Validation du comportement dynamique des architectures de conduite des systèmes de production par simulation

Pascal Meunier, Bruno Denis

Validation du comportement dynamique des architectures de conduite des systèmes de production par simulation

Pascal Meunier — Bruno Denis
Laboratoire Universitaire de Recherche en Production Automatisée
Ecole Normale Supérieure de Cachan
61 avenue du Président Wilson
94235 Cachan Cedex
{meunier,denis}@lurpa.ens-cachan.fr

RÉSUMÉ : Du fait de leur complexité croissante, il devient difficile de valider analytiquement les systèmes durant leur phase de conception. La simulation devient l'un des outils privilégiés des concepteurs. La conception des architectures des conduite des systèmes de production n'échappe pas à cette règle. Dans cette communication, nous présenterons la démarche qui nous a conduits à choisir un simulateur, et les résultats de simulation obtenus sur un exemple significatif de conception d'architecture ainsi que les bénéfices que peut en tirer le concepteur.

ABSTRACT: At the design stage, it's often difficult to assess systems features, because of their complexity. Simulation becomes ones of the best tool to help designers. This paper deals with the design of the control architecture of production system. We present how we have managed the choiced of a simulation tool. Then, after a presentation of an illustrative example, results of a simulation are given.

1. Introduction

Lors de la conception du système de conduite des systèmes de production, il est une phase qui s'intègre entre la spécification de l'application de conduite, et la programmation de l'application dans les équipements de commande : il s'agit de la conception de l'architecture de conduite. La conception d'architecture consiste à partitionner puis répartir une spécification sur une architecture matérielle formée d'automates program-

mables, de calculateurs industriels, de réseaux de terrain, etc. Cette projection d'une partition des spécifications sur une architecture matérielle est appelée architecture opérationnelle. Pour valider ses choix, le concepteur architecte doit pouvoir évaluer les performances, les contraintes de sûreté et de coût. Aujourd'hui, les outils des concepteurs pour valider les architectures opérationnelles sont quasiment inexistants [le Quenven, 95]. Pour pouvoir proposer de tels outils nous avons dans un premier temps formalisé les principales étapes de la conception d'architecture [Denis, 94] [Denis et al., 95], puis nous avons modélisé le comportement dynamique des architectures opérationnelles. A l'aide des modèles de comportement dynamique que nous avons proposés, l'utilisation d'un outil de simulation permettrait à l'architecte de "visualiser" les effets de ces choix sur les performances de l'architecture opérationnelle, comme par exemple des temps de réponses. Notre objectif n'étant pas a priori de créer un outil spécifique de simulation, nous nous sommes mis à la recherche d'un outil existant sur le marché ou dans des laboratoires universitaires.
Dans la section 2 nous exposons notre démarche de choix d'un simulateur en concordance avec nos besoins. Puis, dans la section 3 nous présentons une première utilisation en conception d'architecture de l'outil de simulation retenu.

2. Simulateurs de réseaux de Petri colorés et temporisés

2.1. Objectifs de la conception d'architecture et besoins en simulation

La conception d'architecture de conduite a pour objectifs la détermination d'une architecture matérielle apte à accueillir l'architecture fonctionnelle conformément à un ensemble de contraintes opérationnelles telles que la répartition géographique du processus, ou les temps de réponse attendus. L'architecture matérielle est un ensemble d'équipements d'informatique industrielle interconnectés entre eux par des réseaux de communication (fig. 1b). Issue des spécifications fonctionnelles, l'architecture fonctionnelle (fig. 1a) regroupe l'ensemble des traitements qui devront être *projetés* dans l'architecture matérielle et l'ensemble de données consommées et produites. Une typologie des traitements et des équipements est proposée dans [Denis, 94].

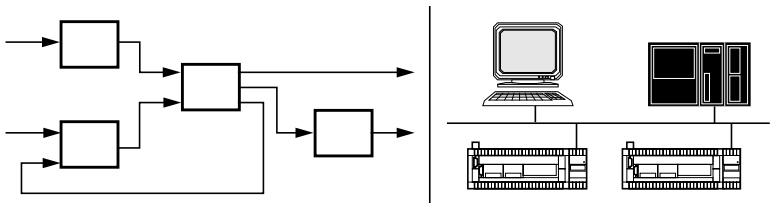
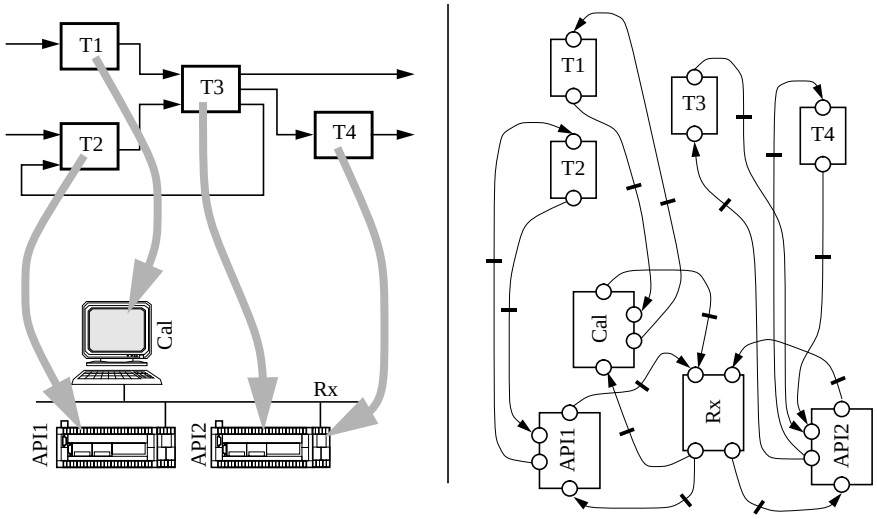


Figure 1. *Architecture fonctionnelle et architecture matérielle*

Fréquemment, l'architecture matérielle est présentée comme le résultat de la conception d'architecture. Mais c'est en fait la projection de l'architecture fonctionnelle dans

l'architecture matérielle (architecture opérationnelle, voir la fig. 2a) qui est le résultat complet de la conception. Pour valider les architectures opérationnelles nous avons proposé de construire un modèle de comportement dynamique par assemblage de modèles génériques de comportement des équipements et des types de traitements [Denis, 94] (fig. 2b).



a - architecture opérationnelle b - modèle de comportement dynamique

Figure 2. *Résultat attendu de la conception d'architecture, et son modèle de comportement*

Parmi les techniques de modélisation permettant la description de comportements dynamiques, nous avons choisi les réseaux de Petri colorés et temporisé. La coloration nous permet de faire des modèles génériques pour lesquels la structure du graphe (places, transitions et arcs) est indépendante de l'architecture. Par exemple la structure d'un modèle dynamique d'un réseau de communication doit être indépendante du nombre d'équipements qui lui est connecté. Seuls les paramètres de coloration doivent permettre d'adapter le modèle à l'architecture modélisée. La temporisation est pour sa part indispensable pour évaluer les caractéristiques temporelles des architectures. L'ensemble des modèles modulaires a été établi en utilisant les réseaux de Petri colorés au sens de l'ouvrage de David et Alla [David et Alla, 92].

2.2. Les critères de choix d'un simulateur

Partant de notre modélisation du comportement dynamique d'une architecture opérationnelle, un simulateur de réseaux de Petri colorés et temporisés permettra au concepteur d'obtenir des résultats quantitatifs sur ses architectures. Cependant, il s'est avéré que la notion de coloration est très variable d'un outil à l'autre. Pour pouvoir les comparer, nous avons mis en évidence trois niveaux de coloration ayant une incidence directe sur les potentialités de modélisation. La figure 3a présente un exemple de réseau de Petri ordinaire. Pour modéliser plusieurs états d'un système avec une seule place, il suffit de

disposer de la coloration des jetons, il s'agit d'un premier niveau de coloration. La figure 3a montre comment les états modélisés par les trois places P_{cyan}, P_{jaune}, et $P_{magenta}$ (resp. P_{vert} et P_{rouge}) de la figure 3a peuvent être regroupés dans une seule place $P_{couleur\_base}$ (resp. $P_{couleur\_composée}$), après avoir introduit les couleurs cyan, jaune, et magenta (resp. rouge et vert). La pondération des arcs assure la cohérence des transitions.

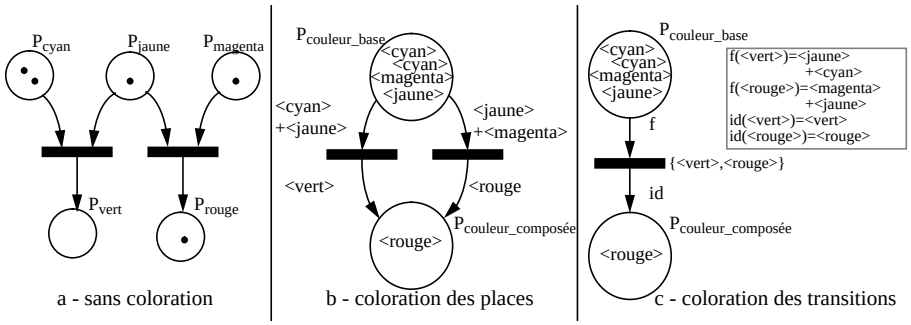


Figure 3. *Différents niveaux de coloration (syntaxe utilisée : [David et Alla, 92])*

On voit que si la coloration des jetons permet une plus grande richesse de modélisation pour les places, les transitions sont quant à elles inchangées. Un second niveau de coloration permet le regroupement de plusieurs évolutions d'un système dans une seule transition, il suffit de particulariser le franchissement des transitions avec des couleurs. La figure 3c montre comment les deux transitions de la figure 3b peuvent être regroupées en une seule, après avoir introduit une pondération des arcs qui s'exprime cette fois-ci en fonction des couleurs qui particularisent la transition. Seule l'utilisation de ces deux niveaux de prise en compte de la coloration permet la construction de modèles génériques pour lesquels la structure du graphe (places, transitions et arcs) est indépendante de la structure des architectures de conduite. Ces deux niveaux sont donc requis pour l'outil de simulation que nous recherchons. Un troisième niveau de la coloration se situe dans la capacité à exprimer des couleurs complexes de jetons sous la forme d'une liste de champs (exemple : <3, bleu, petit, …>). Cela donne une grande souplesse dans l'expression de la pondération des arcs et de la particularisation des transitions. Un ensemble de couleur, ou une pondération peuvent ainsi s'exprimer de façon simple en compréhension, ce qui évite les expressions fastidieuses en extension. Ce troisième niveau de coloration peut être utilisé avec ou bien sans le second niveau. Il a été utilisé pour la modélisation de l'architecture, ainsi sa présence dans l'outil de simulation éviterait la transcription laborieuse des modèles.

Concernant la temporisation, nous avons utilisé dans nos modèles des temporisations déterministes associées aux places. Cependant, il n'est pas très contraignant de passer d'un réseau P-temporisé à un réseau T-temporisé (temporisation associée au transition). La simple existence de temporisations déterministes dans un outil de simulation pouvant répondre à nos besoins.

2.3. Les outils de Simulation de réseaux de Petri colorés et temporisés

Nous avons mené une recherche bibliographique pour identifier les outils existants de simulation et leurs capacités de modélisation et de simulation. A l'aide de la base de données du serveur WWW du DAIMI de l'université d'Aarhus au Danemark, nous avons isolé dix outils aptes à prendre en considération la coloration et la temporisation des modèles. Il n'a pas été possible pour tous ces outils de trouver une documentation pertinente qui permet de juger de leurs capacités. Seuls quatre des outils ont pu être étudiés en détail, soit à l'aide de leur manuel, soit à l'aide de communications scientifiques dont ils ont fait l'objet (tableau 1).

Tableau 1 : *Outils de simulation de réseaux de Petri colorés et temporisés*

Pas de littérature technique disponible	**littérature technique disponible**	
	sous forme de manuels	**sous forme de manuels et de communications scientifiques**
Alpha Sim (USA) Exspect (Pays Bas) INA (Allemagne) Macrotec (Canada) Pntblsim (USA) Tempro (USA)	Poses ++ (Allemagne) [POS, 96] Visual Simnet (Allemagne) [Garbe, 95]	Design/CPN (Danemark) [DES, 93] [Jensen, 92] XSimnet (Finlande) [Gustavson et Torn, 95] [Gustavson et Torn, 94]

2.4. Choix d'un outil de simulation

Le tableau 2 présente le résultat de notre analyse bibliographique en mettant en relation nos besoins et les capacités des outils. Seuls deux outils sont capables de mettre en œuvre la coloration à la hauteur de nos besoins : Design/CPN, et Poses++. C'est le caractère modulaire de la modélisation dans Design/CPN, l'existance de son éditeur graphique et son ergonomie qui l'ont emportés pour notre choix final. De plus Design/CPN est porté par une importante activité scientifique sur la formalisation des réseaux de Petri de haut niveau qui de plus sont à la base des travaux de normalisation du groupe de travail ISO/IEC JTC1/SC7/WG11 de l'ISO.

Tableau 2 : *Aptitude des outils de simulation à répondre à nos besoins*

	coloration des transitions	**coloration complexe**	**temporisation**	**modularité**	**editeur graphique**
Design/CPN	oui	oui	spécifique	oui	oui
Poses++	oui	oui	P-temporisé et T-temporisé	non	non
Visual simnet	non	oui	P-temporisé	non	oui
XSimnet	oui mais pondération des arcs limitée	non	spécifique	oui	non

3. Etude d'une conception d'architecture

3.1. Architecture fonctionnelle

Comme exemple support nous avons choisi une architecture fonctionnelle constituée de trois traitements réactifs (T1, T2, et T3). Sa structure est donnée dans la figure 4a. Une spécification partielle en Grafcet de chaque traitement met en évidence l'aspect du comportement du système sur lequel va porter l'analyse du concepteur. A l'apparition de l'entrée dcy provenant d'un pupitre, le traitement T3 émet deux sorties dm1 et dm2 qui à leur tour sont les entrées respectivement pour les traitements T1 et T2. T1 émet pour un pré-actionneur la sortie AD1 sur occurrence de dm1, et T2 émet la sortie AD2 à la vue de dm1.

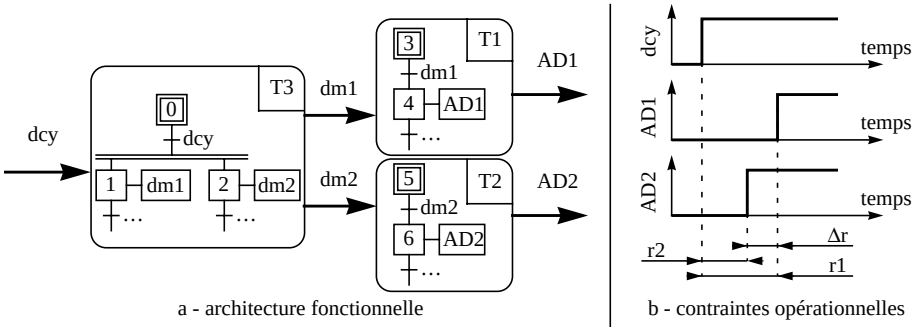
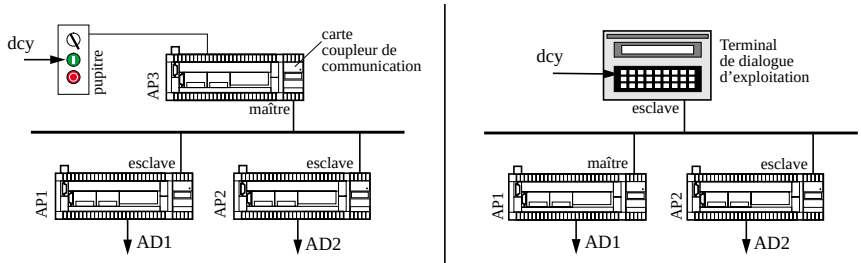


Figure 4. *Architecture fonctionnelle*

Les chronogrammes de la figure 4b mettent en place les temps de réponse que souhaite évaluer l'architecte pour les architectures opérationnelles qu'il va concevoir. Il s'agit des deux retards r1 et r2 qui s'écoulent entre l'apparition de l'entrée dcy et l'émission des commandes AD1 et AD2, et de l'écart temporel qu'il peut apparaître entre l'émission des deux commandes AD1 et AD2.

3.2. Architectures matérielles

Pour des raisons de répartition géographique des pré-actionneurs AD1 et AD2 et de l'entrée dcy, l'architecte est conduit à proposer deux architectures matérielles bâties autour d'un réseau d'automate et de trois équipements. Dans la configuration de la figure 5a, un automate programmable traite chacun des signaux dcy, AD1 et AD2. Dans cette configuration dcy est une entrée de l'automate AP3 qui provient d'un pupitre constitué de bouton poussoir. Le réseau d'automates (de type Profibus-DP ou UNI-TE) est basé sur un protocole d'accès maître-esclave. L'automate qui dispose de l'information dcy a été choisi comme maître sur le réseau. Dans la configuration de la figure 5b, un terminal d'exploitation remplace le pupitre et seulement deux automates sont utilisés. Les terminaux d'exploitation ne peuvent généralement pas assurer le rôle de maître sur le réseau d'automate. L'automate API1 assure cette fonction dans cette deuxième architecture matérielle.



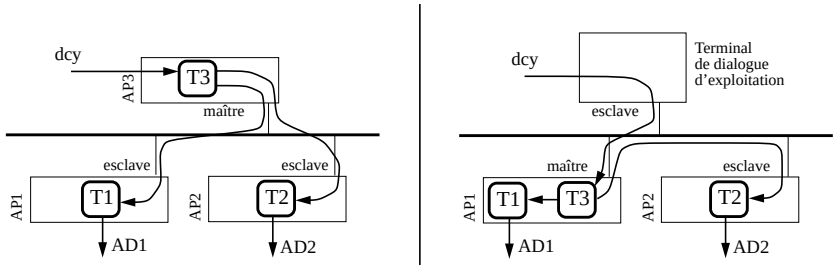
a - Configuration 3 API

b - Configuration 2 API + 1 terminal

Figure 5. *Architectures matérielles pressenties*

3.3. Architectures opérationnelles

Pour chaque architecture opérationnelle l'architecte projette d'une ou de plusieurs manières l'architecture fonctionnelle sur l'architecture matérielle. Cela consiste en la répartition des traitements dans les équipements de commande. Deux projections sont proposées. L'une est transcrite figure 6a et correspond à la projection de l'architecture fonctionnelle sur la configuration à trois automates de la figure 5a. L'autre représentée figure 6b est une projection sur la configuration à deux automates et un terminal de la figure 5b.



a - Configuration 3 API

b - Configuration 2 API + 1 terminal

Figure 6. *Architectures opérationnelles pressenties*

Le moment du choix entre ces deux architectures opérationnelles est arrivé. L'architecte doit décider laquelle des deux répond le mieux à ces exigences. En particulier ces exigences liées aux contraintes temporelles de la figure 4b. Pour l'assister nous allons construire un modèle simulable de ses deux architectures, puis nous allons simuler leur comportement respectif.

3.4. Modélisation de la dynamique de l'architecture

Après avoir construit des modèles génériques de chaque équipement (voir un exemple de modèle de réseaux d'automate en annexe A), chaque module doit être connecté conformément à l'architecture opérationnelle modélisée. L'outils Design/CPN permet la

"fusion" des places entre elles. Cela signifie qu'une place peut être dessinée dans plusieurs modules (sans lien graphique apparent) et ne former qu'une seule et même entité. Il n'est donc pas nécessaire de relier par des arcs les différents modules. Il suffit de fusionner les places d'entrée de certains modules aux places de sortie d'autres modules. La figure 7 montre l'ensemble des modules du modèle de l'architecture opérationnelle

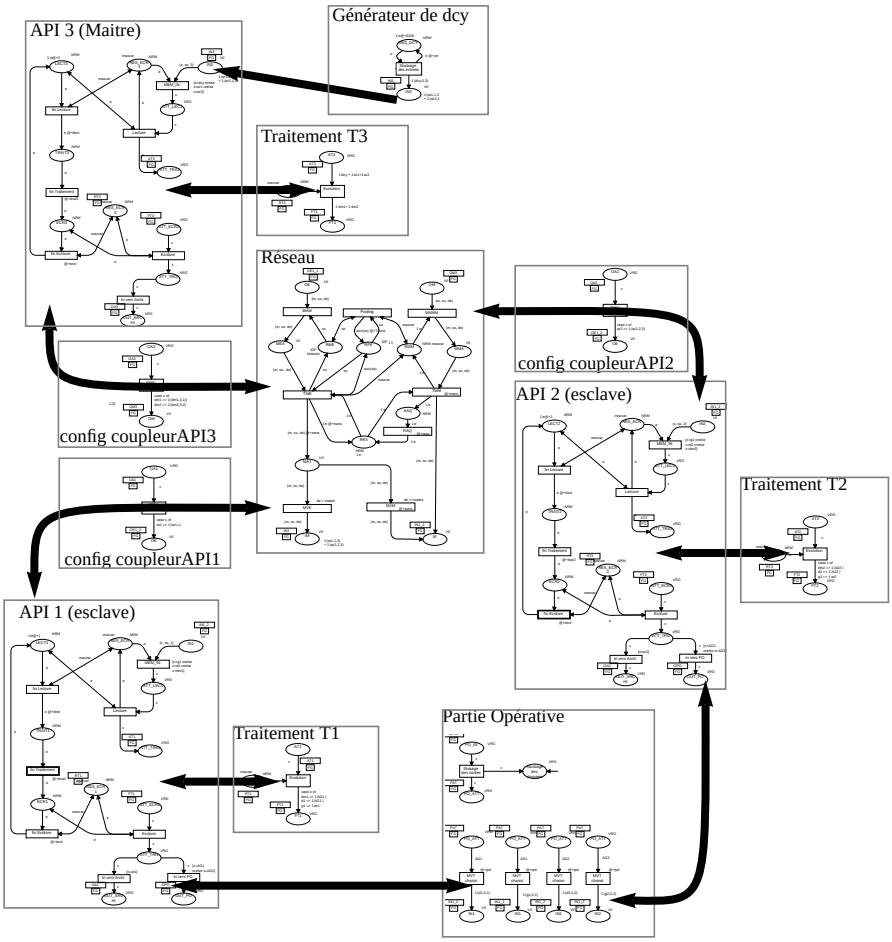


Figure 7. *Modélisation dynamique de l'architecture opérationnelle réalisée avec Design/CPN*

3.5. Simulations du comportement des deux architectures opérationnelles

Maintenant que le modèle de simulation est en place, il faut définir un scénario d'excitation du modèle et les moyens à mettre en œuvre pour observer les indicateurs r1 r2 et Δr. Un générateur d'entrée dcy est ajouté au modèle (un place et une transition temporisée) de manière à générer un signal dcy toutes les dix secondes. Afin de collecter les résultats de simulation, des jetons représentatifs (images de dcy, d'AD1 et AD2) sont accumulés

et datés dans une place prévue à cet effet. Le contenu de cette place est alors exporté sous la forme d'un tableau de valeur vers un logiciel nous servant de grapheur (Matlab). Deux mille entrées dcy ont été générées pour chacun des modèles d'architecture opérationnelle. Les répartitions des retards r1, r2 et Δr pour les deux configurations sont reportés sur la figure 8. L'architecte est maintenant en mesure de juger des effets de ces choix d'architecture. En particulier, la configuration à deux automates admet des bornes supérieures aux retards r1 et r2 plus favobles que la configuration à trois automates, mais au détriment du retard entre les deux ordres AD1 et AD2.

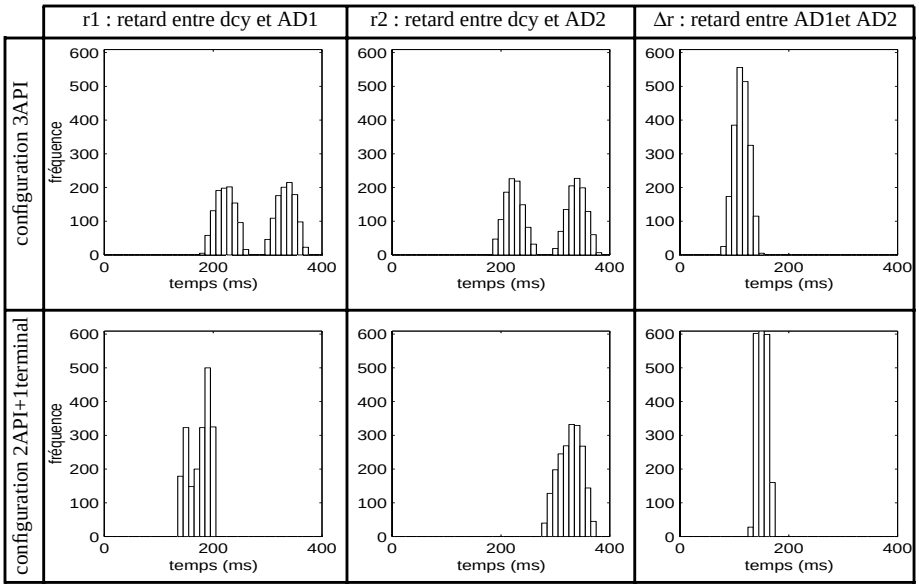


Figure 8. *Résultats de la simulation*

4. Conclusions

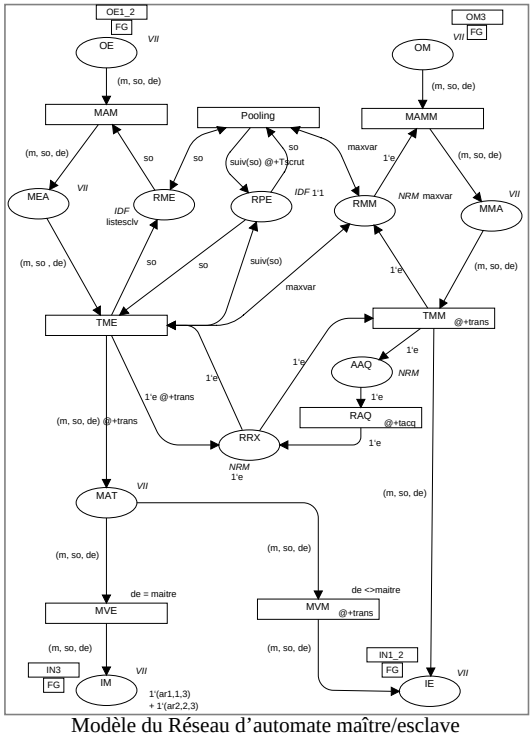
Le choix d'un outil de simulation c'est avéré plus délicat que nous ne l'avions imaginé. Nos besoins en modélisation ont conduit notre choix sur Design/CPN. Cette première utilisation de l'outil pour la simulation du comportement dynamique d'une architecture de conduite est concluante. Nos effort vont se porter à la fois sur la mise au point des modèles génériques par la confrontation avec le comportement de constituants réels en site, et sur la mise en place de scénarii d'excitation plus élaborés.

5. Bibliographie

[DES, 93] *DESIGN/CPN Tutorial for x-windows*. META SOFTWARE CORPORATION, 125 Cambridgepark drive, CAMBRIDGE, MA 02140 U.S.A.

[POS, 96] *POSES ++ Software tool*. GPC mbh, GPC Gessellschaft fur ProzeBautomation & Consulting mbH.
[David et Alla, 92] R. David et H. Alla. *Du GRAFCET aux Réseaux de Petri*. Traité des nouvelles technologies, Série automatique. Hermès, Paris, 2ème édition, 1992.
[Denis, 94] B. Denis. *Assistance à la conception et à l'évaluation de l'architecture de conduite des systèmes de production complexes*. Mémoire de thèse, Université Nancy I, 1994.
[Denis et al., 95] B. Denis, J. Lesage, et J. Roussel. 'A Method for Design and Valuation of Manufacturing System Control Architecture'. *1995 IEEE INTERNATIONAL CONFERENCE ON SYSTEMS, MAN AND CYBERNETICS*, pp. 4486–4491, Vancouver, British Columbia, Canada. 0ct 22-25 1995.
[Garbe, 95] W. Garbe. *Stochastic Petri-net simulator Visual Simnet 1.34*. Wachendorffstr.1 40882 Ratingen Germany.
[Gustavson et Torn, 94] A. Gustavson et A. Torn. 'XSIMNET, A tool in C++ for executing simulation nets'. *Conference on Modeling and Simulation*, pp. 146–150. 1994.
[Gustavson et Torn, 95] A. Gustavson et A. Torn. 'OO tokens and sensor arcs in th tool XSIMNET'. *IMACS European Simulation Meeting*. August 1995.
[Jensen, 92] K. Jensen. *Coloured petri nets, basic concepts, analysis methods and practical use*, vol. 1. SPINGER-VERLAG, 1992.
[le Quenven, 95] T. le Quenven. 'Dimensionnement d'architecture : approche d'un utilisateur'. *Journées d'études SAPID*, Paris. 30–31 Mai 1995.

Annexe A. Extraits du modèle de simulation



Modèle du Réseau d'automate maître/esclave

```
val trans = 103;
val Tscrut = 23;
val maxvar = 5`e;
val listesclv = 1`1 + 1`2;
val tacq = 11;
val nesclvmax = 2;
val maitre = 3;
val idmax = 3;

color NRM = with e timed;
color IDF = int with 0..idmax timed;
color VRG = with dcy|dm1|dm2|AD1|AD2 timed;
color VII = product VRG*IDF*IDF;

fun suiv (so) =
      if so = nesclvmax then
            1`1
      else
            1`(so+1);

var m : VRG;
var so, de : IDF;
```

Déclaration des constantes des couleurs des fonctions et des variables