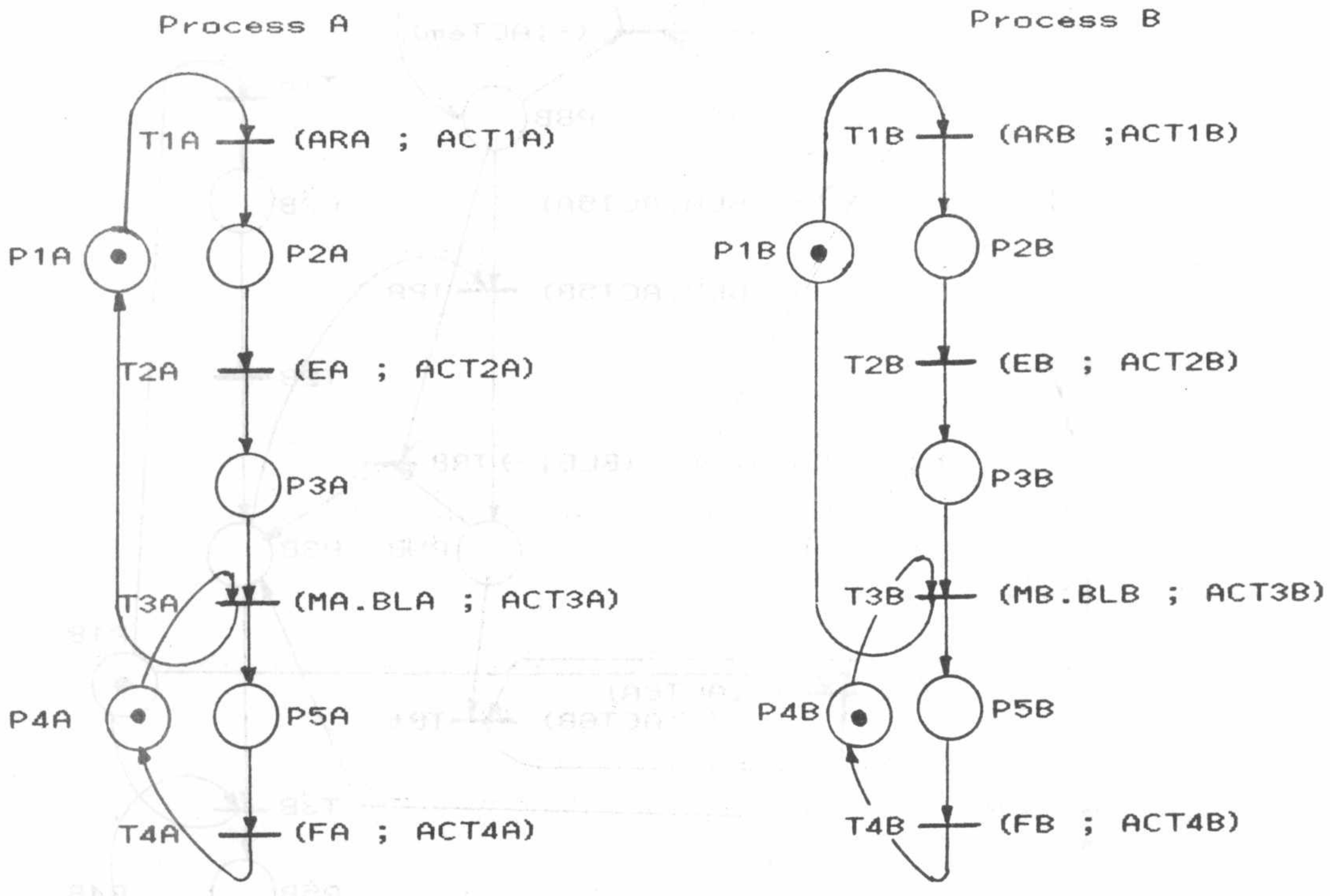


fig.5 Two submodels for processes A and B

It is clear that four transitions between the four states are needed. In figure 5, the four transitions for each process and their iterpretations are shown. The states are represented for process A as following (process B is symetric)

S0: P1A,P4A S1: P2A,P4A S3: P3A,P4A S4: P1A,P5A

The activities triggered by the transitions are as follows

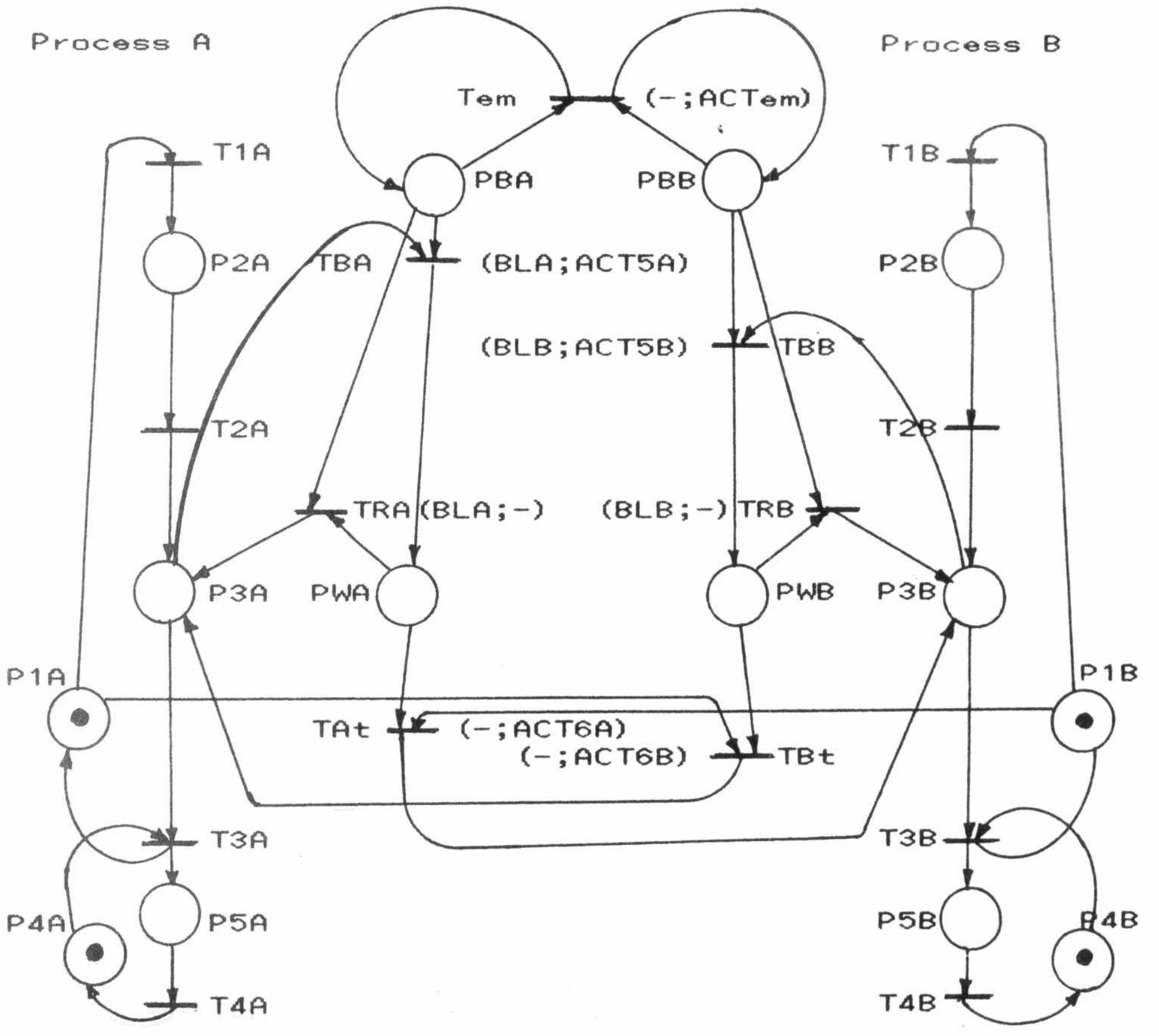
ACT1A: Reset G1A ACT2A: No Operation
ACT3A: Set YA and Set G1A ACT4A: Reset YA and Set G2A

4.3 The interconnection

Since an aircraft on a blocked runway can use the other runway, then process A can transform to complete on process B when the runway A is blocked. A waiting aircraft in the zone Z2 of the runway A, can transfer to the zone Z3 of the runway B, when the two mentioned conditions are true. It is detected during state S2. Two transitions and two places are added for performing this transformation. The process B has exactly the same modifications. The resulting model is given on figure **6**. Note that the arcs incident from P1B (P1A) to the transition TAt (TBt) represents the two conditions (mentioned in 4.1) imposed on the transformation from an airway to another. The corresponding activities are as follows

ACT5A: Set YA

ACT6A: Set CB



Note: The iterpretation of new transitions only are written the others are exactly the same as in figure **5.**

fig.**6** The complete model

4.4 Emergency state

In the demonstrated process, the processing may completely stop when the two runways are blocked. This is represented on the model by the state in which the two places PBA and PBB are both marked. In this case the two runways can not be used, and this must be announced as an emergency state. A transition Tem is added to the model (fig. 5) to represent this state. This transition enabled by the two places (PBA;PBB), and it's fired independent of external events. The looping arcs cause repeated firing of the transition, and consequently the announcement of the emergency state is continuously transmited until at least one runway becomes available again. The normal processing can be resumed by eleminating the token from PBA (or PBB) by the transition TAt (or TBt). The activities triggered by the two transitions Tem and TAt (or TBt) are as follows

ACT6A : Reset YA

ACTem : Transmit Emergency Message and Alarm

CONCLUSION

The systematic hierarchical modeling procedure leads to a complete and well conformed model. The Petri net is a suitable modeling tool for processes which can be integrated from a set of seperate partial processes. Their concurrent processing and interconnection can be easily represented on the final model. Moreover, a Petri net model can be casted directly to programs [5][6], which represents the image of the process. The set of activities switched by the transitions of the resulting model, can be transformed to structured programs modules. Generally, the given modeling procedure can be applied for any real-time control process as a basic step of the top-down design technic.

REFERENCES

[1] Vick, C.R., "Dynamic Resource Allocation in Distributed Computing Systems", UMI Rsearch Press, 1980

[2] Fraley, B., "Design of Realtime systems", Proceeding on Software Design Engineering , October, 1976, pp 57-59

[3] Gaulding, S.N. and Lawson, J.D., "Process Design Engineering, a Methodlogy for Real-time Software Development",IEEE Transaction on Computer, 1985, pp 80-85

[4] Freedman, M.D. and Evans L.B., "Designing Systems with Microcomputers, A Systematic Approach", Prentice-Hall,1983

[5] Nelson, R.A., Haibt, L.M., and Sheridan P.B., "Casting Petri Nets into Programs", IEEE Transaction on Software Engineering, vol. SE-9, 1983, pp 890-602

[6] Agerwala, T., "Putting Petri Nets to Work", IEEE Transaction on Computer, December 1979, pp 85-94

[7] Mustafa, M.S., "Conception et Realisation d'un Automate Programmable par Schemas a Reseaux de Petri",Ph.D. Thesis Paul Sabatier University, Toulouse, France, 1980

[8] Peterson, J.I., "Petri Net Theory and The Modeling of Systems", Prentice-Hall, 1981

[9] Valette, R. and Courvoisier M., "Systeme de Commande en Temps Reel", SCM, Paris, 1980