N° d'ordre : 203

THESE

50376
1988
23

présentée à

L'UNIVERSITE DES SCIENCES ET TECHNIQUES DE LILLE FLANDRES ARTOIS

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITE

en

PRODUCTIQUE, AUTOMATIQUE ET INFORMATIQUE INDUSTRIELLE

par

Jean-Pierre BOUREY

STRUCTURATION DE LA PARTIE PROCEDURALE DU SYSTEME DE COMMANDE DE CELLULES DE PRODUCTION FLEXIBLES DANS L'INDUSTRIE MANUFACTURIERE

soutenue le 23 Mars1988 devant la Commission d'Examen

Membres du Jury :	M. COURVOISIER	Rapporteur
	M. TOULOTTE	Rapporteur
	M. SCHMIDT	Examinateur
	M. SILVA	Examinateur
	M. GENTINA	Examinateur, Directeur de Thèse
	M. CORBEEL	Examinateur
	M. THUREL	Invité

SOMMAIRE

INTRODUCTION GENERALE page 1

CHAPITRE I page 7

CHAPITRE II page 43

CHAPITRE III page 77

CHAPITRE IV page 145

CONCLUSION GENERALE page 189

BIBLIOGRAPHIE page 195

CHAPITRE I

pages

INTRODUCTION 9

I - MODELISATION DE LA PARTIE COMMANDE 13
I.1 - Introduction 13
I.2 - Les réseaux de Petri structurés 13
I.2.1 - Introduction 13
I.2.2 - Le modèle 14
a) Représentation modulaire : les graphes de processus 14
b) Interactions entre processus : les graphes de liaisons 17
c) Localisation des liaisons 22
I.3 - Extensions des réseaux de Petri structurés 23
I.3.1 - Les réseaux de Petri structurés adaptatifs 23
a) Introduction 23
b) Applications à la modélisation 23
c) Conclusion 26
I.3.2 - Les réseaux de Petri structurés adaptatifs colorés 26
a) Introduction 26
b) Coloration en productique 26
c) Extension aux réseaux de Petri structurés adaptatifs 29
I.4 - Conclusion 29

II - MODELISATION DU NIVEAU HIERARCHIQUE 31
II.1 - Introduction 31
II.2 - Choix du modèle 32
II.3 - Description des règles et de leurs interfaces 33
II.4 - Conclusion 35

III - MODELISATION DU PROCEDE 37
III.1 - Introduction 37
III.2 - Les réseaux de Petri temporisés 38

CONCLUSION 41

CHAPITRE II

pages

INTRODUCTION 45

I - DESCRIPTION DE LA DEMARCHE 47
I.1 - Introduction 47
I.2 - Le schéma directeur général 48
I.3 - Le schéma directeur détaillé 49
I.3.1 - La phase de spécifications 51
I.3.2 - La phase de modélisation 54
a) Modélisation de la partie commande 54
b) Modélisation du niveau hiérarchique 54
c) Modélisation du procédé 55
I.3.3 - La simulation 56
I.3.4 - L'implantation 57
I.4 - Conclusion 58

II - LE PREGRAPHE 59
II.1 - Introduction 59
II.2 - Description des règles 59
II.3 - Traduction en réseau de Petri coloré 60
II.4 - Conclusion 68

III - LE SIMULATEUR 69
III.1 - Introduction 69
III.2 - Intérêt d'une approche déclarative 70
III.3 - Description du simulateur 71
III.3.1 - Description générale 71
III.3.2 - Configuration du simulateur 72
III.3.3 - Cycle de simulation 72
III.4 - Conclusion 73

CONCLUSION 75

CHAPITRE III

pages

INTRODUCTION 79

I - PRESENTATION GENERALE DU LOGICIEL 81

II - STRUCTURATION DU PREGRAPHE 83
II.1 - Introduction 83
II.2 - Principe de structuration utilisé 84
II.3 - Description des primitives de structuration 87
II.3.1 - Notations 87
II.3.2 - Les zones de stockage intermédiaire 87
II.3.3 - Les transferts 90
II.3.4 - Les machines 95
a) Les machines sans tampon d'entrée/sortie 97
b) Les machines à un tampon d'entrée/sortie 98
c) Les machines à plusieurs tampons d'entrée/sortie équivalents 99
d) Les machines à un tampon d'entrée et un tampon de sortie 100
II.3.5 - Les assemblages et les désassemblages 101
a) Les palettisations 102
b) Les dépalettisations 110
II.3.6 - Les positionnements 115
a) Les positionnements internes 117
b) Les positionnements externes 118
II.3.7 - Conclusion 120

III - REPARTITION DES SYSTEMES DE COMMANDE 121
III.1 - Introduction 121
III.2 - Les transferts synchronisés 121
III.3 - Les robots 126
III.3.1 - Introduction 126
III.3.2 - Les robots de transfert 127
III.3.3 - Les robots d'assemblage et de désassemblage, de positionnement 128
III.3.4 - Conclusion 130
III.4 - Conclusion 130

.../...

pages

IV - INTEGRATION DU NIVEAU HIERARCHIQUE ET DU PROCEDE 131
IV.1 - Introduction 131
IV.2 - Intégration du niveau hiérarchique 131
IV.2.1 - Introduction 131
IV.2.2 - Détection des conflits 131
a) Introduction 131
b) Le partage de ressources 133
c) Les répétitives et alternatives multiples 133
d) Les sorties de zones 135
e) Conclusion 136
IV.2.3 - Interfaçage avec la partie commande 137
IV.3 - Intégration du procédé 138
IV.3.1 - Introduction 138
IV.3.2 - Choix des entrées 138
a) Introduction 138
b) Spécification de la fonction d'entrée 139
c) Intégration de la fonction d'entrée au modèle .. 140
d) Conclusion 141
IV.3.3 - Le dimensionnement 141
IV.3.4 - La temporisation 142
IV.4 - Conclusion 142

CONCLUSION 143

CHAPITRE IV

pages

INTRODUCTION 147

I - PRESENTATION DE LA CELLULE INITIALE 149
I.1 - Architecture de la cellule 149
I.2 - Les gammes opératoires 152

II - APPLICATION DE LA METHODOLOGIE C.A.S.P.A.I.M. 159
II.1 - Le prégraphe 159
II.2 - L'élaboration du graphe de commande structuré 162
II.2.1 - Structuration du prégraphe 162
II.2.2 - Répartition des processus de commande 172
II.2.3 - Les règles du niveau hiérarchique 172
II.2.4 - Le dimensionnement et la temporisation 173
II.3 - Simulation du graphe de l'atelier 174
II.3.1 - Introduction 174
II.3.2 - Simulation non temporisée et entrée aléatoire au plus tôt 174
II.3.3 - Simulation temporisée et introduction sélective des pièces 175
II.3.4 - Simulation temporisée et entrée au plus tôt 177
II.3.5 - Conclusion sur la simulation 180

III - PROJET D'EXTENSION DE LA CELLULE INITIALE 181
III.1 - Présentation de la nouvelle architecture 181
III.2 - Elaboration du graphe de commande 184

CONCLUSION 187

INTRODUCTION GENERALE

Les ateliers flexibles sont aujourd'hui devenus une nécessité économique : il faut, en effet, à la fois assurer une production de plus en plus diversifiée et plus sophistiquée et accroître la productivité tout en réduisant les coûts de conception et d'exploitation des systèmes de production.

Pour répondre à ces impératifs, il est alors nécessaire d'exploiter au maximum les matériels automatisés de production (machines-outils à commande numérique, robots de manutentions, convoyeurs, ...).

Cette partie mécanique est bien évidemment la partie la plus "visible" de l'atelier flexible. Sa gestion repose sur le contrôle effectué par un ou plusieurs calculateurs, selon une organisation hiérarchisée ou non. De ce fait, il n'est pas rare de consacrer plus de 50 % du coût des investissements consentis en mécanique pour les systèmes de commande **et** le logiciel. (L'atelier Citroën/ Meudon a demandé 75 000 heures d'études dont 25 000 pour le logiciel) /DEF 85/.

Ainsi, la complexité croissante de tels automatismes, le manque actuel de standardisation tant au niveau des machines que des logiciels font qu'un atelier flexible demande une longue période de gestation avant d'arriver à un fonctionnement optimal.

L'adoption de méthodologies de conception **rigoureuses** et **systématiques** des systèmes de commande est alors à prendre en considération lorsque l'on souhaite réduire tout à la fois les coûts de conception et d'exploitation et les durées de gestation des projets.

Le projet C.A.S.P.A.I.M. (**C**onception **A**ssistée de **S**ystèmes de **P**roduction **A**utomatisée en **I**ndustrie **M**anufacturière), développé au Laboratoire d'Informatique Industrielle de l'I. D. N., constitue un effort de synthèse allant dans ce sens. Ce laboratoire travaille en effet à l'élaboration d'une chaîne modulaire de C.A.O. de systèmes de commande prenant en charge le cycle de conception du projet depuis la définition du cahier des charges jusqu'à la phase finale d'implantation.

Le logiciel de structuration que nous présentons dans ce mémoire permet, au sein de la méthodologie C.A.S.P.A.I.M., de construire la partie procédurale du système de commande. Nous avons donc orienté notre travail afin de satisfaire les contraintes essentielles suivantes :

i) Modularité de l'approche de structuration. Dans ce sens, nous avons construit notre logiciel de façon modulaire afin de prendre en charge facilement des modifications éventuelles des primitives de structuration.

ii) Modularité du graphe de commande engendré afin de prendre en compte facilement des modifications inhérentes à des productions futures non encore prévues (modifications ou ajouts de nouvelles gammes opératoires).

iii) Hiérarchisation de l'approche (analyse descendante). En effet, nous déduisons un graphe développé structuré à partir d'un modèle initial agrégé sur lequel une étude du comportement dynamique du système n'est pas envisageable.

iv) Automatisation de la démarche, afin d'accélérer la phase d'étude et de minimiser les erreurs de conception.

Ce mémoire est composé de quatre chapitres. Le premier concerne la description des modèles utilisés. Le deuxième décrit la méthodologie C.A.S.P.A.I.M. afin de situer le contexte de l'outil de structuration présenté dans le troisième chapitre. Enfin, le dernier chapitre concerne un exemple de mise en œuvre.

CHAPITRE I

INTRODUCTION

Très schématiquement, on peut décomposer tout système de production flexible en deux sous-ensembles :

- Le **procédé** regroupant l'ensemble des dispositifs matériels de l'installation (machines, systèmes de manutention, unités de stockage, ...) où circulent et sont transformés les produits.
- Le **système de commande**, chargé d'élaborer les ordres destinés au procédé, en fonction des consignes et des compte-rendus que lui transmet le procédé.

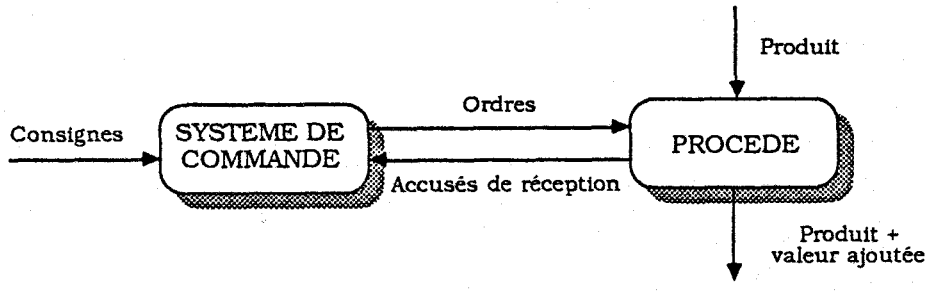


FIGURE 1.1

Dans le domaine de l'analyse et la synthèse des systèmes de commande d'ateliers flexibles, deux démarches essentielles ont été proposées.

La première repose sur une décomposition **hiérarchique** et **fonctionnelle** du système de commande en différents niveaux d'abstraction /ATA 87/. Par exemple, dans le projet S. E. C. O. I. A. /SAH 87/, le système de commande se décompose en trois niveaux d'abstraction (Fig. 1.2) :

i) la **commande locale** des machines qui correspond à des commandes numériques du type commande de robot ou commande de rotation d'outil,

ii) la **coordination de sous-ensembles,**

iii) l'**ordonnancement en temps réel** et le **pilotage global** qui prend en charge les tâches de :

- calculs d'itinéraires,
- gestion de stock,
- coordination globale,
- suivi et supervision.

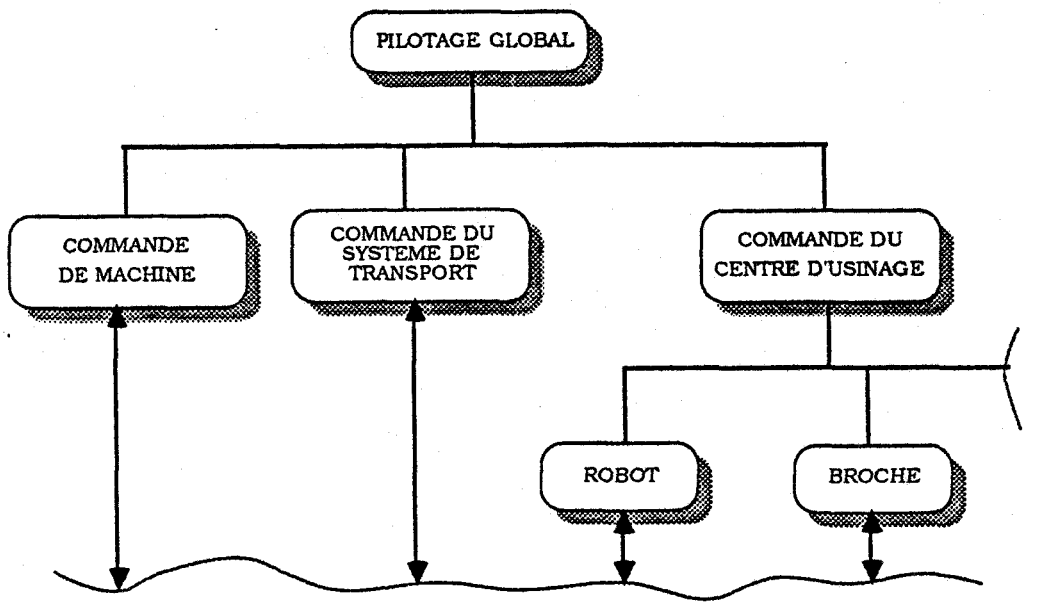


FIGURE 1.2

La seconde démarche consiste d'une part, à "mettre à plat" les différents niveaux qui constituent la partie procédurale du système de commande et d'autre part, à considérer un "niveau hiérarchique" qui paramètre la partie procédurale en intégrant les objectifs de production ainsi que des fonctions de supervision (Fig. 1.3).

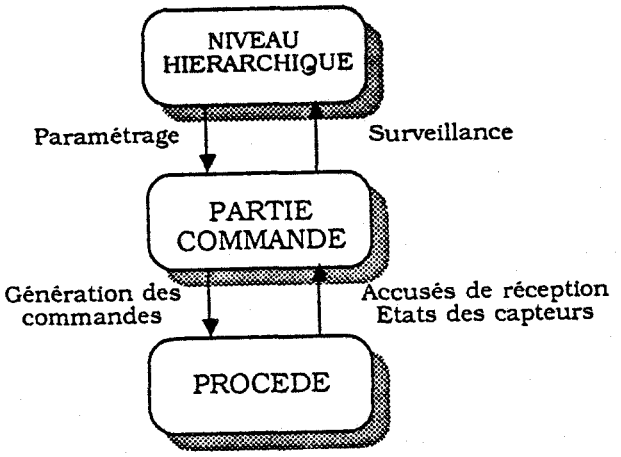


FIGURE 1.3

C'est cette dernière démarche que nous avons adoptée pour la décomposition et la modélisation des systèmes de production flexibles. Dans ce sens, nous décrirons successivement dans ce chapitre, les modèles utilisés dans notre étude pour la description de ces trois niveaux :

- les Réseaux de Petri Structurés Adaptatifs Colorés pour la partie commande,

- les règles de production pour le niveau hiérarchique,

- la temporisation du graphe de commande pour le procédé.

I - MODELISATION DE LA PARTIE COMMANDE

I.1 - Introduction

Afin de bien cerner la base des modèles choisis pour notre approche, nous présenterons dans un premier temps dans cette partie, une gamme d'outils permettant d'aborder par étapes le problème complexe que constitue la modélisation des systèmes de conduite de processus industriels. Le modèle de base utilise les **Réseaux de Petri Structurés** (RdPS). Ils permettent une description modulaire du système de commande par la prise en charge de façon sûre des interactions horizontales entre processus. La modélisation de l'interfaçage avec le niveau hiérarchique visant à assurer la flexibilité de fonctionnement du système (modes de marche, résolution des indéterminismes, ...) nécessitera l'utilisation d'une extension du modèle précédent appelée "**Réseaux de Petri Structurés Adaptatifs**" (RdPSA). Enfin, un dernier niveau d'extension sera également abordé permettant, quant à lui, la prise en compte des différentes classes d'objets qui circulent ou qui sont transformés, la partie commande disposant ainsi d'une image du procédé.

I.2 - Les Réseaux de Petri Structurés (RdPS)

I.2.1 - Introduction

L'apport de la structuration dans le cadre de la programmation a amélioré notablement la conception de grands programmes. Calquer cette démarche méthodique pour la description par des RdP de la commande des systèmes industriels se justifie d'autant plus que le graphe de commande est destiné à être implanté sur les divers organes de pilotage du système. Cette implantation s'effectue par l'intermédiaire des différents langages de programmation dont disposent ces matériels (langages littéraux ou graphiques). La structuration a priori du modèle RdP permet donc une traduction plus aisée et plus fiable dans la phase d'écriture des programmes de commande.

L'introduction de la structuration au niveau des RdP conduit à la définition d'une classe particulière et plus restrictive des RdP : les RdP structurés. Elle permet d'assurer une meilleure correspondance entre le modèle et son cahier des charges en limitant les erreurs de conception. Le modèle RdP structuré adopté /COR 79, 80/ s'appuie sur une décomposition fonctionnelle du système de commande en tâches élémentaires. En particulier, on n'autorise la mise en parallèle de tâches au sein d'un même processus (la structure "Fork-Join") que dans la mesure où ces tâches n'entrent pas en jeu dans une liaison inter-processus, ceci afin de préserver l'identité processus/processeur intéressante pour la phase d'implantation.

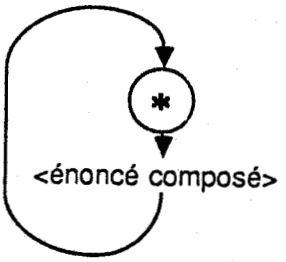
I.2.2 - Le modèle

a) Représentation modulaire : les graphes de processus

La notion de processus correspond ici au concept de programme en informatique, c'est-à-dire à un enchaînement de tâches.

Un processus est défini comme la combinaison séquentielle de 3 structures élémentaires : l'action, l'alternative, la répétitive. En utilisant la notation Backus-Naur, nous avons donc :

<processus> ::=



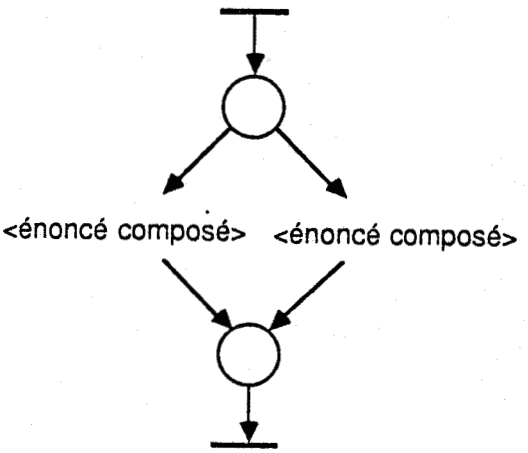
<énoncé composé> ::= <énoncé simple> / <énoncé simple>

<énoncé composé>

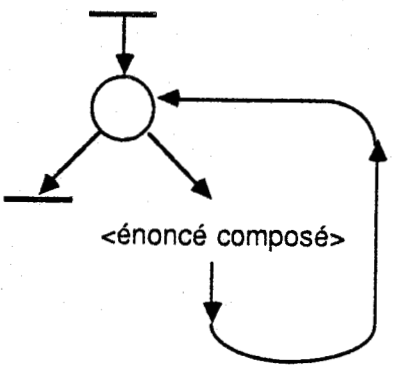
<énoncé simple> ::= <action> / <alternative> / <répétitive>

<action> ::=

<alternative> ::=



<répétitive> ::=

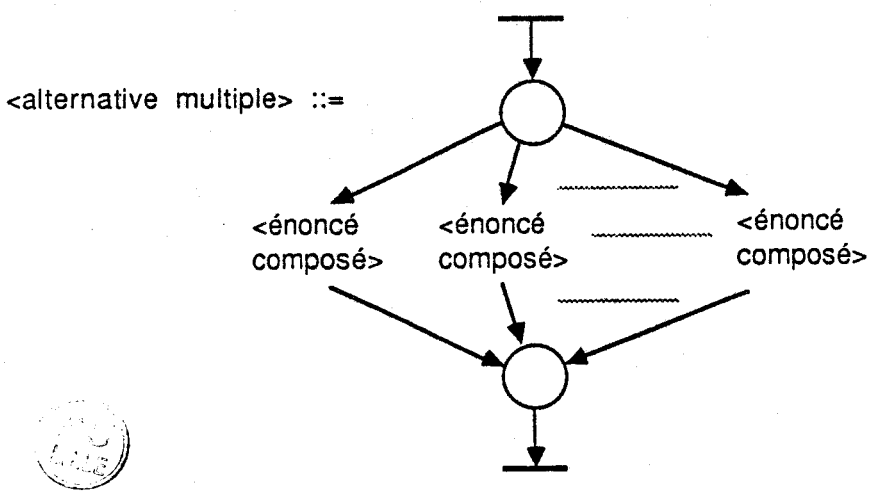


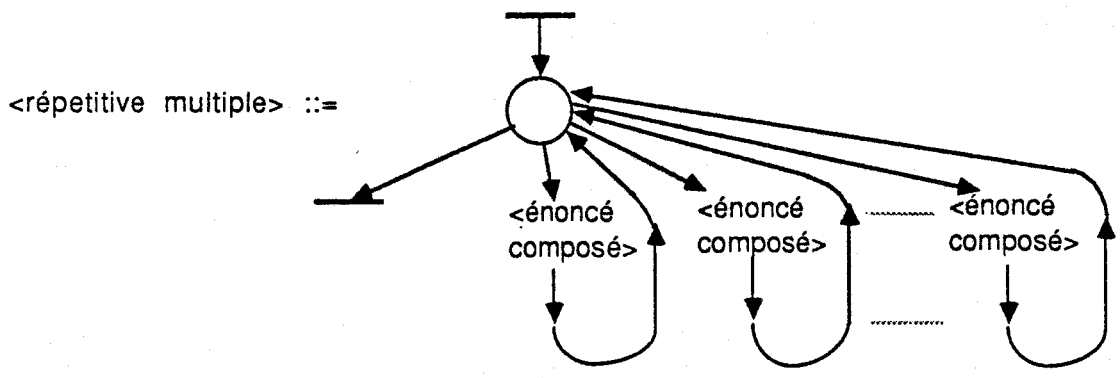
Afin de simplifier la description et de limiter la taille (nombre de places et de transitions) du graphe modélisant le système de commande, nous avons été amenés à définir deux énoncés simples supplémentaires. Ces deux structures sont des représentations simplifiées de combinaisons des structures <alternative> et <répétitive>. Ainsi, nous définissons les énoncés simples :

- <alternative multiple> équivalent à la structure de contrôle CASE-OF du langage PASCAL, et
- <répétitive multiple> équivalent à la structure suivante :

```
WHILE      <CONDITION>
  .          DO CASE <expression-Test> OF
  .               .
  .               .
  .               .
  .
                END ;
END ;
```

La syntaxe graphique de ces deux structures est la suivante :





De façon analogue aux langages de programmation structurés, la notion de bloc a été introduite pour permettre de regrouper un enchaînement séquentiel de tâches et de lui associer un nom. Cette démarche apparaît naturellement lorsque l'on analyse un problème par affinements successifs. Notons que la notion de bloc introduit des contraintes sur leur imbrication : en effet, deux blocs ne peuvent se chevaucher.

Ainsi, il est possible de décomposer un système de commande en n processus en tenant compte d'un certain nombre de critères tels que :

- le regroupement de tâches ayant un lien fonctionnel,
- le niveau de parallélisme de l'installation à piloter,
- les contraintes d'implantation répartie.

La représentation modulaire issue de cette décomposition en processus présente plusieurs intérêts :

- la possibilité d'archiver les graphes de fonctionnement associés à des structures fonctionnelles types utilisées à plusieurs reprises dans des applications différentes, permettant ainsi la création d'une base de données utilisable lors d'une démarche C. A. O. de la conception du système. Cette approche est réalisée dans /KAR 87/,
- l'indépendance des graphes de fonctionnement, ce qui implique que les modifications apportées à un graphe ne soient pas répercutées sur les autres graphes de processus,
- la partition d'un système en sous-systèmes relativement indépendants autorisant une conception progressive du réseau de Petri structuré par affinements successifs des blocs.

b) Interactions entre processus : les graphes de liaisons

Les processus de commande de deux systèmes de production qui **coopérent** au sein d'une même installation ne peuvent être indépendants. Afin de représenter les interactions entre les différents processus, trois types de liaisons ont été définies /COR 79/ :

- L'exclusion mutuelle

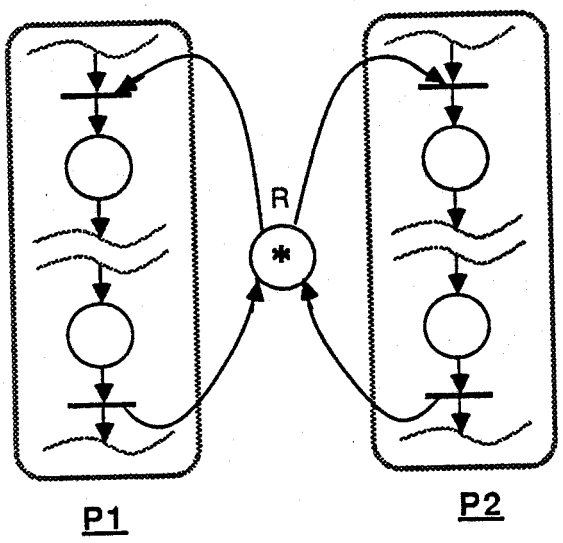


FIGURE 1.4

Cette structure assure l'unicité d'accès à une ressource critique partagée par plusieurs processus. Dans le cadre des systèmes de production, une telle structure permet de préserver un mode de commande unique pour le pilotage d'une entité physique affectée à deux tâches exclusives ; par exemple, un robot assurant deux transferts distincts.

- La synchronisation avec accusé de réception

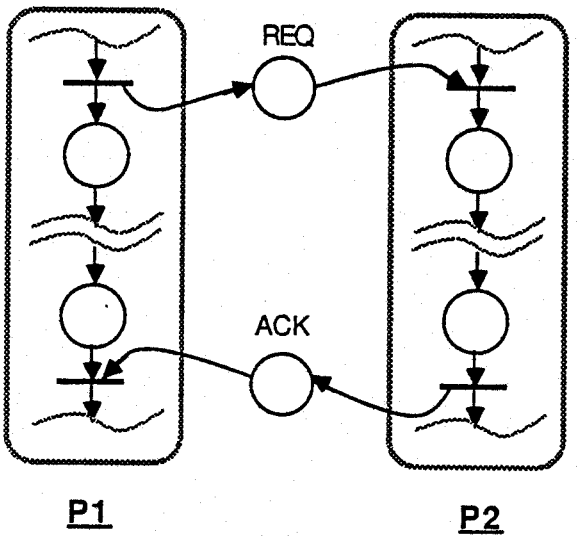


FIGURE 1.5

P1 synchronise P2.

- Le producteur/consommateur

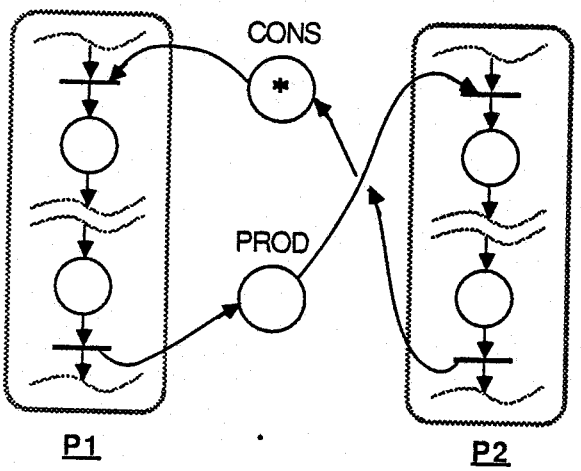
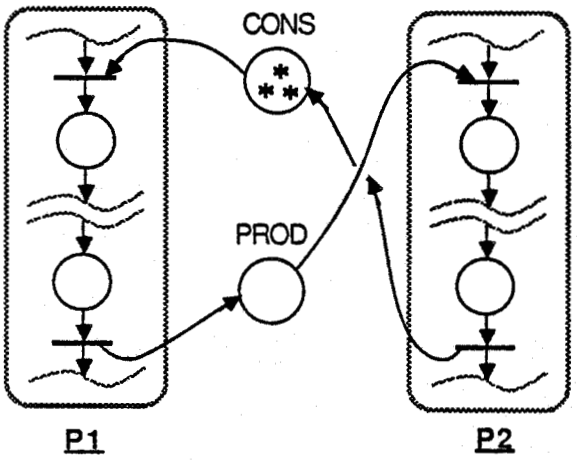


FIGURE 1.6

Le producteur P1 produit des marques dans la place PROD et P2 consomme ces marques.

Remarque : Par construction, les graphes de processus sont saufs, c'est-à-dire que le marquage de leurs places est au plus égal à 1. Il n'en est pas de même pour les places de liaisons qui peuvent avoir un marquage supérieur à 1. C'est le cas :

- pour le partage de deux ressources communes entre trois processus, ou plus fréquemment
- pour la représentation d'un mécanisme producteur/consommateur avec un tampon (place PROD) de capacité supérieure à 1 (Fig. 1.7).



Tampon à 3 places initialement vides

FIGURE 1.7

Dans ce cas, il est nécessaire de connaître et de représenter la structure du tampon en FIFO (First In First Out) par exemple. Ceci nous amène à étendre la structure classique producteur/consommateur en introduisant le concept de file d'attente. Ce mécanisme peut être modélisé par le réseau de Petri structuré de la Figure 1.8 pour un tampon à 3 places (entrée, milieu, sortie).

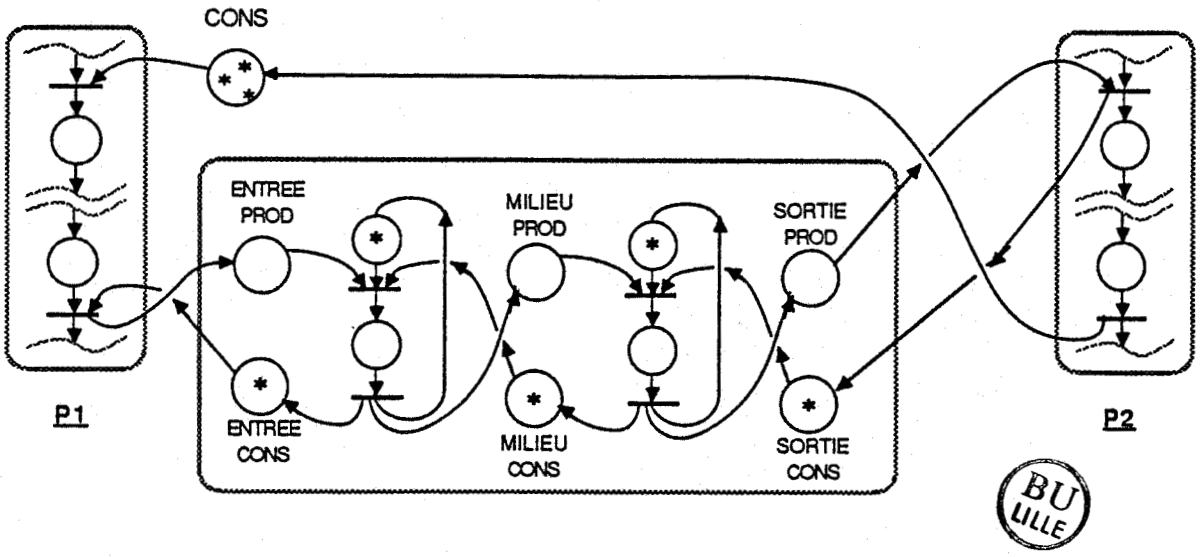


FIGURE 1.8

Par souci de simplicité et surtout de lisibilité, nous représenterons désormais de telles structures multiples par une primitive graphique "⊖" appelée "macro place de type FIFO". La Figure 1.8 devient donc symboliquement :

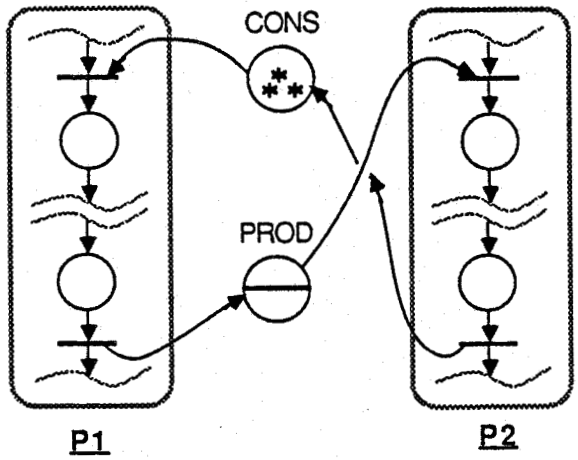


FIGURE 1.9

Des représentations d'autres structures de tampons (LIFO et anneau) ont été proposées dans /COR 85/.

c) Localisation des liaisons

Les interactions entre processus ne se font que par l'intermédiaire des blocs afin d'assurer une "bonne" construction du système de processus /COR 81/. Cependant cette contrainte ne garantit pas l'absence totale de blocages mais tout au plus permet de les minimiser en interdisant des constructions du type de la Figure 1.10.

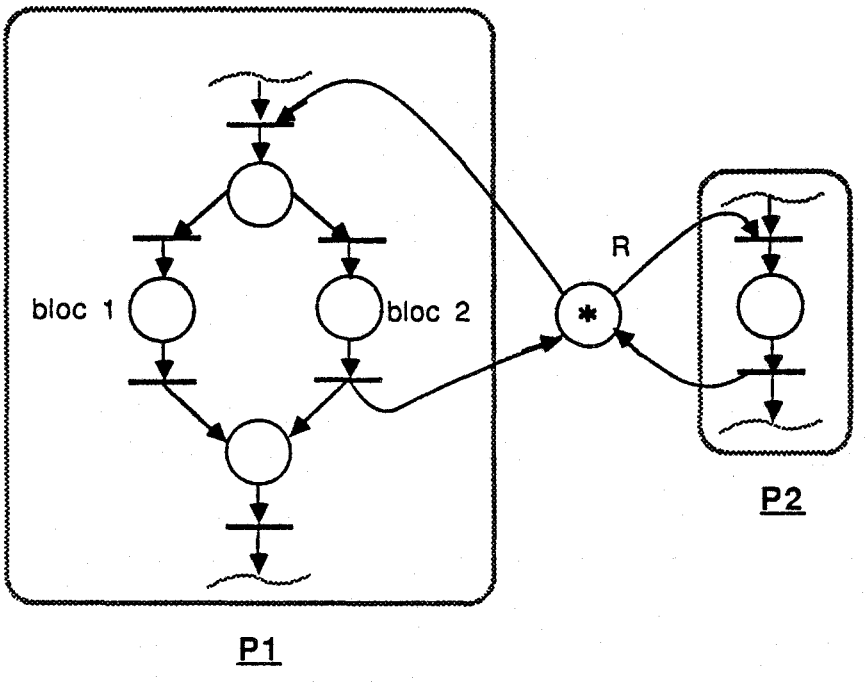


FIGURE 1.10

En effet, le concept de blocage, introduit ici de manière intuitive, est issu de la propriété de "vivacité" des RdP /BRA 83/ dont l'étude dépend non seulement de la structure du graphe mais également et essentiellement du marquage initial /CAS 87/.

Afin de rester cohérents avec la base de données de réseaux de Petri développée dans /KAR 87/, nous conserverons la règle de construction des liaisons par l'intermédiaire des blocs, règle qui n'est en rien contraignante pour la suite de notre étude et qui permettra d'utiliser ultérieurement des résultats issus de cette base de données.

I.3 - Extensions des Réseaux de Petri Structurés

I.3.1 - Les Réseaux de Petri Structurés Adaptatifs

a) Introduction

De nombreux exemples de processus parallèles et plus particulièrement de systèmes flexibles de production ne peuvent être modélisés par des réseaux de Petri ordinaires. En effet, les réseaux de Petri simples sont non déterministes et non flexibles et ne permettent donc pas de résoudre des problèmes tels que :

- les indéterminismes d'accès à une ressource,
- l'interfaçage avec le procédé et avec le niveau hiérarchique,
- la modélisation de la flexibilité relative aux divers modes de marche,
- etc, ...

Dans ce sens, diverses extensions du modèle de base ont été proposées :

- les arcs inhibiteurs /HAC 75a/,
- les réseaux à priorités /HAC 75a, 75b/,
- les réseaux automodifiants /VAL 78/.

Dans /COR 84/ a été proposée une extension des RdPS appelée **R**éseaux **d**e **P**etri **S**tructurés **A**daptatifs (RdPSA) proche des réseaux automodifiants de Valk. En effet, les réseaux adaptatifs n'imposent qu'une restriction supplémentaire : une transition possédant des arcs adaptatifs en amont, n'est franchissable que si le tir de la transition provoque une évolution du marquage amont. Ceci permet d'éviter la génération répétitive de marques par une transition source.

b) Applications à la modélisation

Dans le cadre qui nous intéresse (la productique), l'utilisation des RdP Structurés Adaptatifs nous permet :

- de préciser la sémantique de mécanismes complexes (Time-Out, compteurs d'évènements, ...),
- de résoudre les indéterminismes,
- de modéliser des changements de modes de marche,
- d'intégrer les notions de "gel" de bloc ou de processus,
- d'assurer d'une façon générale le paramétrage du modèle.

* **Résolution des indéterminismes** d'accès pour les ressources communes ou pour les producteurs/consommateurs à accès multiples :

Prenons l'exemple d'une ressource R partagée par deux processus P1 et P2.

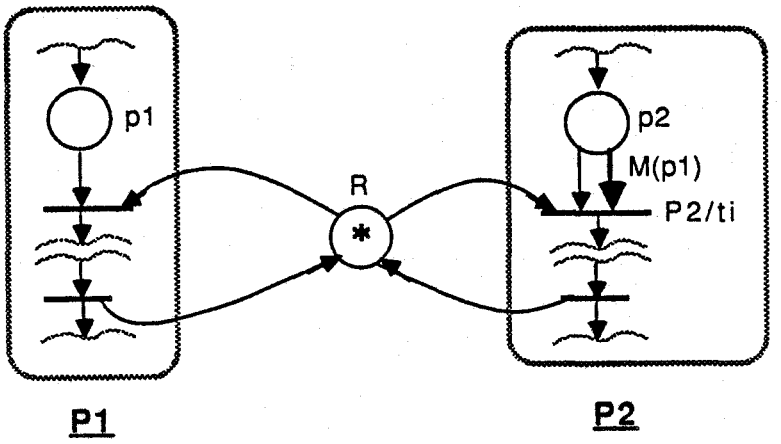


FIGURE 1.11

En cas de requête simultanée de la ressource R (p1 et p2 marquées), le processus P1 est prioritaire. En effet, la transition P2/ti n'est pas activable du fait du nombre insuffisant de marques dans la place p2 : les processus sont saufs, les places des processus sont donc 1-bornées. De cette façon, il est possible de "geler" une marque dans une place.

* **Modélisation des changements de modes de marche** en laissant à l'utilisateur la possibilité de représenter des "aiguillages" par l'intermédiaire de connexion ou déconnexion de sous-graphe :

Considérons l'exemple d'un processus P pour lequel deux sous-réseaux sont définis : le mode normal et le mode dégradé.

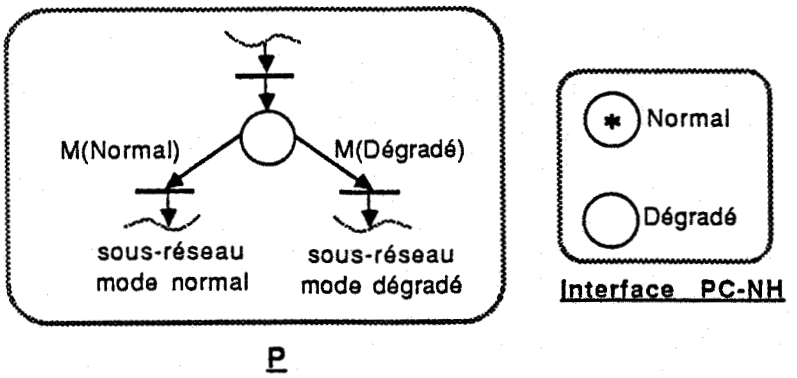


FIGURE 1.12

Le mode de marche du processus P dépend du marquage des places d'interface "normal" et "dégradé".

* **Intégration des notions de "gel" de bloc ou de processus** permettant le contrôle sélectif de l'évolution du graphe :

Prenons, à titre d'exemple, le processus P de la Figure 1.13.

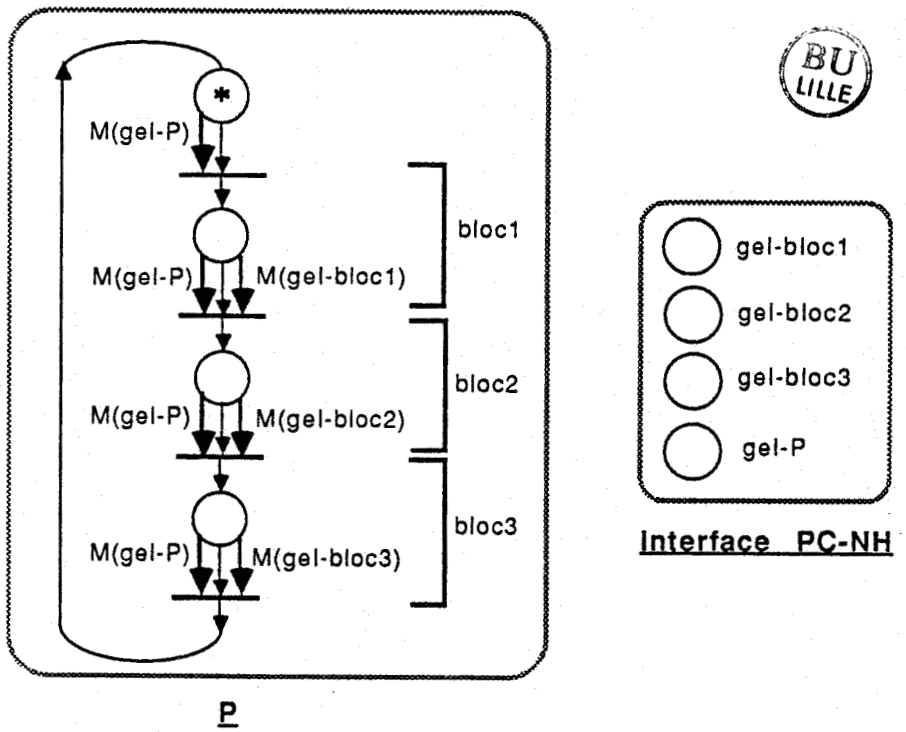


FIGURE 1.13

Le gel du processus P permet d'interdire le tir de toutes les transitions tandis que le gel d'un bloc permet d'interdire toute évolution dans ce bloc.

c) Conclusion

Cet outil de représentation (les réseaux adaptatifs) paraît séduisant. Il permet en effet, en théorie, d'assurer la représentation de tout mécanisme de contrôle flexible. Toutefois, il apparaît assez vite qu'un modèle de ce type amène très rapidement un niveau de complexité difficilement maîtrisable dans la représentation de processus de contrôle de grande taille faisant appel à des mécanismes de communication sophistiqués. En annexe, est proposé un catalogue de solutions pour représenter quelques mécanismes primitifs de haut niveau (compteur, time-out, ...).

I.3.2 - Les Réseaux de Petri Structurés Adaptatifs Colorés

a) Introduction

Lors de la phase de modélisation d'un système réel, la taille du réseau engendré devient généralement importante du fait de l'emploi répété de certains sous-réseaux identiques. Il devient alors intéressant de chercher à réduire la taille du modèle afin d'en faciliter l'écriture. Différentes études ont été menées dans ce sens et, en particulier, celles concernant les abréviations des réseaux par la prise en compte de la notion de coloration des marques /GEN 79/ /JEN 81/ /PET 80/. Il devient alors possible d'effectuer des regroupements fonctionnels ou structurels /CAS 87/.

b) Coloration en productique

Les systèmes de production flexibles sont caractérisés par la diversité des pièces conditionnées par le système. En fonction de leur type, les pièces circulant dans l'atelier subissent des traitements spécifiques différents ou non.

Les réseaux de Petri colorés apparaissent alors naturellement comme un outil de modélisation autorisant la prise en compte des différentes classes de pièces.

Etant donné que les objets subissent, au niveau du procédé, des transformations lors d'opérations telles que les usinages, il est intéressant, voire nécessaire, de reporter ces informations au niveau du choix de la couleur, par une transformation de cette dernière, lors du tir de la transition indiquant la fin d'action (Fig. 1.14).

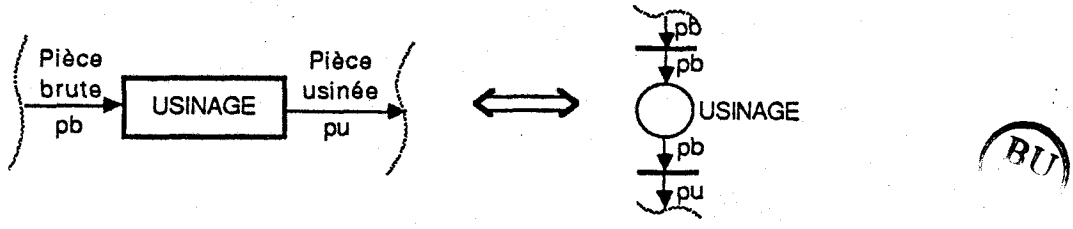


FIGURE 1.14

Ainsi, il est possible de disposer au niveau de la partie commande d'une image de la phase d'évolution du séquencement d'un type d'objet (son état d'avancement dans la gamme opératoire).

De plus, nous réaliserons des regroupements fonctionnels pour des opérations de **même nature**, sur des pièces **différentes** mais s'effectuant sur une **même** entité physique (machine, station d'assemblage, ...).

Exemple :

Soit une machine M sur laquelle sont usinées trois pièces brutes différentes pb1, pb2, pb3 qui donneront respectivement pu1, pu2, pu3 après les usinages spécifiques u1, u2, u3.

Le processus modélisant la commande de cette machine est représenté Figure 1.15.

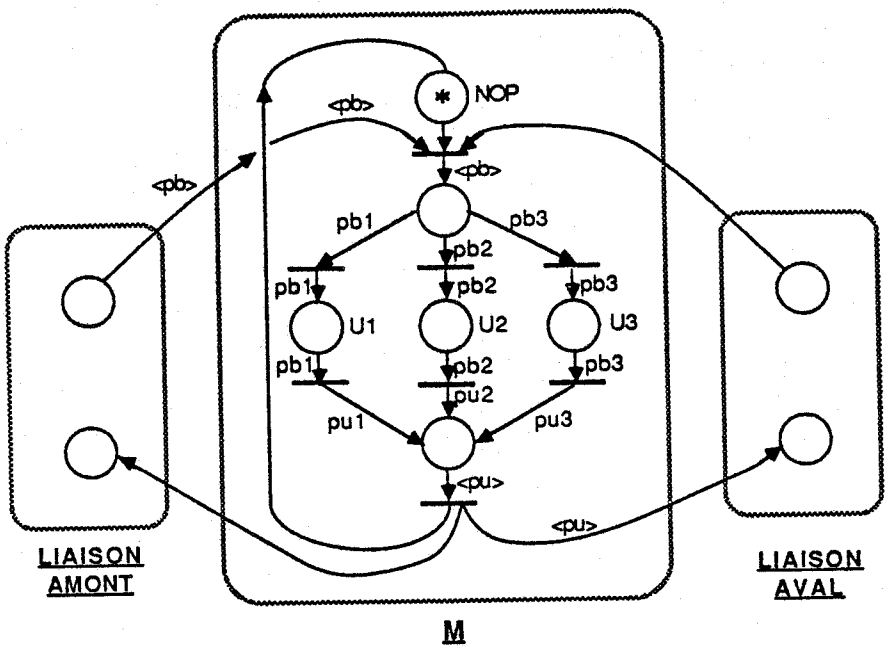


FIGURE 1.15

avec : dom (<pb>) = { pb1, pb2, pb3 }
dom (<pu>) = { pu1, pu2, pu3 }

Après un regroupement fonctionnel des usinages, on obtient :

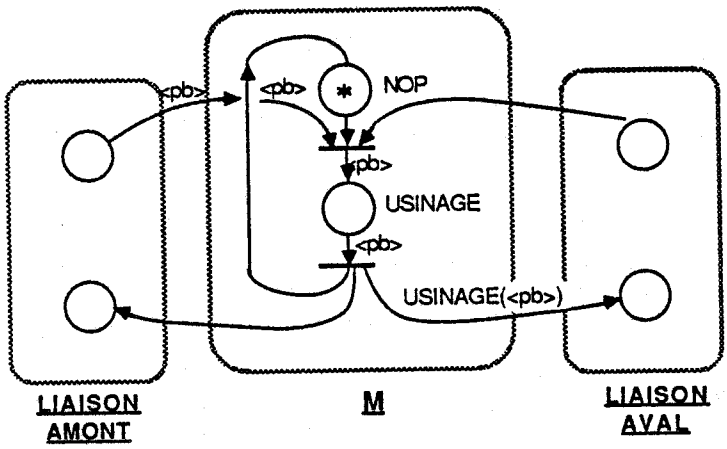


FIGURE 1.16

avec : usinage (pb1) = pu1
usinage (pb2) = pu2
usinage (pb3) = pu3

De même, pour les opérations d'assemblage ou désassemblage, nous réaliserons également de tels regroupements fonctionnels afin de diminuer la taille du modèle engendré. Cependant, nous n'effectuerons pas d'abréviation de niveau supérieur (ex : regroupement de machines de même structure) afin de ne pas atteindre un niveau d'agrégation qui permettrait, certes, de gagner en concision mais qui nuirait certainement à la lisibilité du modèle et qui ne serait d'aucune utilité en phase d'implantation.

c) Extension aux Réseaux de Petri Structurés Adaptatifs

La restriction due à la structuration d'une part, et l'adjonction de l'aspect adaptatif d'autre part, définissent notre modèle de base : les **Réseaux de Petri Structurés Adaptatifs Colorés** (RdPSAC).

Ce modèle est proposé pour représenter la partie commande des systèmes de production flexibles.

L'intégration du concept de coloration ne pose aucun problème dans la définition des primitives de structuration.

La notion d'adaptativité nécessite cependant la redéfinition de la notion de poids des arcs relatifs à une transition /CAS 87/ /COR 85/.

I.4 - Conclusion

Nous avons donc proposé dans cette partie trois classes de modèles présentés ici selon leur complexité croissante.

- RdPSAC
  - RdPSA
    - RdPS : 1) Banalisation du parallèlisme et des communications
    - RdPA : 2) (Dé)connexion de sous-réseaux réglement des indéterminismes
  - RdPC : 3) Agrégation du modèle
    - Représentation des classes d'objets
    - Transformation des informations

Le modèle RdPSAC est suffisant pour représenter correctement la partie commande d'un système flexible de production sans faire appel à des extensions plus puissantes des Réseaux de Petri.

II - MODELISATION DU NIVEAU HIERARCHIQUE

II.1 - Introduction

Afin d'assurer une meilleure productivité, les systèmes flexibles de production doivent intégrer des notions de gestion de production et plus généralement de supervision /FRO 84/ /BEN 86/. Définir une stratégie de production consiste entre autres à :

i) choisir parmi plusieurs ordonnancements possibles, définis lors de la conception du système, celui qui pourra satisfaire au mieux les objectifs de production. Ce choix se fera par exemple, au travers des priorités attribuées à des tâches concurrentes,

ii) réagir aux perturbations (défauts, pannes, ...) en réorganisant le système de commande afin d'assurer un cadre de production réduite : les fonctionnements dégradés.

Dans les deux cas, il s'agit d'un paramétrage du graphe de commande résultant de différentes prises de décision, dont nous regrouperons les mécanismes au sein d'un niveau hiérarchique **distinct** de la partie commande.

Ce choix résulte de la prise en compte de certains problèmes rencontrés lors de la conception ou la réalisation de systèmes de production flexibles :

i) l'intégration au sein du graphe de commande des structures de contrôle et de paramétrage se fait au détriment de la simplicité et donc de la flexibilité de la partie commande,

ii) la concepteur peut ne pas avoir connaissance de tous les contrôles à définir. Grâce à un niveau hiérarchique extérieur, il pourra réaliser des modifications de stratégie sans devoir reconfigurer le graphe de commande,

iii) la définition de priorités ou de changements de modes de marche nécessite parfois des informations issues d'une vision "globale" du système, telle que le taux d'engagement du (ou des) système(s) de convoyage, les temps moyens d'usinages, ...

II.2 - Choix du modèle

Différentes approches ont été étudiées pour la modélisation du niveau hiérarchique :

- les réseaux de Petri /SAH 87/ /COR 85/ qui présentent l'avantage d'être homogènes par rapport au graphe de commande que nous utilisons,
- les règles de production /BOU 86/ /BOU 87/ /MAR 87/.

Notons que la notion de transition pour un formalisme RdP est équivalente à l'énoncé d'une règle de production. Cependant, l'approche par règles de production présente les avantages suivants par rapport à des mécanismes utilisant des RdP pour la modélisation du niveau hiérarchique :

- une plus grande facilité pour traiter des incidents et établir des diagnostics /ABA 87/. Il s'agit en effet le plus souvent, de problématiques abordées en Intelligence Artificielle,
- une plus grande maîtrise de conception de la stratégie, grâce à des modifications aisées des structures de contrôle soit a priori, soit même en cours d'exploitation du procédé. Nous faisons allusion ici à la facilité d'adaptation des stratégies de pilotage, point de vue essentiel aussi bien dans la phase de synthèse qu'en phase d'exploitation.

Pour ces raisons, la démarche que nous avons adoptée est basée sur une description du niveau hiérarchique par règles de production.

Le but de notre étude n'étant pas d'aborder dans le détail les problèmes de choix du type de règles (requêtes ou surveillance /CAS 87/) ou du moteur d'inférence, nous nous limiterons ici à la description de la structure des règles et de leurs interfaces, ainsi qu'à l'utilisation d'un niveau hiérarchique "primitif" qui n'a pour seul objectif, que de permettre une simulation globale du graphe de commande généré. Les problèmes précités constituent néanmoins, l'une des préoccupations actuelles du Laboratoire d'Informatique Industrielle de l'I. D. N..

II.3 - Description des règles et de leurs interfaces

La structure des règles de production comporte :

i) un identificateur de règle,

ii) un ensemble de variables libres et la donnée de leur domaine de variation permettant d'assurer le lien avec les couleurs du graphe de commande,

iii) un énoncé du type SI <condition> ALORS <action> :

- la partie <condition> est constituée par une combinaison logique de prédicats testant l'état du graphe,
- la partie <action> est constituée par une séquence d'opérateurs "ajouter" et "enlever" définis comme suit :

  Ajouter (n,c,p) ajoute n marques de couleur c dans la place d'interface p,

  Enlever (n,c,p) enlève n marques de couleurs c dans la place d'interface p.

Les règles sont du type règles de surveillance /CAS 87/ : elles sont synchrones du cycle de base du mécanisme d'évolution du réseau de Petri.

Les places d'interfaces entre ces règles et le graphe de commande sont des places qui ne sont pas connectées au graphe et dont le marquage est modifié par la partie <action> des règles.

Considérons, à titre d'illustration, le cas de la définition d'une priorité d'accès à une ressource partagée par deux processus P1 et P2.

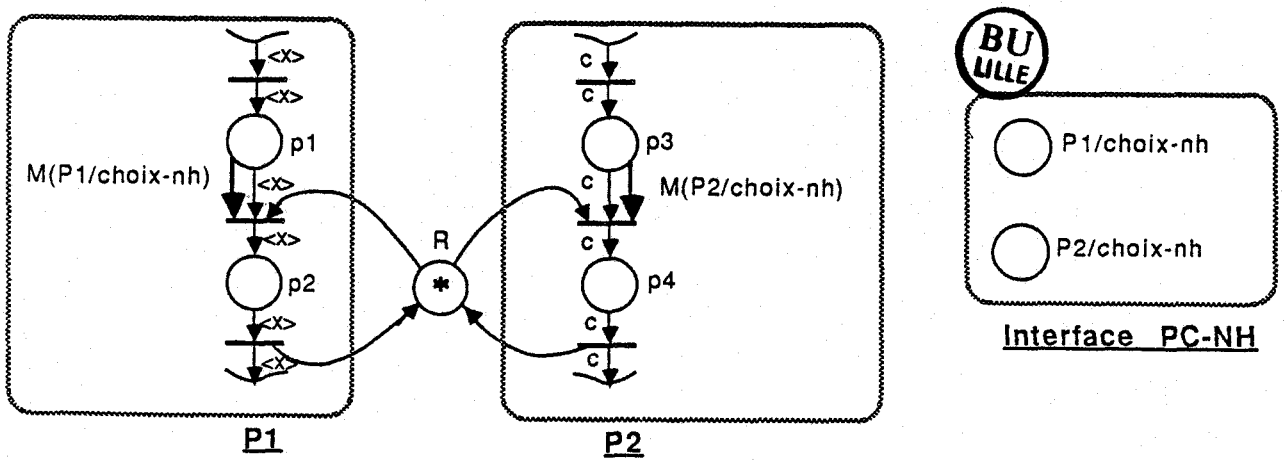


FIGURE 1.17

Définissons une stratégie : en cas de requêtes simultanées, le processus P1 est prioritaire si la place p1 est marquée par une couleur a, sinon P2 est prioritaire.

Règle : Annulation prio/P1

Enoncé SI (M (P2/choix-nh) (*) > 0) et
(M (p3) (c) = 0)
ALORS enlever (1, *, P2/choix-nh)

Règle : Annulation prio/P2

var-libre (x) domaine (x) = { a,b }
Enoncé SI (M (P1/choix-nh) (*) > 0) et
(M (p1) (x) = 0)
ALORS enlever (1, *, P1/choix-nh)

Règle : prio/P1

Enoncé SI (M (p1) (a) > 0) et
(M (p3) (c) > 0) et
(M (P2/choix-nh) (*) = 0)
ALORS ajouter (1, *, P2/choix-nh)

Règle : prio/P2

Enoncé SI (M (p1) (b) > 0) et
(M (p3) (c) > 0) et
(M (P1/choix-nh) (*) = 0)
ALORS ajouter (1, *, P1/choix-nh)

II.4 - Conclusion

Le niveau hiérarchique rudimentaire que nous avons proposé dans cette partie ne doit être considéré que comme le résultat de choix arbitraires visant à une **première** simulation des graphes de commande obtenus par le logiciel décrit dans le Chapitre III et ne prenant pas en compte des problèmes d'implantation réelle ni même d'optimisation de la simulation.

Néanmoins, cette première approche permet de définir dès les premières phases de conception, un ensemble de règles minimales nécessaires au bon fonctionnement du système de production.

Actuellement, des études sont en cours au L. A. I. I. visant :

- à définir les bases d'un niveau hiérarchique intégrant les notions d'architecture répartie, et
- à choisir des techniques d'inférences répondant aux contraintes de décision imposées par les différents systèmes de commande.

III - MODELISATION DU PROCEDE

III.1 - Introduction

La distinction entre le système commande et le procédé apparaît naturellement lors de l'observation d'un système automatisé. Ces deux parties coopérent selon le schéma suivant :

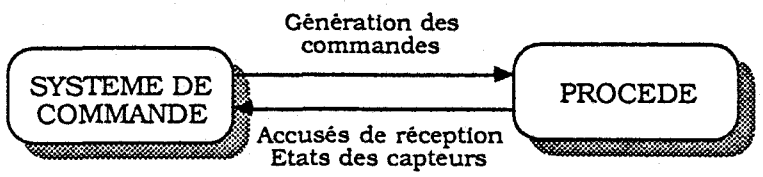


FIGURE 1.18

Note : Nous appelons ici système de commande l'ensemble formé de la partie commande et du niveau hiérarchique.

La partie commande est obligatoirement modélisée par un réseau de Petri **non autonome** /BRA 83/ au sens où son évolution dépend de l'état du réseau, mais également de l'état d'un **environnement**, constitué des variables et opérateurs mémorisant l'état instantané du procédé /CAS 97/.

Le modèle du système peut donc être représenté par la Figure 1.19 :

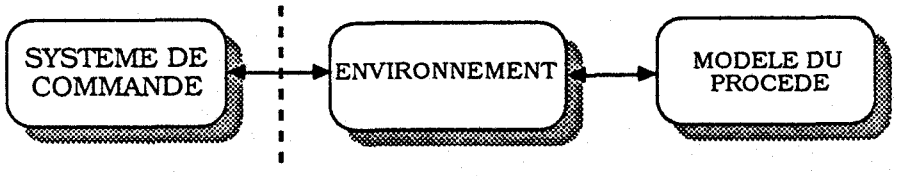


FIGURE 1.19

Dans la phase de conception et d'évaluation des performances d'un système, il est nécessaire de construire un environnement provisoire dont la durée de vie est limitée à la phase de conception. L'environnement réel s'y substituera dans la phase d'implantation.

Il existe dans la littérature, deux approches différentes pour appréhender cet environnement :

i) la création d'un modèle spécifique du procédé : c'est l'approche qui a été retenue, par exemple dans :
   - PIASTRE /CAR 83/ : les auteurs proposent ici de générer le grafcet dual du grafcet de commande,
   - /DEF 86/ : l'auteur utilise ici une famille d'automates d'états finis permettant de restituer l'image du comportement dynamique du procédé vu de la commande.

ii) l'extension du modèle RdP afin d'y intégrer la dynamique imposée par le procédé : les réseaux de Petri temporisés.

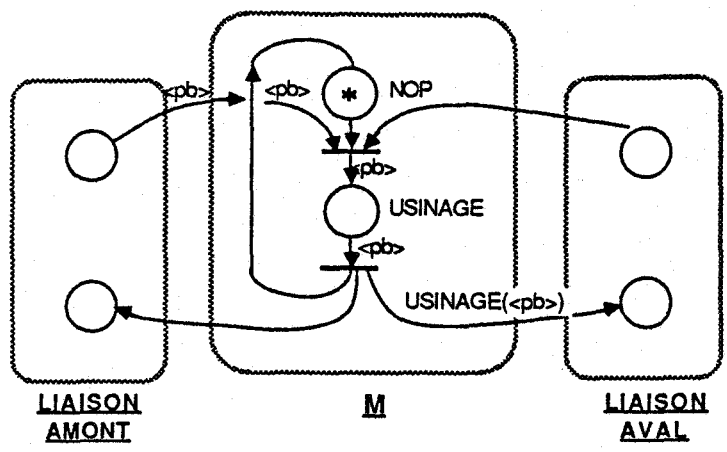
Nous ne nous intéresserons, dans la suite de cette étude, qu'à cette deuxième approche. Notons cependant, qu'une étude de faisabilité de la modélisation du procédé par des langages orientés objet a été réalisée dans /CAS 87/. Cette étude constitue une première étape d'une démarche prospective du L. A. I. I. visant l'intégration rigoureuse d'un modèle du procédé afin d'obtenir une évaluation complète des performances du système.

III.2 - Les Réseaux de Petri Temporisés
/RAM 73/ /MOA 76/ /SIF 77/ /CHR 83/

Les réseaux de Petri temporisés sont des réseaux non autonomes dont l'environnement fournit une référence de temps commune. Il sont utilisés pour étudier le comportement dynamique des systèmes en tenant compte uniquement de la durée de leurs actions.

Dans de tels réseaux, la présence d'une marque dans une place commande une action à laquelle est associée une durée. Le modèle que nous utilisons pour définir la partie commande étant les Réseaux de Petri Structurés Adaptatifs Colorés, nous serons amenés naturellement à définir des durées "colorées", c'est-à-dire fonction de la couleur de la marque.

Reprenons à titre d'exemple, la machine M décrite au § I.3.2.b dont le graphe de commande est le suivant :



avec : dom (<pb>) = { pb1, pb2, pb3 } et usinage (pb1) = pu1
usinage (pb2) = pu2
usinage (pb3) = pu3

FIGURE 1.20

Afin de faire une évaluation par simulation du comportement dynamique de cette machine, nous définissons une durée pour chaque usinage en accord avec le système physique réel. Nous aboutissons à la définition d'une fonction de temporisation de la place USINAGE.

Par exemple : tempo-usinage (pb1) = 100 UT (Unité de Temps)
tempo-usinage (pb2) = 150 UT
tempo-usinage (pb3) = 180 UT

Les marques <pb> qui arrivent à l'instant t dans la place USINAGE seraient indisponibles jusqu'à l'instant t + tempo-usinage (<pb>).

La temporisation permet ainsi, dès la phase de conception, une **première** évaluation quantitative des performances du système tout en préservant la structure du graphe de commande.

Cependant, cette solution qui présente l'avantage d'être simple et rapide à mettre en œuvre, n'en est pas moins inadaptée à l'étude plus fine du comportement des systèmes dont les durées d'action ne résultent pas uniquement des commandes. C'est le cas, par exemple, d'un convoyeur débrayable pour lequel les temps de transfert entre deux points sont fonction de manière générale de l'état interne du procédé, par exemple :

- la vitesse du convoyeur,
- l'existence de butée de débrayage,
- l'encombrement du convoyeur en objets spécifiques devant satisfaire des contraintes temporelles parfois contradictoires.

De ce point de vue, la nécessité d'intégrer un modèle spécifique du procédé est donc tout à fait justifiée.

CONCLUSION

Dans cette première partie, nous avons présenté les différents modèles nous permettant d'une part, de concevoir le système de commande d'un processus de production flexible, et d'autre part, d'en faire une première évaluation par simulation.

En premier lieu, le modèle RdPSAC permet de préciser l'architecture parallèle d'ensemble de la partie commande et d'en caractériser les modes automatiques sans règlement des conflits.

En second lieu, le niveau hiérarchique est modélisé par un ensemble de règles de production permettant d'assurer un comportement déterministe de la partie commande.

Enfin, grâce à la temporisation des places, il est possible d'intégrer au sein du graphe de commande le minimum d'informations, concernant le procédé, nécessaires à une évaluation des performances du système.

CHAPITRE II

INTRODUCTION

La conception du système de commande d'ateliers flexibles de production constitue un problème complexe dépendant d'un grand nombre de paramètres de natures diverses : flexibilité, productivité, coût, ...

A partir de cette constatation, la nécessité d'une méthodologie d'étude et de conception assistée semble s'imposer, pour pouvoir plus **facilement** tenir compte des interactions entre les différents choix et surtout pouvoir procéder **rapidement** à une évaluation qualitative et quantitative d'une famille de solutions admissibles (prototypage rapide).

De telles méthodologies présentent plusieurs avantages :

- elles permettent de gagner un temps considérable dans la phase de conception et de mise au point du système de commande,
- elles diminuent le nombre d'erreurs tant dans la phase de spécifications que dans les étapes de synthèse du système de commande et d'implantation sur site grâce à des outils informatiques puissants,
- elles améliorent la documentation des projets.

L'automatisation de la démarche de conception n'est généralement possible qu'après une structuration en différentes étapes de la spécification à l'implantation.

De nombreuses études ont été menées dans ce sens telles que celles de MARTINEZ /MAR 87/, MOSER /MOS 87/ et les projets S. E. C. O. I. A. /COU 84/ et PIASTRE /PRU 87/.

Si ces méthodologies permettent d'obtenir des résultats nouveaux et intéressants, nous pouvons noter qu'elles ne prennent généralement pas en compte l'intégralité des éléments suivants :

i) la validation des spécifications du cahier des charges par une analyse de complétude et de cohérence des gammes opératoires,

ii) la description exhaustive des trois niveaux (PC, NH, Procédé) permettant d'assurer une évaluation a priori du système aussi complète que possible,

iii) la structuration de la partie commande permettant notamment :
- des modifications aisées a posteriori,
- la mise en évidence du parallélisme du système,
- un meilleur suivi du comportement dynamique du système en cours d'exploitation,
- etc ...

Dans ce sens, nous présenterons dans ce chapitre une méthodologie de conception permettant l'analyse, la synthèse et la validation de systèmes de production flexibles à partir de la spécification en langage naturel du cahier des charges.

Dans le souci de mieux situer notre approche, nous consacrerons une partie de ce chapitre à la présentation du projet C. A. S. P. A. I. M.(*) dans sa globalité. Puis, nous décrirons plus particulièrement deux étapes essentielles constituant les points d'entrée et de sortie du logiciel de structuration présenté dans le Chapitre III afin d'en préciser le contexte : l'obtention d'un modèle intermédiaire agrégé et la simulation du graphe développé.

(*) (Conception Assistée de Systèmes de Production Automatisés pour l'Industrie Manufacturière)

I - DESCRIPTION DE LA DEMARCHE

I.1 - Introduction

La finalité du projet C.A.S.P.A.I.M. est de fournir un outil méthodologique constitué d'un ensemble de logiciels d'aide à la conception et à l'implantation du système de commande d'ateliers flexibles, à partir de la spécification, en langage naturel, du cahier des charges.

Cet outil est tout d'abord basé sur une méthode de conception **interactive** de type "modélisation-simulation". Le modèle utilisé doit traduire au mieux la réalité du système physique qu'il représente. De ce fait, il doit prendre en compte les aspects stratégiques/décisionnels et productifs très complexes qui relèvent en général des systèmes flexibles de production, en particulier :

i) la description des paramètres produits : types de pièces, gammes opératoires, etc...,
ii) la représentation des architectures retenues : architectures physiques (systèmes de manutention, plan d'implantation de l'atelier, ...) et informatiques (matériels et réseaux de communication),
iii) la définition des stratégies de contrôle et de pilotage de l'atelier,
iv) les contraintes de production : délais, débits, ...

Face à de telles contraintes de modélisation, le souci de toute l'équipe du L. A. I. I. a été de choisir des modèles adaptés à une description cohérente de tels systèmes. Dans ce sens, nous avons choisi de modéliser :

- la partie commande par Réseaux de Petri Structurés Adaptatifs Colorés,
- le niveau hiérarchique par règles de production,
- le procédé par langage orienté objet.

Afin de décrire dans son ensemble le projet C.A.S.P.A.I.M., nous en donnerons tout d'abord les idées générales. Nous présenterons ensuite de manière plus détaillée chaque étape et son état d'avancement actuel.

I.2 - Le schéma directeur général

Le schéma directeur général du projet C.A.S.P.A.I.M. est donné sur la Figure 2.1.

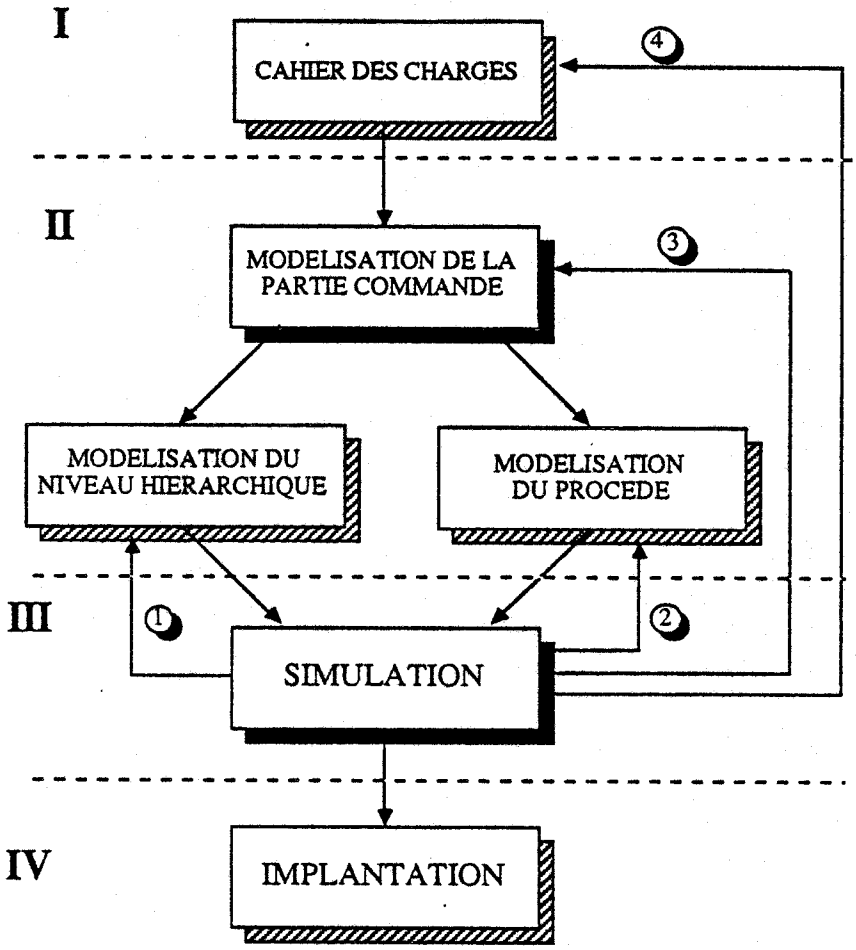


FIGURE 2.1

Sur ce schéma, nous distinguerons quatre phases de conception :

- la spécification du cahier des charges de l'application considérée (Phase I),
- la modélisation (Phase II),
- la simulation (Phase III),
- enfin, la phase de synthèse du système de commande sur un support informatique réparti ou non, phase que nous appellerons "Implantation" (Phase IV).

Cette méthode de conception permet non seulement d'étudier la faisabilité du système complet à partir de premières simulations globales, mais également d'effectuer des études de définition plus précises /BON 85/ par des simulations de détail ; par exemple, nous étudierons les modes de marches dégradées après avoir procédé à des affinages successifs des différents modèles (bouclages ① ② ③).

Remarque sur l'état d'avancement du projet :

Les étapes hachurées sur la Figure 2.1 sont l'objet d'études en cours au Laboratoire d'Informatique Industrielle de l'I. D. N..

De ce fait, nous avons effectué arbitrairement un certain nombre de choix afin de réaliser au mieux la jonction cohérente entre la spécification du cahier des charges et l'implantation.

Ceci a permis de démontrer à la fois la faisabilité d'une telle démarche, et son intérêt méthodologique.

Dans ce sens, nous proposons de modéliser :

i) le niveau hiérarchique par règles de surveillance (cf. Chapitre I, § II.2),

ii) le procédé par temporisation des places du graphe de commande (cf. Chapitre I, § III.2).

Ces choix, qui ne sont bien évidemment pas définitifs, présentent l'avantage d'être facilement et donc rapidement exploitables.

I.3 - Le schéma directeur détaillé

Compte tenu de la remarque précédente, le schéma directeur détaillé du projet C.A.S.P.A.I.M. dans son état actuel est donné Figure 2.2. Nous y retrouverons les quatre phases : spécification, modélisation, simulation, implantation qui sont constituées respectivement de l'ensemble des étapes {1,2,3}, {4,5,6,7,8,9,10,11}, {12}, {13}.

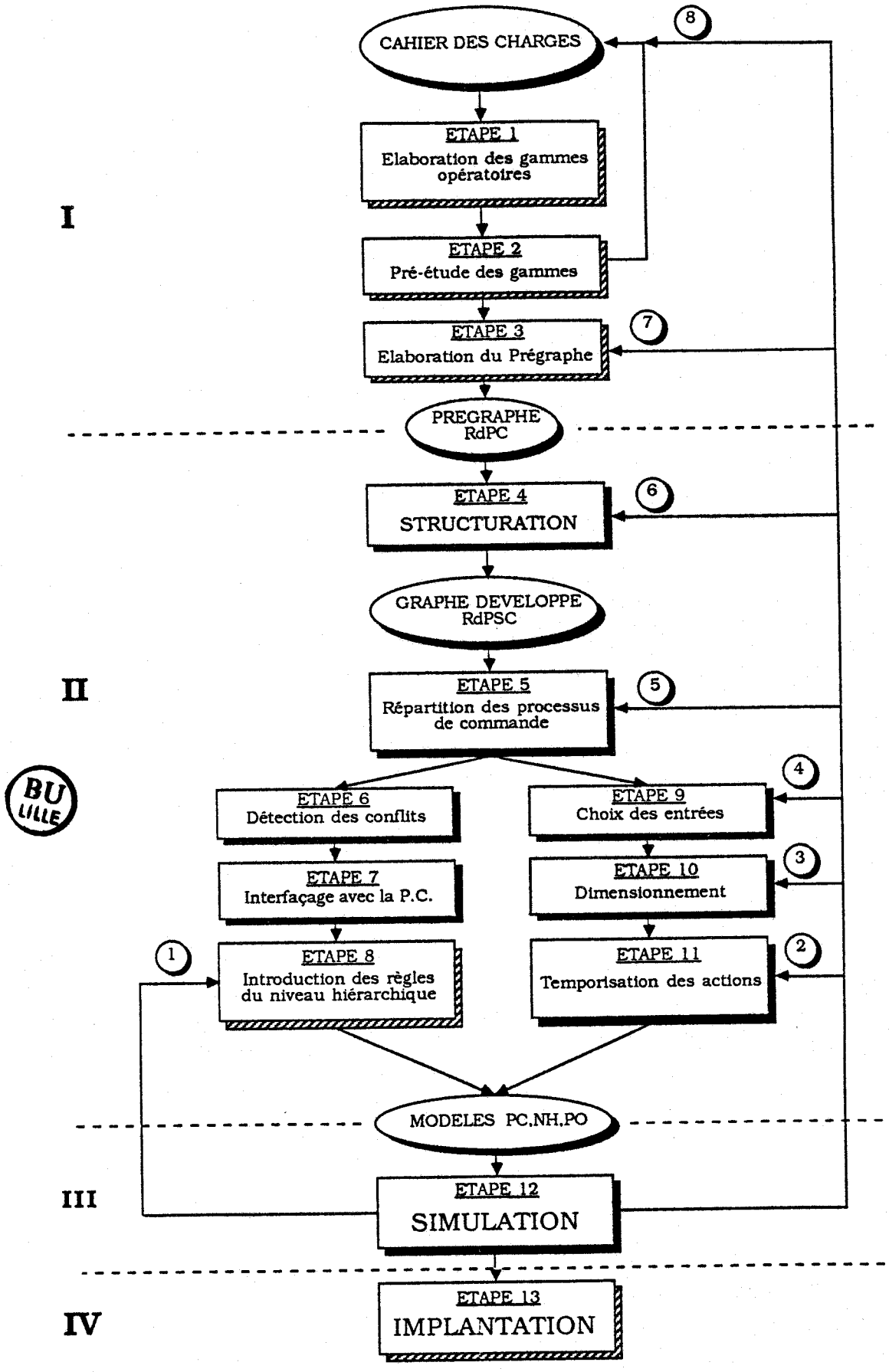

CAHIER DES CHARGES
8
ETAPE 1
Elaboration des gammes opératoires
I
ETAPE 2
Pré-étude des gammes
ETAPE 3
Elaboration du Prégraphe
7
PREGRAPHE RdPC
ETAPE 4
STRUCTURATION
6
GRAPHE DEVELOPPE RdPSC
II
ETAPE 5
Répartition des processus de commande
5
ETAPE 6
Détection des conflits
ETAPE 9
Choix des entrées
4
ETAPE 7
Interfaçage avec la P.C.
ETAPE 10
Dimensionnement
3
1
ETAPE 8
Introduction des règles du niveau hiérarchique
ETAPE 11
Temporisation des actions
2
MODELES PC,NH,PO
III
ETAPE 12
SIMULATION
ETAPE 13
IMPLANTATION
IV

FIGURE 2.2

I.3.1 - La phase de spécifications

Afin de bien préciser le contexte dans lequel nous nous situons, rappelons tout d'abord quelques définitions données dans /BOU 84/.

*** Le cahier des charges :**

Rédigé en langage naturel, ce document est établi à partir des besoins du client. Il comporte un ensemble de spécifications (requêtes, contraintes). Citons, à titre d'exemples, quelques éléments du cahier des charges :

- les caractéristiques d'utilisation telles que la maintenabilité, la sûreté de fonctionnement, etc...
- quantités à produire,
- délais à garantir,
- prix,
- etc....

*** Le dossier de conception :**

Etabli à partir du cahier des charges par le bureau d'étude, ce document spécifie un ensemble de **caractères fonctionnels** de l'objet à fabriquer :

- plans d'ensemble,
- plans détaillés,
- caractéristiques matérielles des différents sous-ensembles,
- états de surfaces,
- etc....

L'élaboration d'un tel dossier utilise très largement les outils proposés dans la cadre de la C. A. O..

*** Le dossier de fabrication :**

C'est un dossier technique définissant les **moyens** de production à mettre en œuvre :

- machines à employer,
- gammes opératoires,
- outils et outillages nécessaires.

ETAPE 1

Elaboration des gammes opératoires

Cette étape vise à établir à partir du cahier des charges une description fonctionnelle élémentaire du système sous forme de règles de production. Chaque gamme opératoire est ainsi décrite par un ensemble de règles de production dont la structure est la suivante :

prémisses → conséquents

où prémisses et conséquents sont des prédicats ou conjonctions de prédicats de la forme :

(présence objet lieu)

qui traduit qu'un objet est présent en une position repérée d'un support.

Dans le cas qui nous intéresse (la productique), la notion d'objet recouvre deux interprétations :

- les pièces, au sens de produit transformé dans le système,
- les moyens ou supports de production, au sens d'outils, de palette, de platines d'usinage, etc....

Nous reviendrons plus précisément sur la description des règles dans le paragraphe II.

Après une étude des logiciels de spécification existants, une partie de l'équipe du L. A. I. I. a été chargée de réaliser un logiciel basé sur EPOS permettant de répondre à l'objectif de cette étape. Ce logiciel, qui s'inspire du guide concepteur décrit dans /AUG 87/, proposera, au travers d'une interface utilisateur puissante, un outil de structuration des données du cahier des charges permettant de fournir une documentation complète et d'effectuer une première analyse de complétude (détection d'opération manquante).

Cette étape, qui est en cours d'étude, prendra également en compte des travaux déjà existants dans le domaine de l'élaboration de gammes opératoires /BOU 84/ /BOU 87a/ /CAM 85/ /BOU 87b/.

ETAPE 2

Pré-étude des gammes

L'ensemble des gammes opératoires étant décrit pour chaque type de pièce par des règles de production, nous utilisons un logiciel chargé de l'analyse de la complétude et de la cohérence. Ce dernier est l'objet d'une étude actuellement en cours d'achèvement au L. A. I. I. /KAP 87/.

Ce logiciel utilise un raisonnement de type Intelligence Artificielle. Il est constitué d'étapes coopérantes inférant en chaînage avant et arrière permettant :

i) d'aider le concepteur dans la constitution de sa base de règles décrivant les séquences opératoires,

ii) d'éviter les incomplétudes et les incohérences en comparant les faits initiaux, les faits terminaux et les règles énoncés avec les faits initiaux, les règles utilisées et faits terminaux obtenus.

En cas d'échec au cours de cette étape, le concepteur est amené à modifier de façon interactive sa base de faits et sa base de règles.

ETAPE 3

Elaboration du prégraphe

Cette étape, qui sera détaillée au paragraphe II, se décompose en deux sous-étapes :

- la **modulation** qui permet la duplication de certains sites ou certaines actions afin d'accroître la flexibilité du système,
- l'**élaboration du prégraphe** proprement dite qui, à partir des données (règles) issues de la modulation, génère un modèle intermédiaire que nous appelons "**Prégraphe**" et qui est une agrégation des règles sous forme de Réseau de Petri Coloré. Ce modèle intermédaire, qui constitue le résultat de la phase de spécifications, représente l'ossature initiale du modèle procédural structuré qui conduira à l'élaboration de la partie commande du système.

I.3.2 - La phase de modélisation

Cette phase, dont toutes les étapes (exceptée l'étape 8) constituent l'ensemble du logiciel détaillé au Chapitre III, a pour but d'assister la construction des trois modèles du système. Ils sont, rappelons-le, indispensables à l'analyse qualitative et quantitative du comportement dynamique du procédé et de sa commande.

a) Modélisation de la partie commande

ETAPE 4

Structuration

Nous déduisons du prégraphe un modèle développé **structuré** qui met en évidence le parallélisme de fonctionnement du système.

ETAPE 5

Répartition des processus de commande

Dans cette étape, nous prenons en compte les contraintes relevant de l'architecture matérielle de l'atelier. Nous indiquons les processus qui représentent les commandes des mêmes entités physiques. Par exemple, deux processus différents commandent le même robot pour qu'il effectue deux transferts différents. Dans ce sens, nous sommes conduits à la mise en place de sémaphores correspondant aux ressources communes.

b) Modélisation du niveau hiérarchique

ETAPE 6

Détection des conflits

Cette étape a pour but de détecter, à partir d'une scrutation systématique du graphe de commande, tous les conflits "structurels", c'est-à-dire les conflits inhérents à la structure du graphe tels que les accès simultanés à une ressource commune ou, de manière plus générale, les indéterminismes.

ETAPE 7

Interfaçage avec le graphe de commande

Cette phase concerne la génération des arcs adaptatifs et des places d'interface qui seront utilisés lors de l'activation des règles.

ETAPE 8

Introduction des règles du niveau hiérarchique

A partir de la liste des conflits et des objectifs stratégiques qui peuvent être issus directement du cahier des charges et/ou des résultats de simulation, nous introduisons l'ensemble des règles permettant le pilotage. Cette étape étant en cours d'étude, la saisie des règles est assurée actuellement par un module spécifique simplifié du simulateur. Compte tenu de la complexité des aspects stratégiques dans les systèmes flexibles de production, la génération des règles ne pourra être totalement automatisée mais sera en fait assistée par le logiciel en cours d'écriture.

c) **Modélisation du procédé**

ETAPE 9

Choix des entrées

Nous choisissons, à ce niveau, le flux des pièces qui pénétrent dans le système. Cette étape n'a de sens qu'en vue d'une simulation. En effet, dans le système réel, le flux des pièces entrantes n'est fonction que des unités de production amont.

De ce fait, il est nécessaire de générer de façon artificielle des entrées de pièces pour effectuer la simulation du système "isolé", prenant ainsi en charge les contraintes et séquences de production du cahier des charges.

ETAPE 10

Dimensionnement

Nous affectons à toutes les zones tampons de taille variable, une dimension qui correspondra tout au long de la simulation à la taille maximale de la zone. Le simulateur permettra de valider ces dimensionnements.

ETAPE 11

Temporisation

De la même façon, nous introduisons une temporisation pour chaque place du graphe de commande qui correspond à une action effective. Ceci constitue une première approche d'un modèle du procédé qui permettra une **première simulation** de type quantitatif.

Remarque : Par souci de lisibilité, ces trois dernières étapes ont été représentées de façon séquentielle sur la Figure 2.2. Elles sont bien évidemment indépendantes et peuvent donc être traitées dans un ordre quelconque.

Les modèles des trois parties (partie commande, niveau hiérarchique, procédé) étant obtenus, il est donc possible d'aborder la phase de simulation.

I.3.3 - La simulation

ETAPE 12

A ce stade de la conception, nous pouvons effectuer une étude complète du comportement dynamique du système. C'est dans ce sens que nous avons développé un simulateur /CAS 87/ qui prend en compte les trois modèles et dont nous donnerons les grandes lignes dans le paragraphe III.

La simulation du système complet permet alors de mettre en évidence d'éventuelles anomalies de comportement et par voie de conséquences de trouver la (ou les) solution(s) en jouant sur les éléments suivants (Fig. 2.2) :

- la stratégie de pilotage de l'installation par modification de la base de règles du niveau hiérarchique (bouclage ①),
- la temporisation de certaines places (bouclage ②) dans le cas, par exemple, d'une modification de la distance physique entre deux lieux qui entraînera, de fait, une modification de la durée des opérations de transfert des objets entre ces deux emplacements,
- la taille des zones tampons (bouclage ③),
- les flux d'entrées de pièces (bouclage ④) afin de trouver, par exemple, celui qui amènera le meilleur taux d'occupation des machines,

- la répartition des processus de commande (bouclage ⑤) lors de l'ajout ou du retrait d'un robot ou même lors d'une synchronisation de deux opérations,
- la structure du graphe de commande (bouclage ⑥) en changeant le type d'une machine et donc son graphe de commande,
- le prégraphe (bouclage ⑦) en dédoublant un site ou une opération (étape de modulation) afin d'accroître la flexibilité de l'installation en permettant, par exemple, à une pièce d'être usinée indifféremment sur deux machines identiques,
- le cahier des charges (bouclage ⑧) dans le cas d'une modification de l'implantation de l'unité de production qui aurait pour effet de modifier les trajets suivis par les pièces et donc leurs gammes opératoires.

Remarque : Le coût d'une modification dépend bien évidemment du bouclage que l'on utilise, chaque étape de la conception ayant une incidence plus ou moins importante sur les étapes situées en aval.

Lorsque les résultats de simulation répondent à l'ensemble des critères imposés, la solution est donc techniquement admissible. Il est alors possible d'envisager la phase d'implantation.

I.3.4 - Implantation

ETAPE 13

Dans cette phase, nous effectuons la synthèse du système de commande (niveau hiérarchique et partie commande) sur un support informatique formé le plus généralement :

- d'un organe de pilotage (micro-ordinateur),
- d'un (ou plusieurs) réseau(x) local (locaux) de niveau 2 /THO 85/,
- et d'un ensemble d'automates programmables industriels (A.P.I.) et/ou d'armoires de commande numérique (C.N.C.).

Cette synthèse est réalisée à partir du graphe de commande, de l'ensemble des règles et du moteur d'inférences qui forment le niveau hiérarchique.

L'architecture matérielle générale d'un tel système de commande est donnée Figure 2.3. Les liaisons fonctionnelles devront être assurées par le(s) réseau(x) local (locaux).

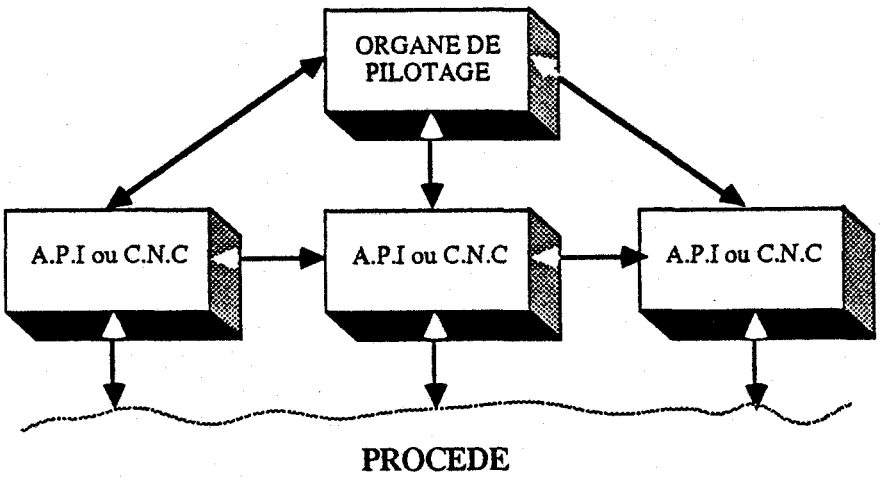


FIGURE 2.3

Deux possibilités "limites" d'implantation sont alors envisageables.

En premier lieu, il est possible d'implanter le niveau hiérarchique sur l'organe de pilotage, les processus du graphe de commande sur l'ensemble des automates et les liaisons interprocessus sur le réseau local.

Une seconde option consiste à implanter l'intégralité du système de commande sur l'organe de pilotage et à se servir du réseau pour télécharger les tâches à exécuter sur les automates correspondants.

Il est toutefois possible de choisir une solution intermédiaire d'implantation qui soit un compromis entre ces deux cas extrêmes. Les problèmes liés à l'implantation du système de commande constituent une part importante des travaux actuellement en cours au L. A. I. I. et ont déjà abouti à la réalisation de quelques maquettes /BEA 87/ /CRU 87/.

I.4 - Conclusion

La démarche proposée permet de tester différentes configurations de l'unité de production étudiée grâce à une conception **modulaire** et **interactive** du système de commande. Cette méthode pragmatique procède d'une logique réaliste qui permet une assistance informatique très significative en cours de conception.

II - LE PREGRAPHE /GEN 87/ /BOU 87c/ /KAP 87/

II.1 - Introduction

Dans cette partie, nous nous intéressons plus particulièrement à l'élaboration du prégraphe (Etape 3) afin d'en dégager les primitives essentielles et ainsi d'appréhender plus facilement l'étape de structuration, développée au Chapitre III.

Notre présentation se limitera donc ici à la description des règles et de leur traduction en réseau de Petri coloré.

II.2 - Description des règles

L'hypothèse de départ est la suivante :

> Pour chaque type de pièce, toutes les fonctionnalités de l'unité de production étudiée sont décrites par un ensemble de règles de production décrivant les étapes élémentaires des séquences opératoires (gammes).

Ces règles de production ont la structure suivante :

prémisses ⟶ conséquents

où prémisses et conséquents sont constitués d'un ensemble de prédicats du type :

(présence objet lieu)

Ce prédicat est, actuellement, le seul à être utilisé par le système pour élaborer le graphe de commande. Il a permis de décrire convenablement les exemples d'unités de production traités à ce jour.

Nous serons cependant amenés à étendre l'ensemble des prédicats afin de diminuer l'importance de certains traitements d'analyse de l'ensemble des règles. En effet, il est nécessaire parfois d'adjoindre à une règle une information supplémentaire (information associée) qui lèvera toute ambiguïté lors de la structuration du graphe de commande.

Prenons à titre d'exemple, une règle décrivant l'usinage d'une pièce sur une machine. Cette règle aura la même structure qu'une règle décrivant le retournement d'une pièce, sur la même machine, par un robot extérieur.

(présence pièce-brute machine)
⟶ (présence pièce-usinée machine)

(présence pièce machine)
⟶ (présence pièce-retournée machine)

Afin de pallier cet inconvénient, nous utiliserons trois types d'informations associées qui correspondent aux trois types d'actions rencontrées lors de la description de gammes opératoires :

i) les actions de **transformation**, telles que les usinages, pour lesquelles la pièce ne quitte pas le lieu considéré,
ii) les actions **positionnelles**, telles que les transferts, pour lesquelles la pièce quitte le lieu considéré,
iii) les actions que nous appelons "**informationnelles**", dont le résultat ne dépend pas de la partie commande mais, par exemple pour des opérations de métrologie, de l'état de la pièce dans le procédé. Ce dernier cas justifie d'ailleurs la nécessité de disposer d'un modèle du procédé pour la conception du système de commande.

Il est alors possible de décrire complétement et de façon non ambiguë les fonctionnalités du système de production pour chaque type de pièces et d'aborder la phase de traduction en RdPC.

II.3 - Traduction en réseau de Petri coloré

Compte tenu de la nature des règles que nous utilisons, il est possible d'associer à chacune d'elles une traduction en réseau de Petri coloré. En effet, une règle décrit les conditions de franchissement d'une transition par sa partie "prémisses" et les modifications du marquage induites par le franchissement (partie "conséquents" de la règle). De ce fait, nous prendrons comme convention qu'une place du RdPC représente un lieu physique qui ne sera représenté qu'une et une seule fois, et qu'une marque représente un objet dans un état d'avancement donné dans sa gamme opératoire. Ainsi, de manière générale, la Figure 2.4 donne la traduction sous forme de RdPC de la règle suivante :

(présence o1 L1) (présence o2 L2) ... (présence oi Li)
→ (présence p1 M1)
(présence p2 M2)
⋮
(présence pj Mj)

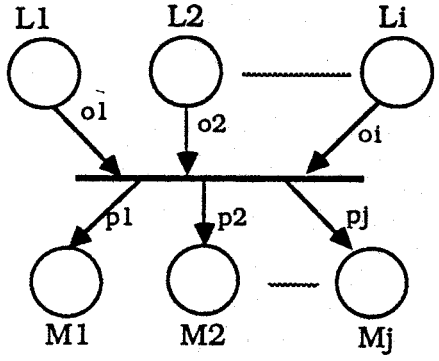


FIGURE 2.4

L'activation de la règle correspond au tir de la transition qui, "physiquement", représente le transfert et la transformation simultanés d'objets.

L'observation des systèmes réels, conduit à proposer quatre règles généralement suffisantes pour décrire la majorité des installations. Nous donnons, ci-après, pour chaque type de règle :

- sa forme (F),
- les informations associées (I),
- son utilisation (U),
- sa traduction en Réseau de Petri Coloré (T).

Type 1

(F) (présence o1 L1) ⟶ (présence o2 L2)

L1 ≠ L2	o1 = o2 ou o1 ≠ o2

(I) Positionnelle

(U) Description de transferts simples (o1 = o2) ou avec repositionnement (o1 ≠ o2)

(T)

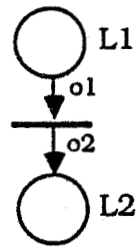


FIGURE 2.5

Type 2

(F) (présence o1 L1) ⟶ (présence o2 L1)

(I)
- Transformation
- Positionnelle
- Informationnelle

(U) (respectivement aux informations associées) :
- Transformations physiques
  (usinages, traitements thermiques ou chimiques, lavage, ...)
- Positionnement par un organe externe
  (robot, bras manipulateur, ...)
- Métrologie : dans ce cas, o2 n'est pas obligatoirement unique mais peut prendre plusieurs valeurs

(T)

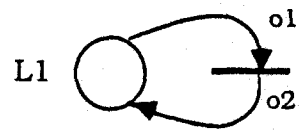


FIGURE 2.6

Type 3

(F) (présence o1 L1) ∧ (présence o2 L2) ⟶ (présence o3 L3)
avec éventuellement : L3 = L2 ou L1

(I) - Positionnelle
- Transformation

(U) - Transfert avec positionnement (L3 ≠ L2 L3 ≠ L1)
- Palettisation (L3 = L1 ou L2)

(T)

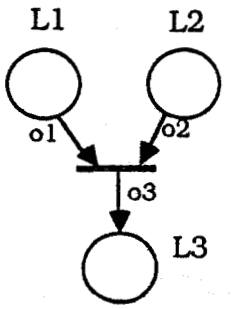


FIGURE 2.7

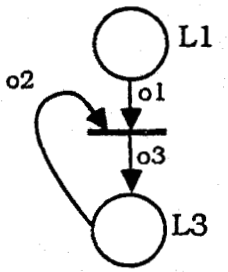


FIGURE 2.8

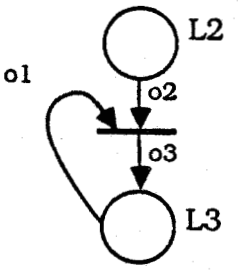


FIGURE 2.9

La Figure 2.8 (resp. Figure 2.9) représente particulièrement un transfert-assemblage comportant une contrainte temporelle :

L'objet o2 (resp. o1) doit être présent sur le lieu L3 **avant** que le transfert de l'objet o1 (resp. o2) puisse être effectué de L1 (resp. L2) vers L3, pour réaliser **ensuite** l'assemblage donnant o3.

Cette traduction présente l'avantage de décrire, par exemple, de manière concise des opérations de palettisation : sur la Figure 2.8

o2 représente une palette,
o1 représente une pièce,
o3 représente la pièce palettisée.

Type 4 (dual du Type 3)

(F) (présence o1 L1) ⟶ (présence o2 L2) ∧ (présence o3 L3)
avec éventuellement : L1 = L2 ou L3

(I)
- Positionnelle
- Transformation

(U)
- Transfert avec positionnement
- Dépalettisation

(T)

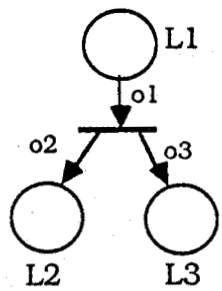


FIGURE 2.10

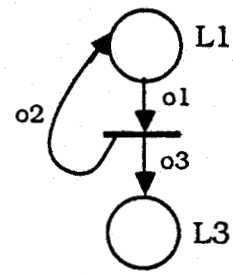


FIGURE 2.11

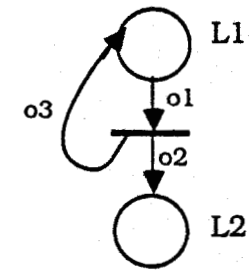


FIGURE 2.12

Sur les Figures 2.11 et 2.12, nous pouvons faire les remarques duales des Figures 2.8 et 2.9 concernant ici l'exemple de la dépalettisation.

A partir de ces quatre types de règles, l'utilisateur peut décrire un grand nombre d'installations. Il peut aussi construire ses propres règles. Ainsi, s'il désire décrire une opération complexe telle que :

Dépalettisation d'une pièce en L1, transfert et assemblage sur une platine d'usinage en L2.

Il introduira la règle correspondante :

(présence pièce-palettisée L1) ∧ (présence platine libre L2)
⟶ (présence palette-libre L1)
(présence pièce-sur-platine L2)

qui sera automatiquement traduite en RdPC (Figure 2.13).

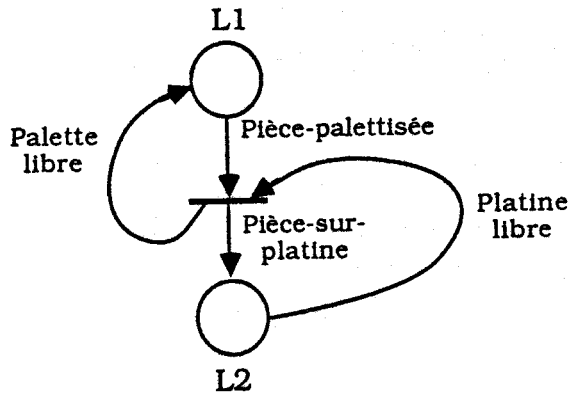


FIGURE 2.13

Après la phase de traduction en réseau de Petri coloré, le système exécute une phase d'agrégation interactive des transitions. L'utilisateur a ainsi la possibilité d'agréger des transitions qui représentent des opérations exécutées par le **même** système physique ayant de plus la **même** structure et les **mêmes** lieux opératoires. Cette agrégation se fait en introduisant des couleurs variables.

Exemple :

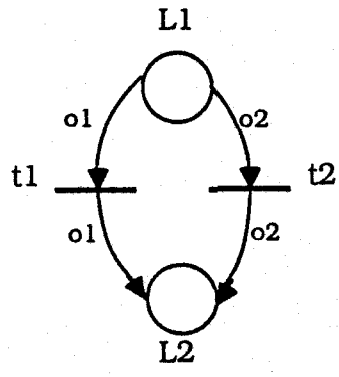
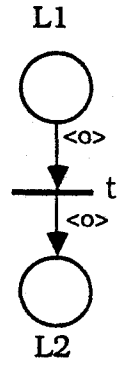


FIGURE 2.14

t1 et t2 représentent des opérations de transfert des pièces o1 et o2 depuis L1 vers L2. Si ces deux opérations sont exécutées par le **même** système de manutention (robot, chariot filoguidé), l'utilisateur peut agréger t1 et t2 pour donner la forme agrégée suivante :



avec dom (<o>) = { o1 , o2 }

FIGURE 2.15

Cette phase d'agrégation achève l'élaboration du prégraphe. Nous obtenons ainsi un modèle intermédiaire décrivant le rôle fonctionnel de l'atelier flexible pour chaque gamme opératoire, sur lequel :

- les places, qui n'apparaissent qu'une seule fois sur le graphe, représentent des lieux physiques par lesquels transitent des objets (pièces, palettes, ...) et où sont effectuées des opérations d'usinage, d'assemblage, d'aiguillage, de transfert, etc...,
- et les transitions représentent des actions effectuées sur les objets du système.

II.4 - Conclusion

Le prégraphe constitue donc, de par sa construction, une base complète, cohérente et concise, permettant d'aborder les problèmes d'élaboration du modèle développé structuré de la partie commande.

En effet, le mode de représentation utilisé, permet de considérer le prégraphe comme une spécification fonctionnelle et formelle du dossier de fabrication mais également comme une première synthèse de la partie procédurale de la commande.

La taille des problèmes rencontrés justifie de toute évidence l'automatisation d'une telle démarche dans le sens du meilleur choix des solutions possibles. Il est en effet exclu de balayer toutes les opportunités de choix dans une démarche traditionnelle.

III - LE SIMULATEUR /CAS 87a/

III.1 - Introduction

L'analyse du système de commande des unités de production flexible nécessite des outils spécifiques permettant, dès la phase de conception, une évaluation tant qualitative que quantitative du système :

- l'évaluation qualitative du système consiste à pouvoir détecter, analyser et résoudre les blocages et indéterminismes induits par une mauvaise conception ou par une définition insuffisante du système de commande,
- l'évaluation quantitative consiste à pouvoir intégrer l'aspect temporel aux divers modèles utilisés afin d'analyser les performances du système et de les optimiser en dimensionnant éventuellement certains paramètres (taille des tampons, nombre de ressources, etc...) ou en modifiant la stratégie adoptée.

Les outils théoriques de validation (étude du graphe de couverture, méthodes d'algèbre linéaire /MEM 80/, ...) fournissent des résultats intéressants sur le "bon fonctionnement" du graphe de commande. Ils sont insuffisants lorsqu'il s'agit :

- de valider des comportements, notamment en régime transitoire (changements de modes de marche très fréquents en production flexible),
- d'évaluer les performances en terme de production,
- d'évaluer différentes stratégies de pilotage par la prise en compte d'un niveau hiérarchique,
- etc ....

Ainsi, la complexité inhérente aux modèles retenus (RdPSAC, règles de production, langages objets) interdit, dans l'état actuel des travaux publiés sur ce thème, toute validation formelle, qualitative et surtout quantitative du système de commande. Ceci justifie pleinement la mise en œuvre de simulateurs intégrant l'ensemble de ces trois modèles (PC, NH, Procédé).

Le simulateur que nous présenterons dans cette partie repose donc sur la modélisation des trois niveaux d'un système de production flexible : partie commande, niveau hiérarchique et procédé. Il est à la fois un outil de validation et d'aide à la conception du niveau hiérarchique dans la mesure où il existe actuellement le seul élément possible d'évaluation des choix stratégiques.

III.2 - Intérêt d'une approche déclarative

Lors de la conception d'un système de production flexible, il est indispensable d'adopter une démarche structurée au sens d'une analyse par niveaux croissants de complexité. Il s'agira, par exemple :

i) de concevoir le mode de marche normal,
ii) d'y intégrer la typologie des objets,
iii) de définir la stratégie du niveau décisionnel,
iv) d'intégrer les différents modes de marche,
v) d'optimiser les performances en tenant compte des objectifs de production.

Cette démarche, généralement effectuée par des méthodes de type essai/erreur, nécessite une convivialité telle qu'il soit possible de compléter ou de corriger aisément le(s) modèle(s) conçu(s).

Cette convivialité est assez restreinte dans des systèmes tels que "OVIDE" du Groupe S.Y.S.E.C.A., le logiciel SLAM ou le simulateur de C. GIROD /GIR 84/. Ces derniers ont l'avantage d'une grande rapidité d'analyse inhérente aux langages **procéduraux** utilisés (FORTRAN, PASCAL, etc...).

Dans le cadre du L. A. I. I., une partie de l'équipe /CAS 87a/ a été chargée de réaliser le simulateur. Elle a, dans ce sens, choisi un langage de type déclaratif, en l'occurence Le_Lisp de l'I.N.R.I.A. comme langage de réalisation du simulateur, de manière à ce que l'utilisateur puisse à tout instant, accéder à l'intégralité de son univers de travail, c'est-à-dire les données relatives aux modèles et la façon de les exploiter.

Le temps d'exécution est certes plus important mais le niveau d'interactivité obtenu compense très efficacement cette relative perte de performances /CAS 85/.

III.3 - Description du simulateur

III.3.1 - Description générale

Ce logiciel de simulation a été réalisé en Le_Lisp, compilé sur VAX 11/750 ou interprété sur compatible IBM-PC puis porté sur Macintosh. Il permet de simuler un ensemble de processus modélisés par RdPAC interprétés au niveau des transitions et temporisés au niveau des places. Les interfaces avec l'environnement du procédé et le niveau hiérarchique sont définies, ainsi qu'un modèle du niveau hiérarchique. L'intégration d'un modèle du procédé est en cours de spécification /CAS 87b/.

Ce logiciel est scindé en 14 modules permettant l'édition, la correction, la sauvegarde et la simulation des différents modèles.

Le réseau de Petri est décrit sous forme de p-listes associées aux places et aux transitions. Pour chaque transition, sont précisés :

- la liste éventuelle des variables libres,
- leur domaine de variation (variables locales),
- la liste des arcs amonts,
- la liste des arcs avals,
- éventuellement, les évènements attendus ou générés tels que "fin de temporisation" par exemple.

La p-liste associée à une place contient :

*** des caractéristiques statiques :**

- son type (pc, po, pd, fifo, rien s'il s'agit d'une place de liaison),
- la liste des transitions en aval,

*** des caractéristques dynamiques :**

- la liste des marques actives,
- la liste des marques gelées par temporisation ou par le niveau hiérarchique.

Cette structure extensible et modulaire basée sur les p-listes a été conçue pour permettre une interactivité maximale. Les modifications ou corrections du modèle reviennent à redéfinir localement les p-listes.

III.3.2 - Configuration du simulateur

Avant toute simulation, l'utilisateur définit le cadre de simulation qu'il souhaite, en jouant sur les paramètres définis dans le statut :

la définition ou non
- du graphe de commande,
- du niveau hiérarchique,
- du procédé,
- des temporisations,
- de time-out
- de macro-places de type fifo.

Le statut du système est modifié lors de la spécification croissante du système. Ainsi, dans la phase de configuration, il est possible d'effectuer une "simulation partielle" en opérant un choix parmi les caractéristiques définies dans le statut et en ne chargeant que les modules correspondants.

III.3.3 - Cycle de simulation /CAS 85a/

Le simulateur combine les deux approches classiques de la simulation : par activité et par évènement /BEL 85/.

La simulation s'interrompt dans les cas suivants :
- le système est bloqué : aucune transition ou règle n'est activable et les échéanciers sont vides,
- un point d'arrêt défini par l'opérateur est atteint,
- un indéterminisme structurel ou d'agrégation a été détecté,
- l'interruption programmée "Intervention Opérateur" a été déclenchée.

Le système entre alors dans une boucle de dialogue permettant la consultation de l'état instantané du réseau. Après d'éventuelles modifications (marquages, règles, structure du graphe, etc...) la simulation peut être reprise au point où elle a été interrompue.

III.4 - Conclusion

Les résultats fournis par la simulation (mise en évidence d'un indéterminisme, d'un blocage, recueil de statistiques) sont exploités et interprétés de manière interactive par l'opérateur. Ils conduisent à définir les modifications à apporter au système afin d'atteindre les objectifs prédéfinis.

De plus, le simulateur utilisé avec un modèle de description du procédé, peut également permettre, lors des phases d'implantation et d'exploitation du système réel, de détecter les erreurs ou les défaillances des organes de commande par comparaison de l'évolution du système simulé avec celle du système réel.

CONCLUSION

Dans ce chapitre, nous avons proposé une méthodologie permettant de fournir un ensemble de logiciels chargés d'assurer l'analyse, la synthèse et la validation du système de commande de cellules de production flexible. L'ambition du L. A. I. I. est de couvrir l'ensemble du processus de conception de la phase de spécification du cahier des charges jusqu'à la phase d'implantation sur le site.

Afin de mieux appréhender la phase d'élaboration automatique du modèle procédural développé décrivant la partie commande, nous avons présenté deux éléments charnières de la méthodologie proposée :

- l'élaboration d'un modèle intermédiaire intégrant l'ensemble des gammes opératoires : le prégraphe,
- la simulation interactive permettant d'effectuer une étude détaillée du comportement dynamique du système complet ainsi qu'une évaluation quantitative de ses performances.

Ces deux éléments constituent respectivement les points d'entrée et de sortie des outils logiciels qui seront présentés dans le chapitre suivant.

CHAPITRE III

INTRODUCTION

Nous présentons, dans ce chapitre, la phase de modélisation du projet C.A.S.P.A.I.M. et plus particulièrement le logiciel chargé de la réalisation de cette phase et de l'interconnexion avec le simulateur /CAS 87/.

Ce logiciel a pour but d'assister la construction des trois modèles du système (PC, NH, Procédé) à partir du prégraphe, modèles qui sont indispensables à une analyse quantitative et qualitative du comportement dynamique du procédé et de sa commande.

Dans ce sens, nous effectuerons une présentation générale du logiciel, puis nous aborderons successivement :

- la structuration du prégraphe,
- la répartition des systèmes de commande,
- l'intégration du niveau hiérarchique et du procédé.

I - PRESENTATION GENERALE DU LOGICIEL

Le logiciel, qui est scindé en sept modules (Fig. 3.1) correspondant aux étapes 4 à 7 et 9 à 11 du projet C.A.S.P.A.I.M., permet :

i) la création du modèle Réseau de Petri Structuré Adaptatif et Coloré de la partie commande à partir du prégraphe,

ii) l'intégration du procédé par temporisation des places du réseau.

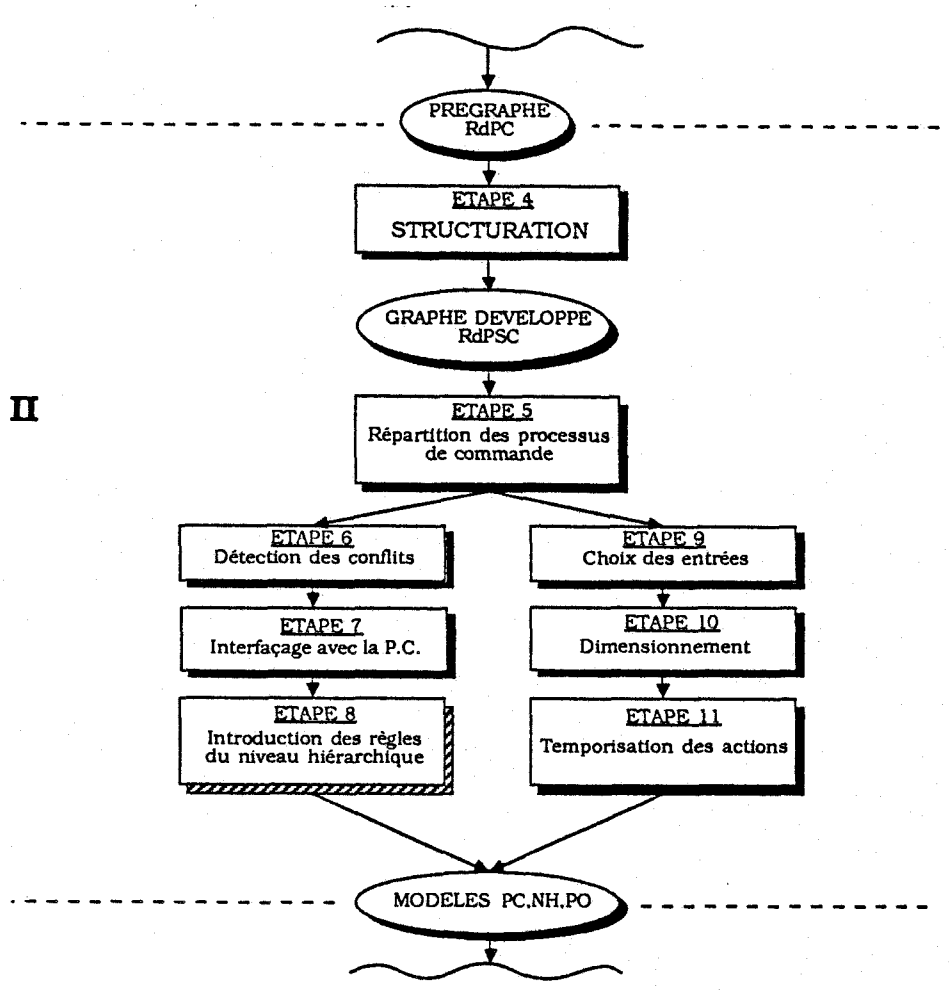


FIGURE 3.1

Remarque : Rappelons que l'introduction des règles du niveau hiérarchique et l'intégration d'un modèle spécifique du procédé, basé sur l'utilisation de langages orientés objet, constituent une part importante des travaux développés actuellement au L. A. I. I. et ne sont donc pas pris en charge par le logiciel présenté dans ce chapitre.

Le logiciel de simulation ayant été réalisé en Le_Lisp, deux solutions étaient envisageables pour la réalisation de notre logiciel : la première consistait à écrire les programmes dans un langage procédural et à les interfacer avec le simulateur ; la seconde, à écrire directement les différents modules en Le_Lisp.

Après une étude des problèmes d'interfaçage de Le_Lisp avec d'autres langages, la seconde solution s'est avérée plus intéressante pour prendre en compte facilement les différentes possibilités de retour arrière dans la méthodologie proposée.

Nous avons donc écrit notre logiciel en Le_Lisp (Version 15.2) interprété sur VAX 11/750 puis nous l'avons porté sur MACINTOSH et IBM PC sur lesquels existait une version du simulateur.

II - STRUCTURATION DU PREGRAPHE

II.1 - Introduction

Le but de cette étape (Etape 4 du projet C.A.S.P.A.I.M.), est de déduire du prégraphe un modèle **développé** et **structuré** permettant une réelle prise en charge du parallélisme dans un fonctionnement concurrent des tâches élémentaires.

L'originalité de notre démarche réside dans le fait que, contrairement aux méthodes de modélisation qui consistent à agréger le graphe grâce à l'utilisation d'un modèle de haut niveau (réseau de Petri stochastique /BAL 87/ ou réseaux de Petri colorés à prédicats /DES 85/ /MAR 87/ par exemple), nous déduisons du prégraphe un modèle **développé.** Ceci permet de mettre en évidence le parallélisme du système ainsi que la concurrence entre les tâches élémentaires (processus).

L'expression du modèle initial sous forme d'un réseau de Petri développé conduit à une structuration du modèle, nécessitée par notre objectif d'automatisation de la démarche de conception et d'implantation du modèle. La structuration du modèle permet, en effet, de faciliter :

- la visualisation des chemins de couleur et donc la visualisation des chemins suivis par les objets dans le système,
- la traduction en un code implantable grâce au nombre limité de primitives structurées (graphe de processus et de liaisons) utilisées dans la description du modèle,
- les modifications a posteriori (ajout des modes dégradés) grâce à la modularité de la description,
- la maintenance logicielle,
- etc, ...

De ce fait, la structuration du modèle accroît la **flexibilité** du système de commande dans le sens de la reconfiguration des organes de pilotage. Ceci est d'autant plus important qu'un système de production n'est jamais figé mais est sujet à des modifications en cours d'exploitation, ajout ou modification de gammes opératoires, par exemple. Il convient toutefois de noter la nécessité de prévoir a priori l'existence de ces gammes.

II.2 - Principe de structuration utilisé

Rappelons tout d'abord que notre ensemble de données initiales est fourni sous la forme du modèle **prégraphe**. Il s'agit d'un réseau de Petri coloré pour lequel :

- les transitions représentent des actions effectuées sur les objets (pièces, palettes, ...),
- les places représentent des lieux physiques par lesquels transitent des objets et/ou sur lesquels s'effectuent des opérations de transformation ou de transfert d'objets.

Le principe de structuration utilisé repose sur une analyse fonctionnelle du prégraphe.

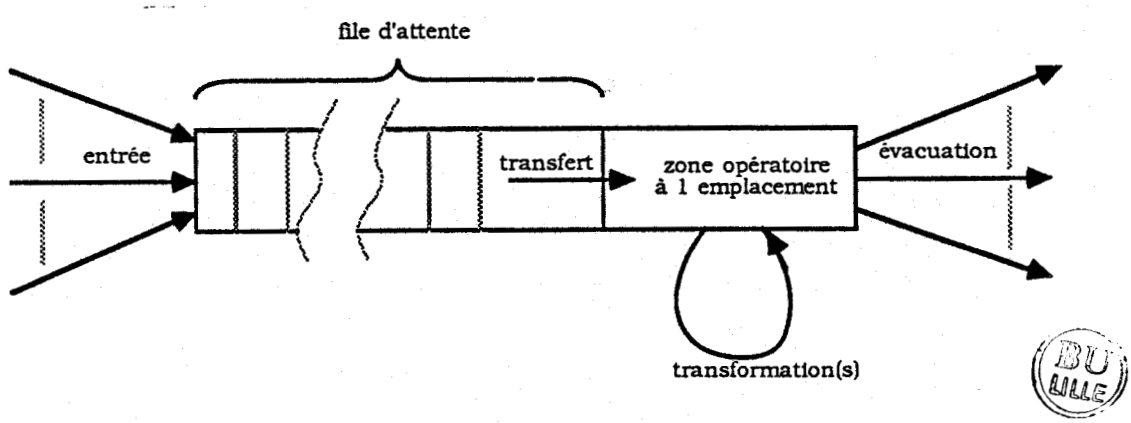
L'étude de cas concrets d'installations flexibles industrielles nous a conduit à décomposer les lieux physiques en :

- une file d'attente à n places ($n \geq 0$),
- une zone appelée "zone opératoire" sur laquelle sont effectuées, le cas échéant, une ou plusieurs opérations de transformation. Cette zone a une capacité d'une place.

De même, il est possible d'effectuer une décomposition de la séquence des opérations que subissent les objets sur de telles zones selon les primitives suivantes :

- entrée d'objet(s) dans la file d'attente,
- transfert d'un objet sur la zone opératoire,
- transformation(s) de l'objet,
- évacuation vers d'autres lieux d'objet(s) résultant de la transformation.

Nous pouvons résumer ces deux décompositions par la Figure 3.2 :



Décomposition d'une place P

FIGURE 3.2

Cette approche pragmatique de la description des lieux physiques permet, comme nous le verrons par la suite, de modéliser facilement :

- des zones de stockage intermédiaire,
- des machines réalisant des transformations sur les objets,
- etc, ...

en jouant sur la taille des différentes zones (opératoire et file d'attente) et sur la modélisation de la (des) transformation(s) subie(s) par les pièces.

A partir de cette décomposition fonctionnelle des lieux représentés par des places sur le prégraphe, nous construisons le modèle RdPSC correspondant, qui est représenté sur le schéma de la Figure 3.3 :

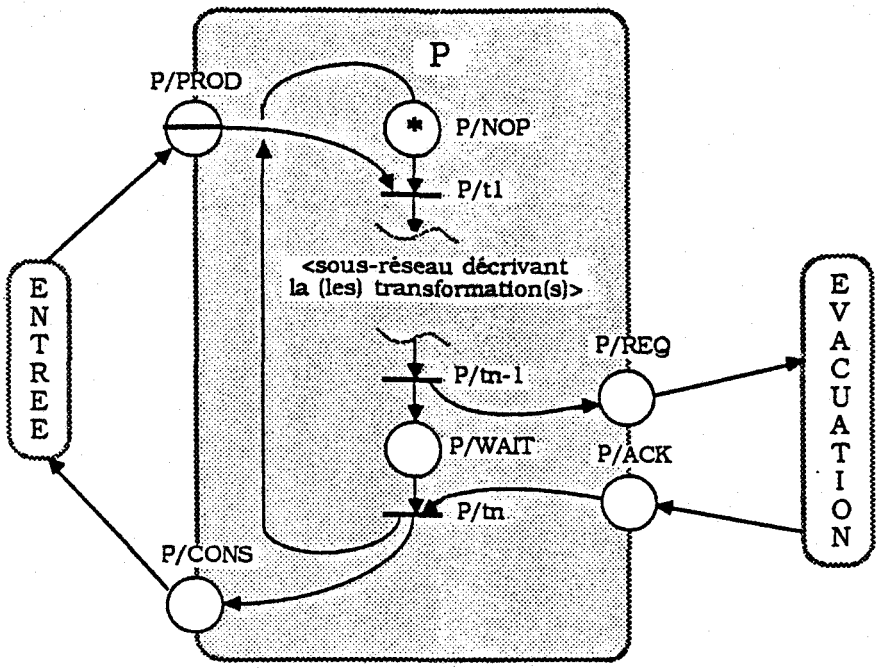


FIGURE 3.3

Ce RdP est construit de telle façon que :

i) la liaison producteur/consommateur représente la file d'attente dont la taille effective sera précisée lors de l'étape de dimensionnement. Dans cette liaison, le nombre de marques de la place P/CONS indique aux processus d'entrée, le nombre d'emplacements disponibles dans la file d'attente, tandis que la macro-place P/PROD contient les marques représentant les objets présents dans cette file,

ii) le processus P assure le transfert **d'un objet** depuis la file d'attente jusqu'à la zone opératoire, ainsi que la gestion de la transformation de l'objet,

iii) la liaison de synchronisation avec accusé de réception permet d'indiquer aux processus chargés de l'évacuation qu'un objet est prêt à être évacué. Lorsque l'objet est évacué, les processus d'évacuation émettent un accusé de réception (place P/ACK). Ceci permet de libérer la zone opératoire, donc d'autoriser le transfert d'une nouvelle pièce depuis la file d'attente en libérant un emplacement dans la file (ajout d'une marque dans la place P/CONS après le tir de la transition P/t_n),

iv) les processus d'entrée et d'évacuation d'objets sont des processus de transfert entre deux lieux du système (cf. § II.3.3).

Les liaisons introduites ici permettent d'assurer un fonctionnement sûr et efficace des tâches concurrentes avec la tâche P.

Dans la suite de cette partie, nous proposons un catalogue des primitives de structuration basées sur cette description des lieux physiques.

II.3 - Description des primitives de structuration

II.3.1 - Notations

Le logiciel que nous avons mis au point permet de générer un graphe développé et par incidence d'accroître considérablement le nombre de places et de transitions. De ce fait, nous avons pris un certain nombre de conventions relatives à la dénomination des objets "graphiques" créés. Certaines de ces conventions seront justifiées au fur et à mesure de la description des primitives de structuration.

Considérons un processus de nom P.

Ce nom sera constitué du nom de l'élément du prégraphe auquel il correspond et d'une partie spécifique.

Les places de ce processus seront notées P/<nom spécifique de place> où <nom spécifique de place> correspond à la désignation de l'action engendrée (ex : P/USINAGE).

Les transitions seront notées P/t_i ($i \in \{ 1, ..., n \}$).

Le cas échéant, les places de liaisons producteur/consommateur en amont seront notées (P/PROD, P/CONS) et celles des liaisons de synchronisation en aval (P/REQ, P/ACK).

II.3.2 - Les zones de stockage intermédiaire

Il s'agit de zones sur lesquelles les objets ne subissent pas d'opération de transformation. C'est le cas, par exemple, d'une zone de stockage de palettes vides. Cette zone est définie sur un convoyeur par le positionnement d'une butée (Fig. 3.4).

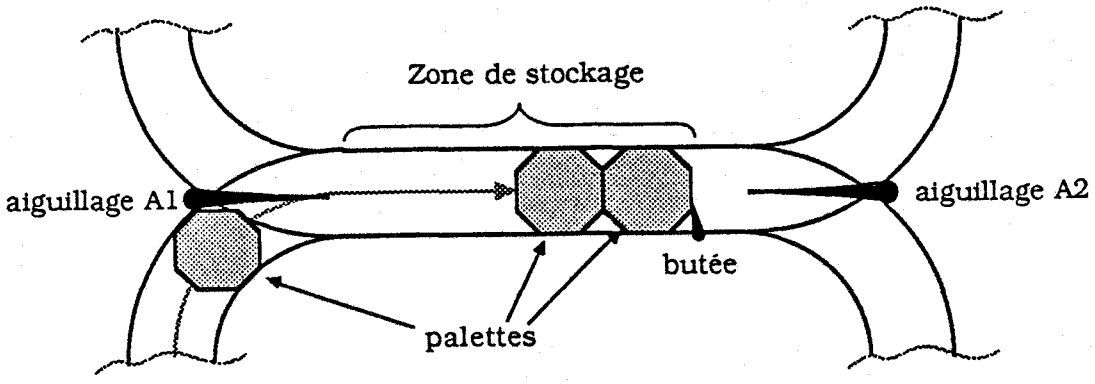


FIGURE 3.4

La représentation de telles zones sur le prégraphe est donnée sur le schéma de la Figure 3.5 sur laquelle :

* $E_i \longrightarrow T$ (resp. $T \longrightarrow S_j$) correspond au développement structuré du processus de transfert des objets de E_i à T (resp. T à S_j).

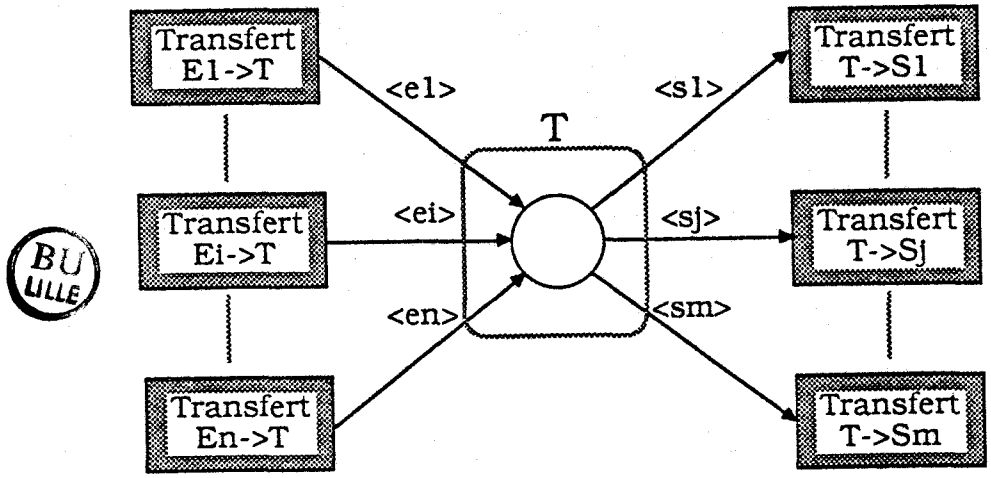


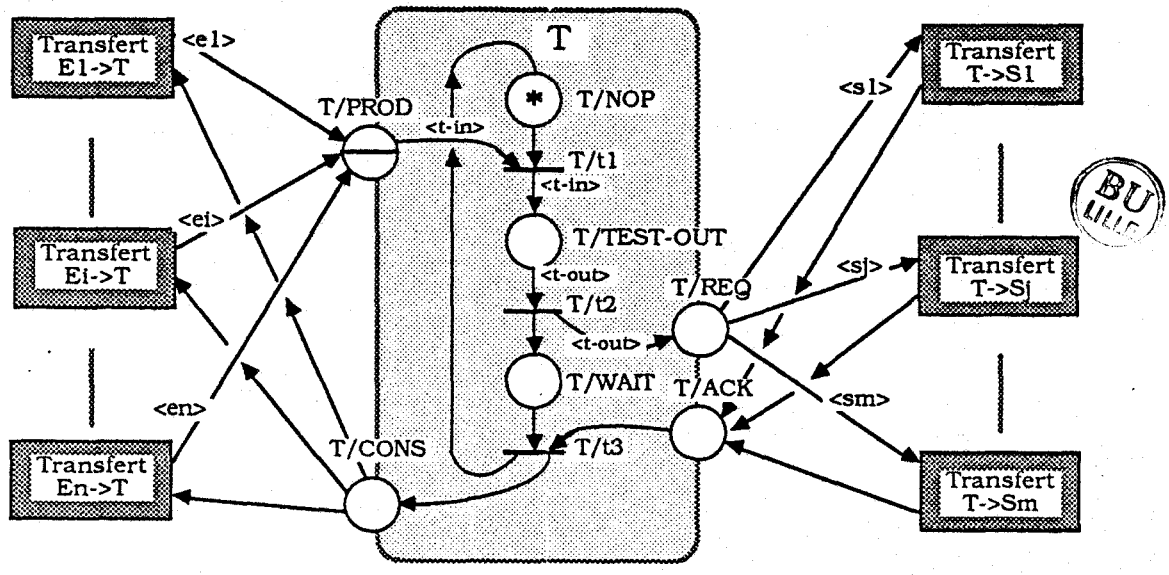
FIGURE 3.5

* $\langle e_i \rangle$ (resp. $\langle s_j \rangle$) est une variable dont le domaine de couleurs représente l'ensemble des types d'objets transférés de E_i vers T (resp. T vers S_j).

Etant donné que T est une zone de stockage intermédiaire, tout objet qui entre dans cette zone en sort à un niveau d'avancement identique dans sa gamme opératoire. De ce fait, il n'y a pas de transformation d'objet, donc pas de transformation de couleur. Ceci se traduit par l'égalité suivante :

$$\bigcup_{i=1}^{n} \text{dom}(\langle e_i \rangle) = \bigcup_{j=1}^{m} \text{dom}(\langle s_j \rangle)$$

Le RdP structuré, qui est donné sur le schéma de la Figure 3.6, a été obtenu en utilisant le principe du § II.2.



$$\text{dom}(\langle t\text{-in} \rangle) = \text{dom}(\langle t\text{-out} \rangle) = \bigcup_{i=1}^{n} \text{dom}(\langle e_i \rangle)$$

FIGURE 3.6

Comme nous le verrons, ce schéma correspond à la primitive de base que nous utiliserons pour le développement en RdP Structuré d'un grand nombre de places du prégraphe.

Dans ce RdPS, nous avons créé une place, notée T/TEST-OUT, qui n'est pas utilisée pour des zones de stockage simple mais qui s'avère intéressante dans l'optique d'une modification a posteriori du logiciel de commande lors de l'exploitation de l'installation. En effet, si la zone de stockage est transformée en station d'assemblage, par exemple, nous ajouterons aisément, au niveau de la place TEST-OUT, une "boucle d'assemblage" (cf. § II.3.5). Ceci permet d'accroître la **flexibilité** de reconfiguration du système de commande généré.

De plus, la nature des liaisons utilisées en entrée (PROD/CONS) et en sortie (REQ/ACK) nous amène à faire les remarques suivantes :

i) La liaison (T/REQ, T/ACK) permet d'aiguiller en sortie **une** pièce vers **une** zone et une seule. Ce type de liaison génère donc un conflit lorsque $\exists$ i,j/dom(<si>) $\cap$ dom(<sj>) $\neq \emptyset$ (cf (§ IV.2.2.d).

ii) La liaison (T/PROD, T/CONS) ne permet pas quant à elle, d'éviter que deux transferts aient lieu simultanément. Afin d'éviter ces collisions, nous générons automatiquement une exclusion mutuelle, notée "mutex-entrée/<nom-de-zone>" sur **tous** les processus de transfert intervenant dans cette liaison. Cette liaison d'exclusion mutuelle est construite de la même façon que les liaisons traduisant les différentes contraintes inhérentes au système de transport (cf § III.3.2).

II.3.3 - Les transferts

Il s'agit de transferts d'objets d'un lieu à un autre. Afin d'illustrer la primitive de passage du prégraphe au réseau de Petri structuré correspondant à de tels transferts, considérons l'exemple schématisé par la Figure 3.7. Cet exemple concerne le transfert d'objets de type o_1, o_2, ..., o_n depuis un lieu L1 jusqu'au lieu L2. Ce transfert est assuré par un robot manipulateur R.

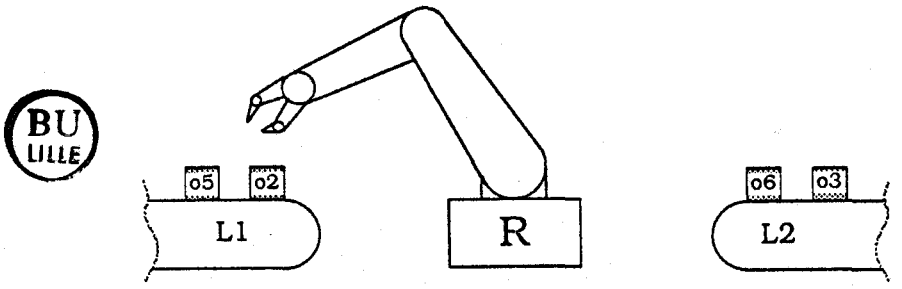
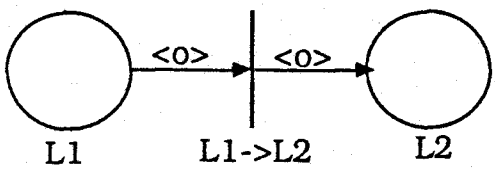


FIGURE 3.7

L'analyse de ce système simple, nous donne le modèle "prégraphe" issu de la traduction de l'application en RdPC de règles de type 1 (cf Chapitre II, § II.3).



dom (<o>) = { o_1, o_2, ..., o_n }

FIGURE 3.8

Remarque importante : Sur ce modèle, la transition L1 ⟶ L2 ne modèlise pas le robot en tant que système de transport, mais seulement l'action de transfert d'objets d'un point à un autre. Cette transition impose seulement que les objets o_1, o_2, ..., o_n soient transférés par le **même système de transport** (robot, convoyeur, chariot filo-guidé, ...).

Nous décrivons ci-après chaque phase de l'opération de transfert et nous en déduisons le graphe structuré correspondant.

1ère Phase :

Pour que la transfert ait lieu, il faut :

i) qu'il existe un objet à transférer depuis L1,

ii) qu'il existe un emplacement vide sur L2.

Compte tenu du principe de structuration explicité dans le paragraphe II.2, cette première phase se traduit par :

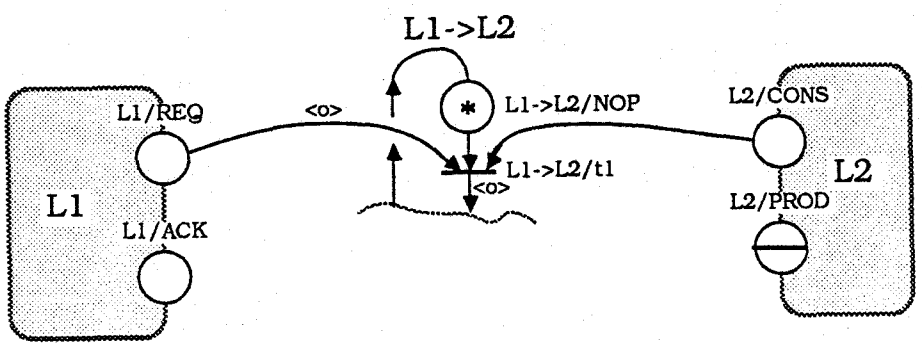


FIGURE 3.9

2ème Phase :

La décision de transfert ayant été prise, le transfert peut débuter : sur notre exemple, le robot quitte sa position de repos, saisit l'objet à transférer et commence son mouvement vers L2.

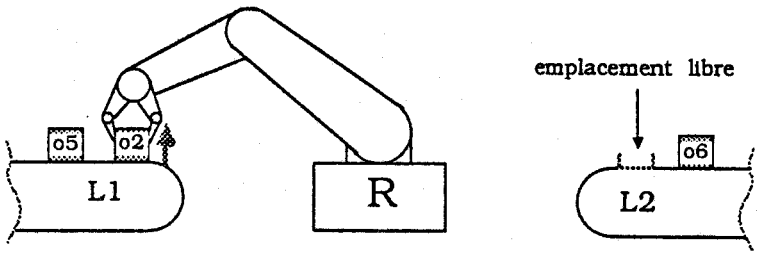


FIGURE 3.10

Cette phase correspond à la place notée "trsf-start" sur le RdP structuré de la Figure 3.11.

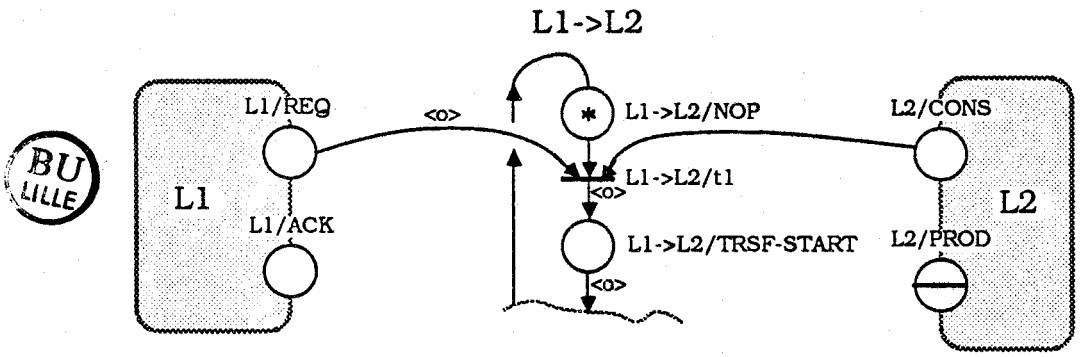


FIGURE 3.11

Cette phase se prolonge jusqu'à ce que le système de transport ait quitté la zone de travail critique, permettant ainsi à l'objet suivant de L1, d'être transféré sur la zone opératoire.

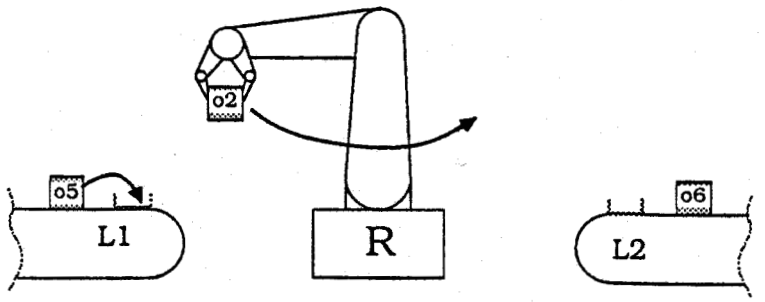


FIGURE 3.12

Ceci se traduit sur le graphe structuré, par l'émission de l'accusé de réception vers L1.

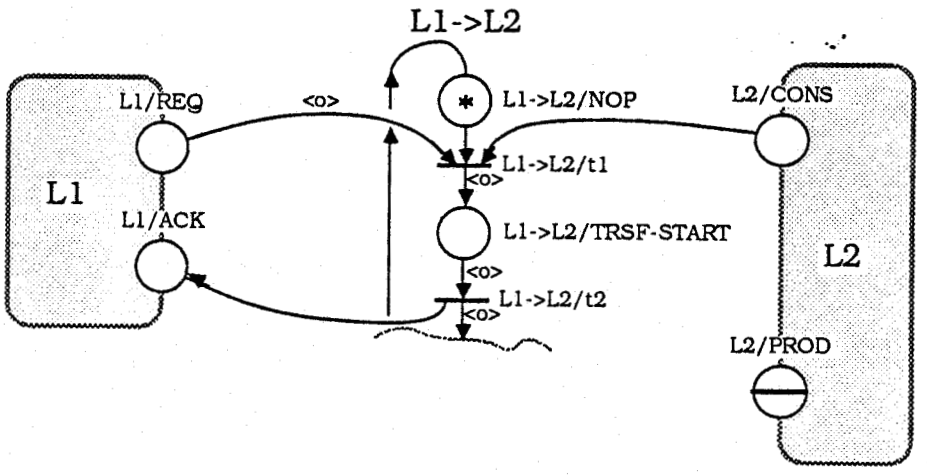


FIGURE 3.13

3ème Phase :

Cette phase est constituée par la fin de transfert, le dépôt de l'objet sur le lieu de destination et le retour du système de transport dans la position de repos (dans le cas d'un robot). L'objet est alors dans la file d'attente du lieu de destination et un nouveau transfert peut débuter (retour la la Phase 1).

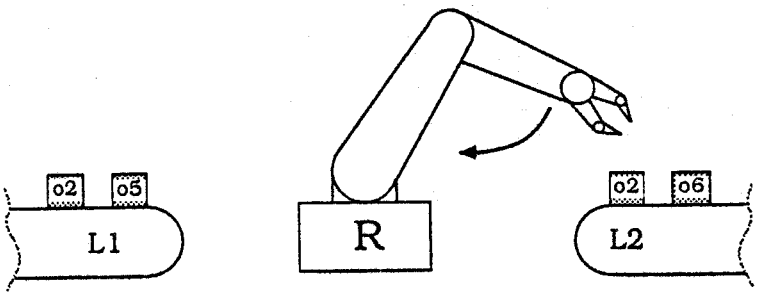


FIGURE 3.14

Cette phase correspond à la phase notée "trsf-end" sur le RdP structuré de la Figure 3.15.

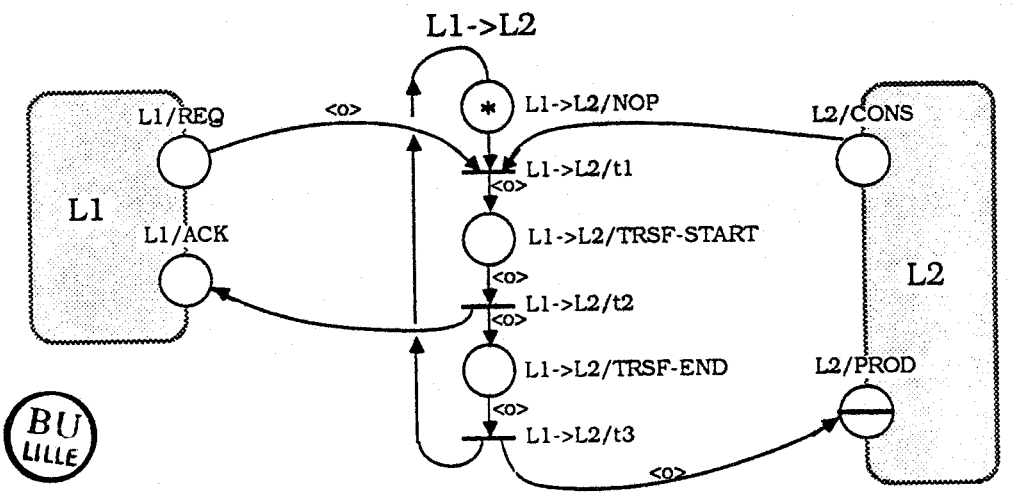


FIGURE 3.15

La Figure 3.15 constitue donc le réseau de Petri structuré correspondant au développement de la transition de transfert de la Figure 3.8.

De plus, ce type de développement sera utilisé pour la traduction en RdPS de chaque **opération** de transfert que nous rencontrerons par la suite. La prise en compte des contraintes imposées par le système de transport (robot assurant deux opérations de transfert différentes) ne sera effectuée que dans l'étape de répartition des systèmes de commande (cf § III).

Cet exemple illustre bien notre double préoccupation de gérer à la fois le parallélisme le moins contraignant pour les tâches L1 et L2 et la sûreté d'exécution de L1 et L2 en présence de la liaison L1 → L2.

II.3.4 - Les machines

Nous nous intéressons dans ce paragraphe à la structuration des primitives du prégraphe issues de la traduction en RdPC et de l'agrégation de règles de type 2 (cf Chapitre II, § II.3) qui concernent les **transformations physiques** et les **opérations de métrologie** sur des **machines.** Le RdPC correspondant est rappelé sur le schéma de la Figure 3.16 :

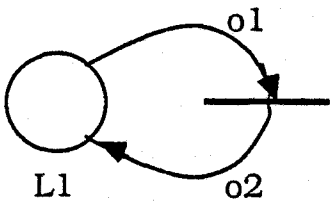


FIGURE 3.16

Notations :

Dans la suite de ce paragraphe, nous appellerons :

- M, la machine représentée par la place,
- USINAGE, l'opération de transformation représentée par la transition,
- <b>, <av>, <ap>, <u>, les variables dont les domaines de couleurs correspondent respectivement aux types de pièces :
  - . brutes arrivant sur la machine,
  - . avant usinage,
  - . après usinage,
  - . usinées quittant la machine.

  Cette distinction entre les différents types de pièces intervient dans le cas d'usinages successifs.

Avec ces conventions, la Figure 3.16 devient donc :

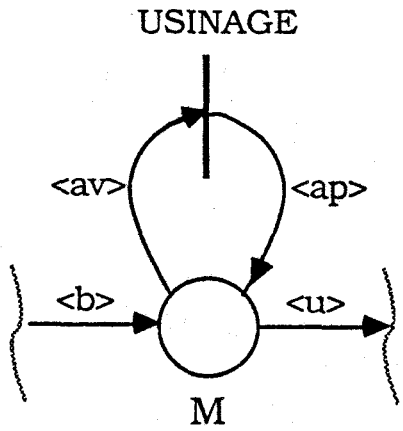


FIGURE 3.17

De manière évidente, nous avons :

$$dom(<b>) \subseteq dom(<av>) \quad \text{et} \quad dom(<u>) \subseteq dom(<ap>)$$

Notre objet est maintenant de déduire de ce schéma un RdP structuré décrivant l'enchaînement des opérations à effectuer pour différentes architectures matérielles de machines (selon le nombre et la nature de la gestion des tampons d'entrée/sortie).

Dans ce sens, nous proposons quatre primitives de structuration correspondant aux quatre types de machines les plus couramment rencontrés dans les installations industrielles.

Pour chacune d'elles, nous donnerons le schéma physique et le RdPS correspondant qui a été obtenu en respectant la méthode de structuration explicitée dans le paragraphe II.2.

a) Les machines sans tampon d'entrée/sortie

Ces machines ne sont en fait constituées que d'un poste d'usinage (Fig. 3.18) :

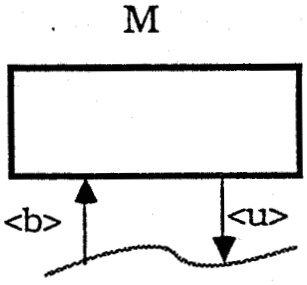


FIGURE 3.18

Pour ce type de machine, le graphe structuré développé est donné Figure 3.19 :

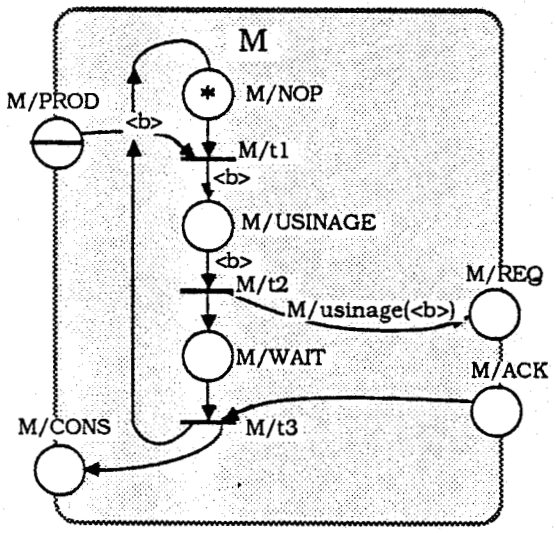


FIGURE 3.19

Sur ce RdPSC, la fonction M/USINAGE (<b>) correspond au regroupement fonctionnel des usinages tel qu'il a été défini au Chapitre I, § I.3.2.b.

b) Les machines à un tampon d'entrée/sortie

Cette machine dispose d'un tampon noté ES, par lequel la pièce à usiner doit transiter avant d'être transférée sur le poste d'usinage noté P. Elle est ensuite usinée. Après l'usinage, la pièce est transférée sur le tampon puis, selon sa gamme opératoire, quitte la machine ou est de nouveau usinée.

Le schéma physique est donné Figure 3.20 et le réseau de Petri structuré correspondant Figure 3.21.

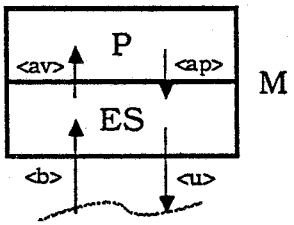


FIGURE 3.20

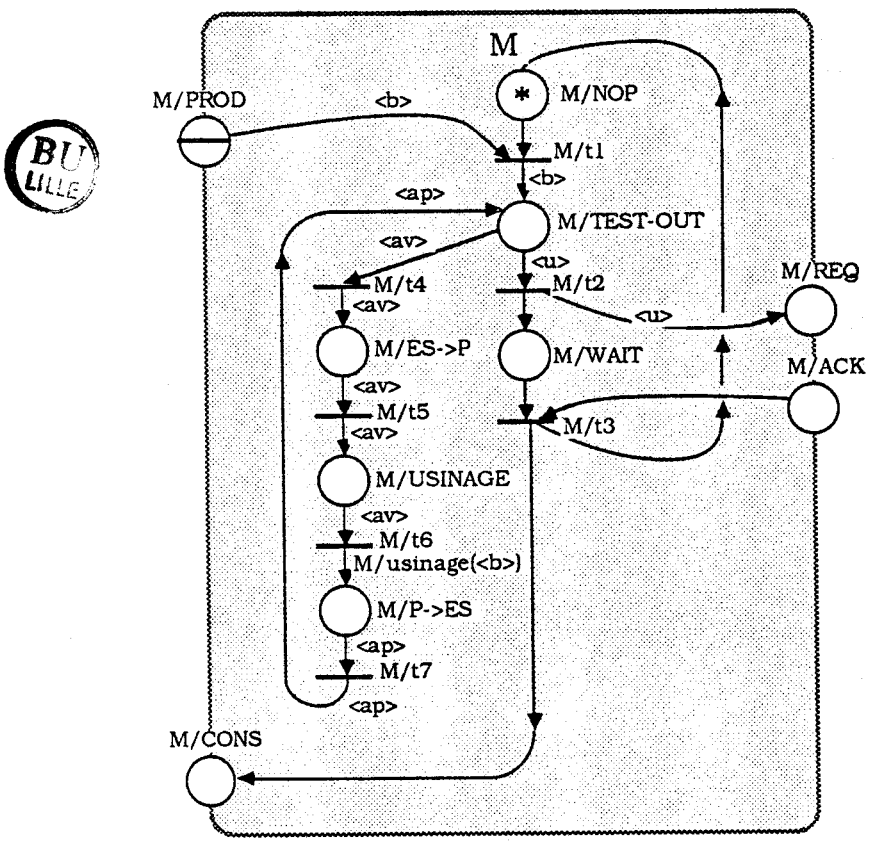


FIGURE 3.21

La structure répétitive que nous avons utilisée, nous permet de décrire simplement toutes les gammes d'usinages constituées d'une succession d'usinages et de transferts sur le tampon d'entrée/sortie.

c) Les machines à plusieurs tampons d'entrée/sortie équivalents

Ces machines correspondent au schéma de la Figure 3.22 :

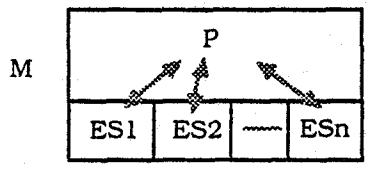


FIGURE 3.22

Ce type de machine peut être considéré comme l'association de n machines du type précédent avec une contrainte d'unicité de l'accès au poste d'usinage.

Nous construirons donc n réseaux de Petri semblables à celui de la Figure 3.21 et nous traduirons la contrainte d'accès par une ressource critique notée m/ressource sur la séquence d'opération $ES_i \longrightarrow P$, USINAGE, $P \longrightarrow ES_i$.

Nous donnons (Fig. 3.23) le réseau de Petri structuré correspondant à une machine à deux tampons d'entrée/sortie.

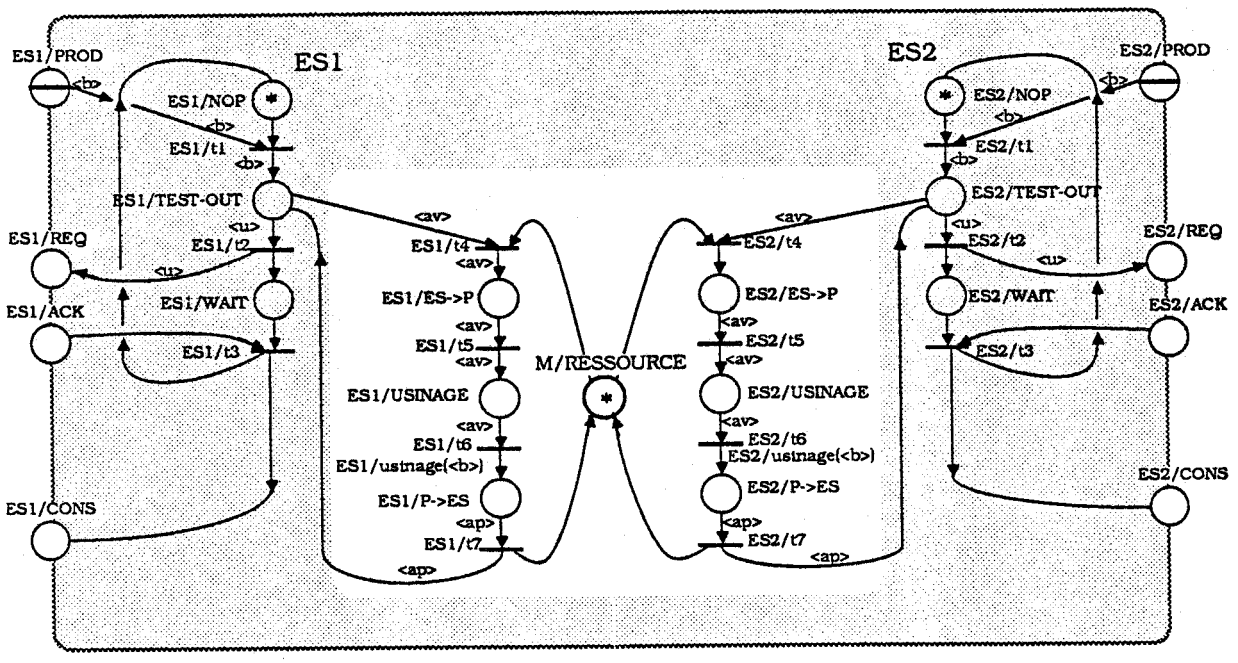


FIGURE 3.23

d) Les machines à un tampon d'entrée et un tampon de sortie

Ce type de machine diffère légèrement des types précédents dans le sens où les tampons d'entrée et de sortie peuvent être de taille supérieure à 1. Nous avons donc pris en compte ce type de configuration schématisé par la Figure 3.24 :

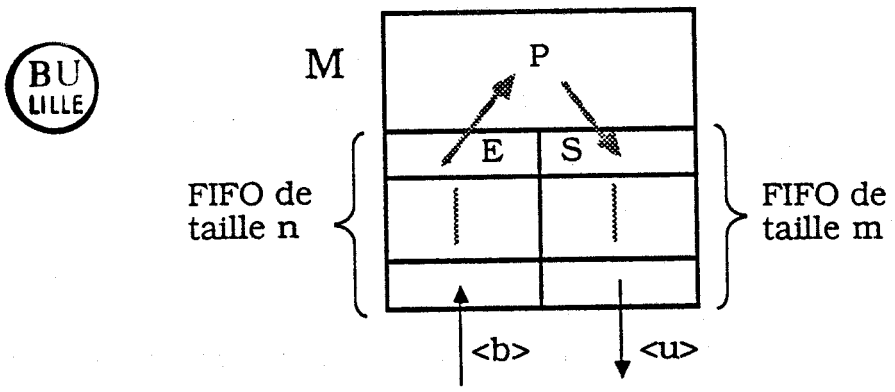


FIGURE 3.24

Pour créer le graphe structuré, nous avons décomposé la machine en deux zones distinctes :

- le tampon d'entrée et le poste d'usinage,
- le tampon de sortie.

En gardant le principe de structuration défini dans le paragraphe II.2, nous aboutissons au RdPSC de la Figure 3.25 :

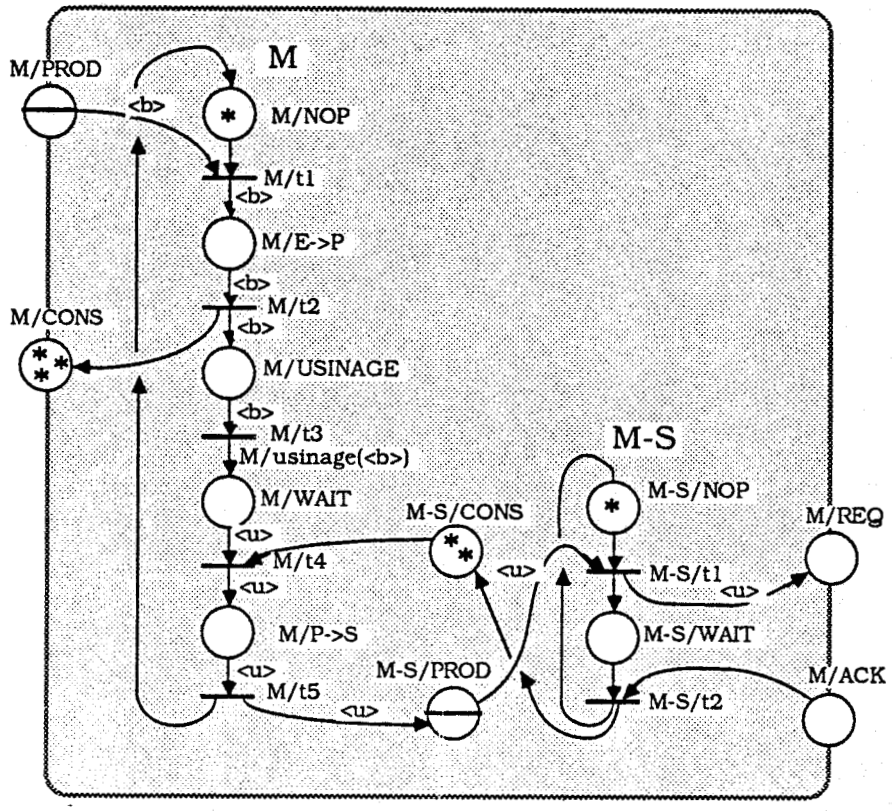


FIGURE 3.25

La tâche M-S permet d'arbitrer le transfert en sortie du tampon S.

Cette décomposition nous permet de distinguer les deux tampons et, de ce fait, de leur attribuer une taille (nombre de marques des places "/CONS") adéquate dans le cas où la configuration matérielle n'est pas totalement définie.

II.3.5 - Les assemblages et les désassemblages

Dans ce paragraphe, nous nous intéresserons à la structuration des primitives du prégraphe issues de la traduction en RdPC et de l'agrégation des règles de types 3 et 4 (cf Chapitre II, § II.3). Plus particulièrement, nous présenterons les primitives de structuration des opérations de palettisation et dépalettisation qui correspondent à des assemblages avec contraintes d'antériorité (i.e. : présence de la palette avant l'arrivée de la pièce à palettiser).

a) Les palettisations

Ces opérations correspondent au séquencement suivant :

- arrivée d'une palette "pa" sur la zone opératoire appelée ici zone d'assemblage de la station d'assemblage "SA",
- transfert, depuis un lieu "PI", de la pièce à palettiser "pi" sur la palette,
- assemblage de l'ensemble pièce-palette "pp",
- départ de l'ensemble "pp" de la zone d'assemblage.

Cet enchaînement, représenté schématiquement Figure 3.26, est modélisé sur le prégraphe par la Figure 3.27.

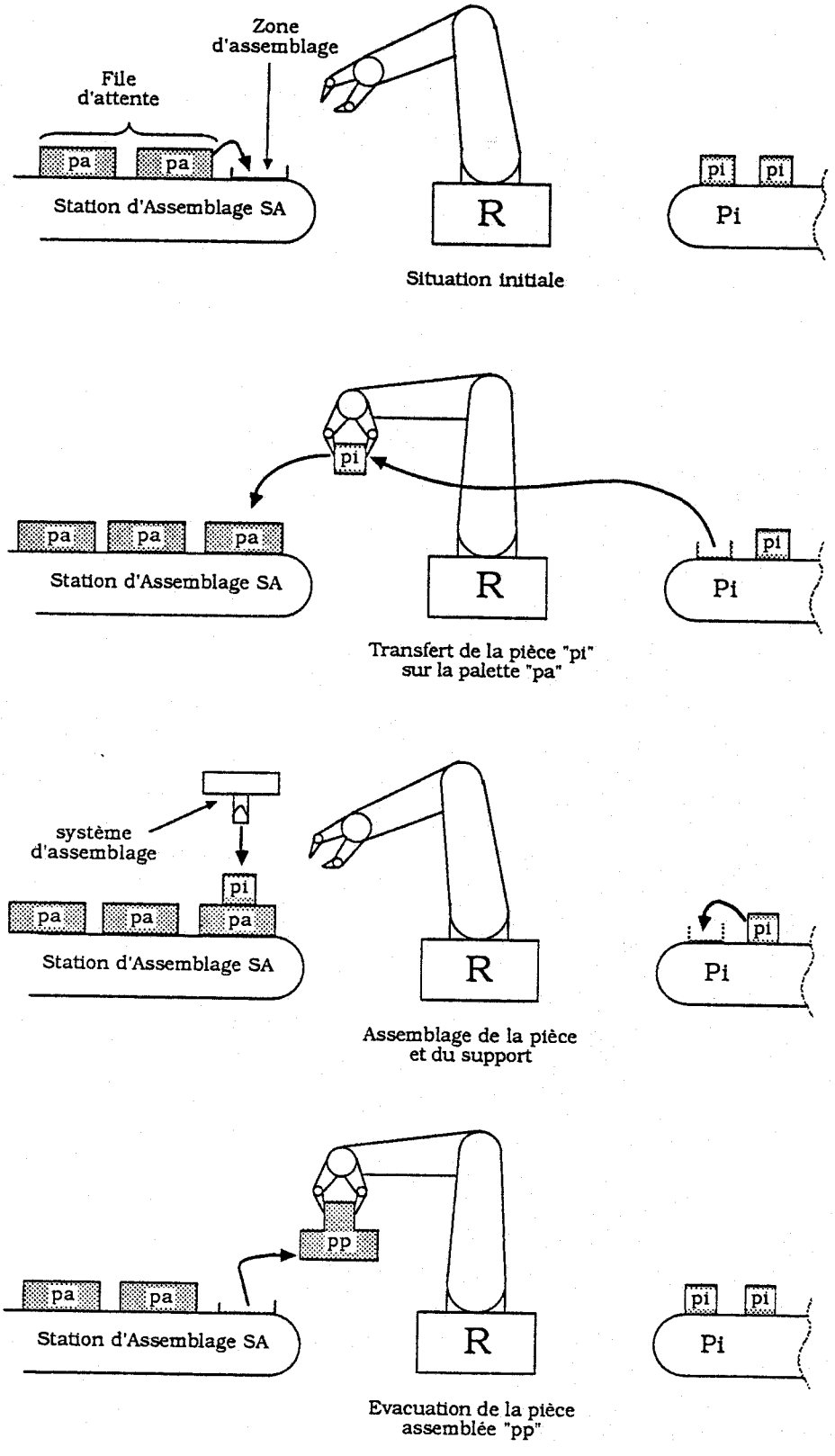

Zone
d'assemblage
File
d'attente
pa
pa
Station d'Assemblage SA
R
pi
pi
Pi
Situation initiale
pi
pa
pa
pa
Station d'Assemblage SA
R
pi
Pi
Transfert de la pièce "pi"
sur la palette "pa"
système
d'assemblage
pi
pa
pa
pa
Station d'Assemblage SA
R
pi
Pi
Assemblage de la pièce
et du support
pp
pa
pa
Station d'Assemblage SA
R
pi
pi
Pi
Evacuation de la pièce
assemblée "pp"

FIGURE 3.26

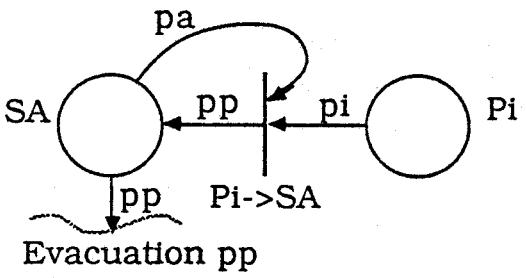


FIGURE 3.27

En décomposant les différentes opérations (arrivée de la palette sur la zone d'assemblage, requête pour le transfert de la pièce, palettisation, évacuation de la pièce palettisée), nous obtenons le graphe structuré de la Figure 3.28.

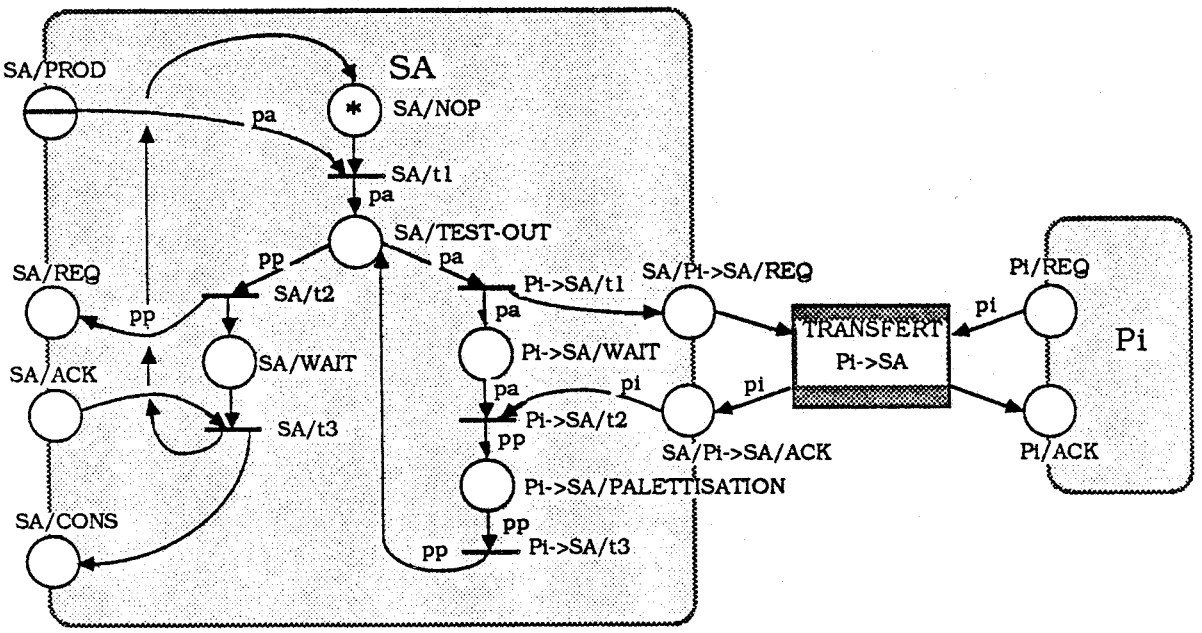


FIGURE 3.28

La méthode de construction de ce RdPS nous amène à faire les remarques suivantes :

i) La partie gauche correspond au développement d'une zone tampon simple, ce qui permet, lors d'un passage dans un mode dégradé par exemple, de transformer la station d'assemblage en zone de stockage intermédiaire,

ii) La structure "répétitive" peut être généralisée à une structure "répétitive multiple" pour des stations d'assemblage sur lesquelles sont palettisées des pièces provenant de **lieux différents** (Fig. 3.29 et 3.30) ou sur lesquelles sont effectuées des opérations d'assemblages successifs avec contraintes d'antériorité (Fig. 3.31 et 3.32).

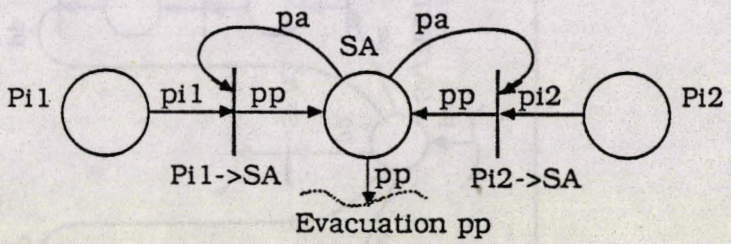


FIGURE 3.29

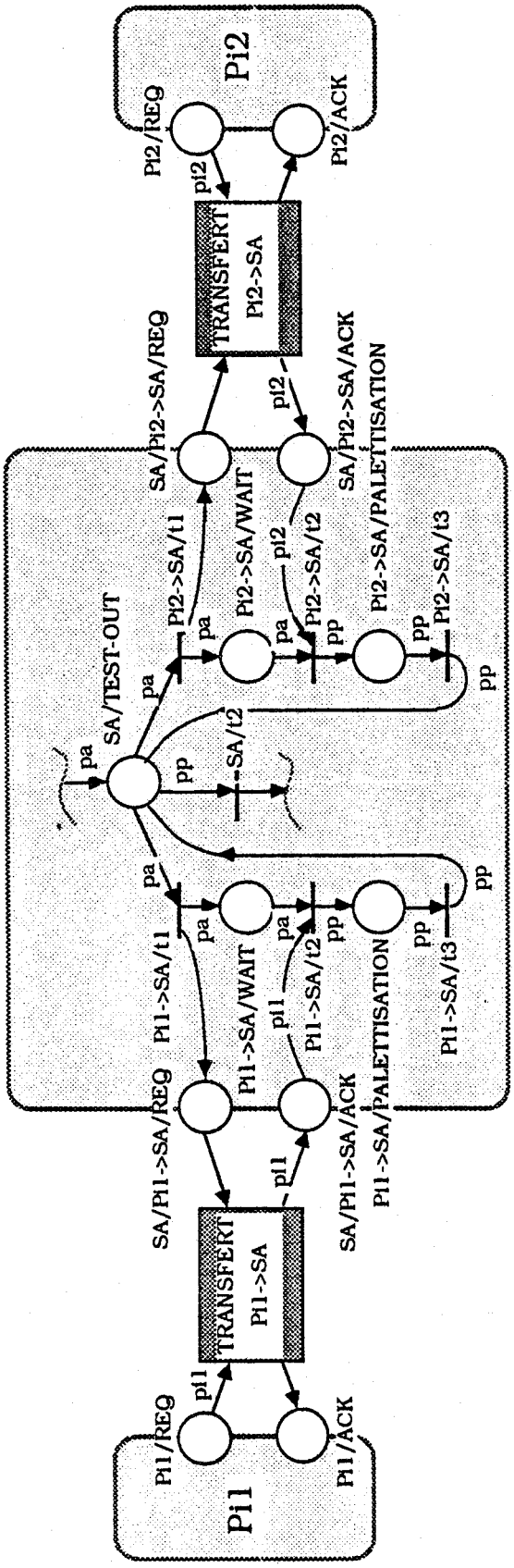

Pi2
Pi2/REQ
Pi2/ACK
pi2
TRANSFERT
Pi2->SA
SA/Pi2->SA/REQ
SA/Pi2->SA/ACK
Pi2->SA/t1
Pi2->SA/WAIT
Pi2->SA/t2
Pi2->SA/PALETTISATION
Pi2->SA/t3
SA/TEST-OUT
SA/t2
pa
pp
Pi1->SA/t1
Pi1->SA/WAIT
Pi1->SA/t2
Pi1->SA/PALETTISATION
Pi1->SA/t3
SA/Pi1->SA/REQ
SA/Pi1->SA/ACK
pi1
TRANSFERT
Pi1->SA
Pi1/REQ
Pi1/ACK
Pi1

FIGURE 3.30

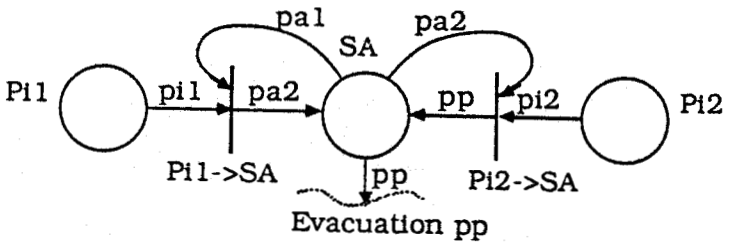


FIGURE 3.31

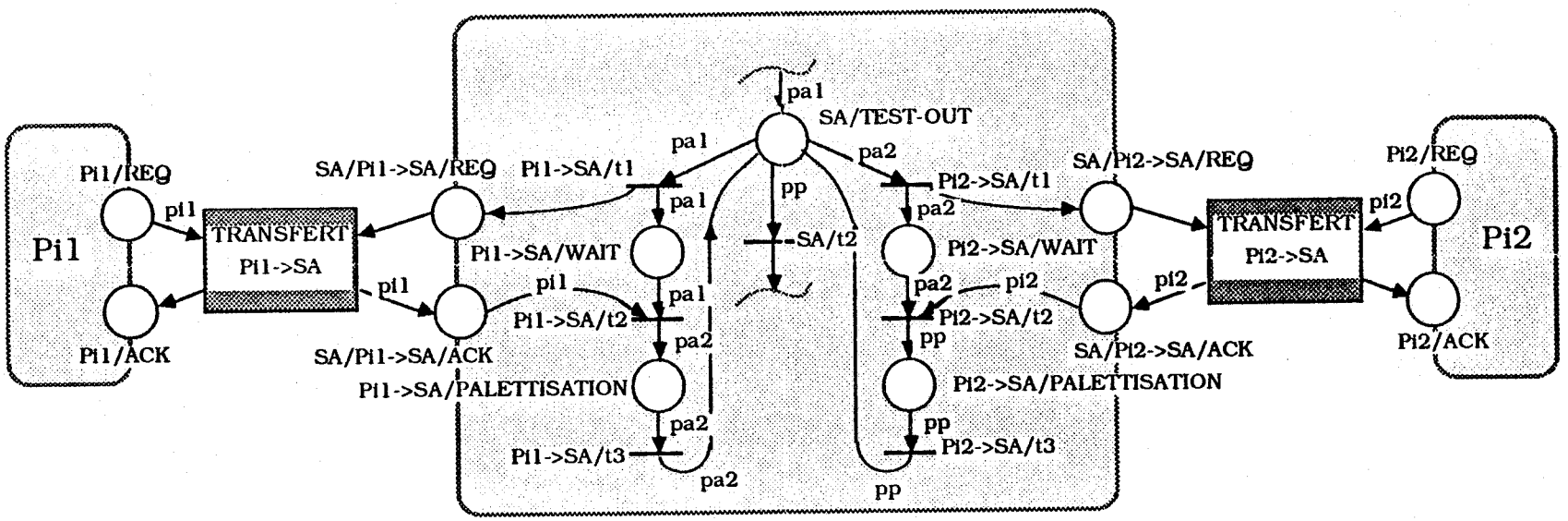

Pi1
Pi1/REQ
pi1
TRANSFERT
Pi1->SA
Pi1/ACK
SA/Pi1->SA/REQ
SA/Pi1->SA/ACK
Pi1->SA/t1
Pi1->SA/WAIT
Pi1->SA/t2
Pi1->SA/PALETTISATION
Pi1->SA/t3
pa1
pa2
pp
SA/TEST-OUT
SA/t2
Pi2->SA/t1
Pi2->SA/WAIT
Pi2->SA/t2
Pi2->SA/PALETTISATION
Pi2->SA/t3
SA/Pi2->SA/REQ
SA/Pi2->SA/ACK
TRANSFERT
Pi2->SA
pi2
Pi2/REQ
Pi2/ACK
Pi2

FIGURE 3.32

iii) Les liaisons de synchronisation vers les processus de transfert peuvent être dupliquées afin de décrire des assemblages sur une **même** palette de **plusieurs** pièces arrivant de **lieux différents** et dans un ordre quelconque (Fig. 3.33 et 3.34).

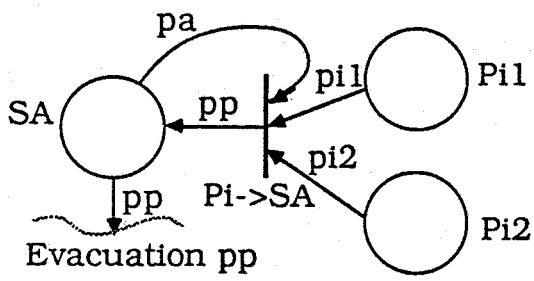


FIGURE 3.33

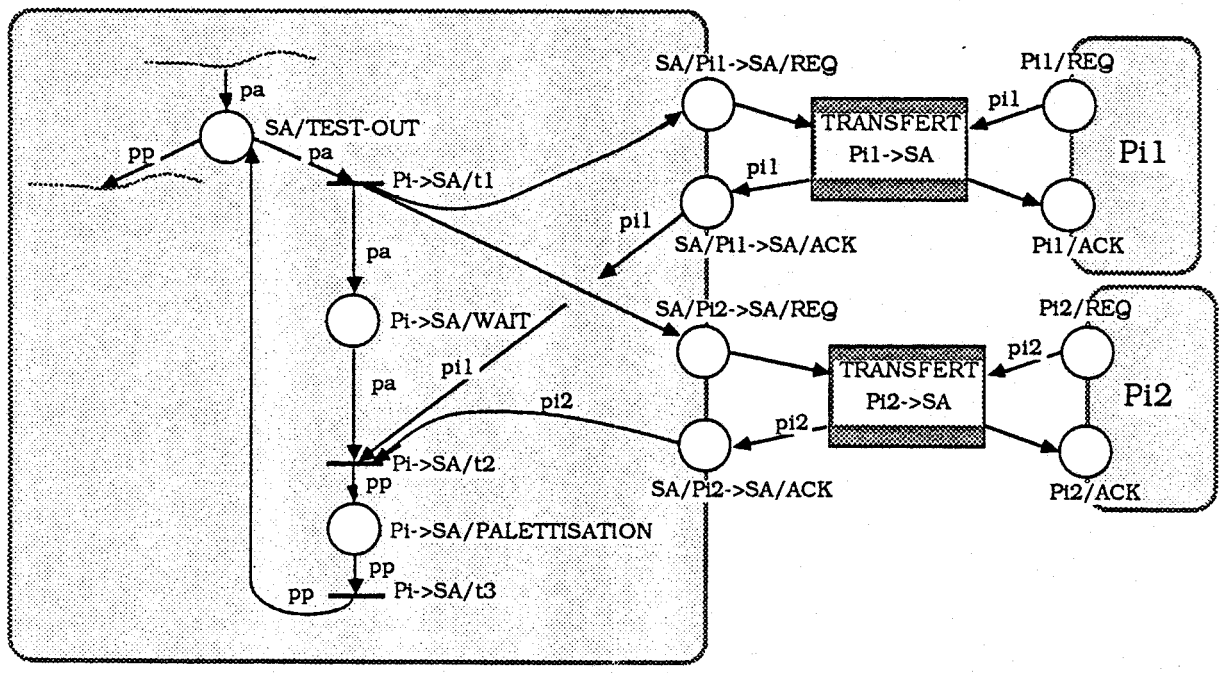
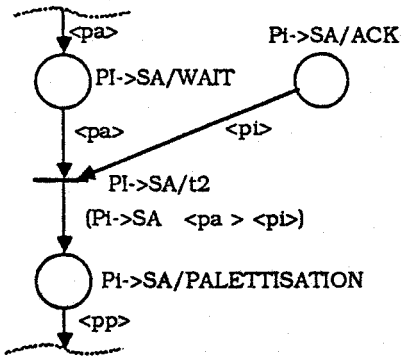


FIGURE 3.34

iv) Le type de palette et de pièce peut être variable. C'est le cas, par exemple, d'une station d'assemblage sur laquelle sont palettisées **différentes** pièces sur **différentes** palettes. Pour prendre en charge de tels cas de figure, nous introduisons des variables de couleurs sur les arcs concernés et une fonction de palettisation qui met en correspondance les palettes, les pièces et les pièces palettisées. Cette fonction peut être définie, en Le_Lisp par exemple, de la manière suivante :



avec : dom (<pa>) = { pa_1, ..., pa_n }
dom (<pi>) = { pi_1, ..., pi_n }
dom (<pp>) = { pp_1, ..., pp_p } $p \leq n \times m$
et
(de PI → SA / palettisation (pa pi)
(cond ((and (equal pa 'pa_1) (equal pi 'pi_1)) 'pp_1)
((and (equal pa 'pa_2) (equal pi 'pi_2)) 'pp_2)
⋮
((and (equal pa 'pa_n) (equal pi 'pi_m)) 'pp_p)))

Toutes ces remarques sont prises en charge par notre logiciel qui construit automatiquement le RdP après une analyse de la (ou des) transition(s) de palettisation.

b) Les dépalettisations

Il s'agit de l'opération inverse de la précédente qui est représentée sur le prégraphe par la Figure 3.35.

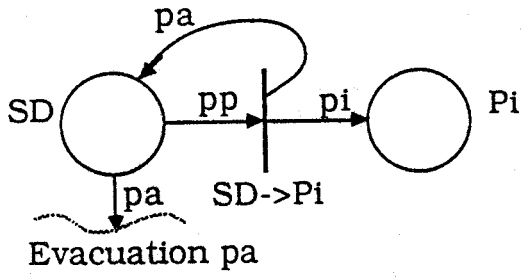


FIGURE 3.35

Le réseau de Petri structuré correspondant est obtenu de la même façon que précédemment :

- création de la zone tampon SD,
- ajout de la structure "répétitive" gérant le désassemblage et l'évacuation de la pièce.

Nous obtenons donc le graphe structuré de la Figure 3.36 :

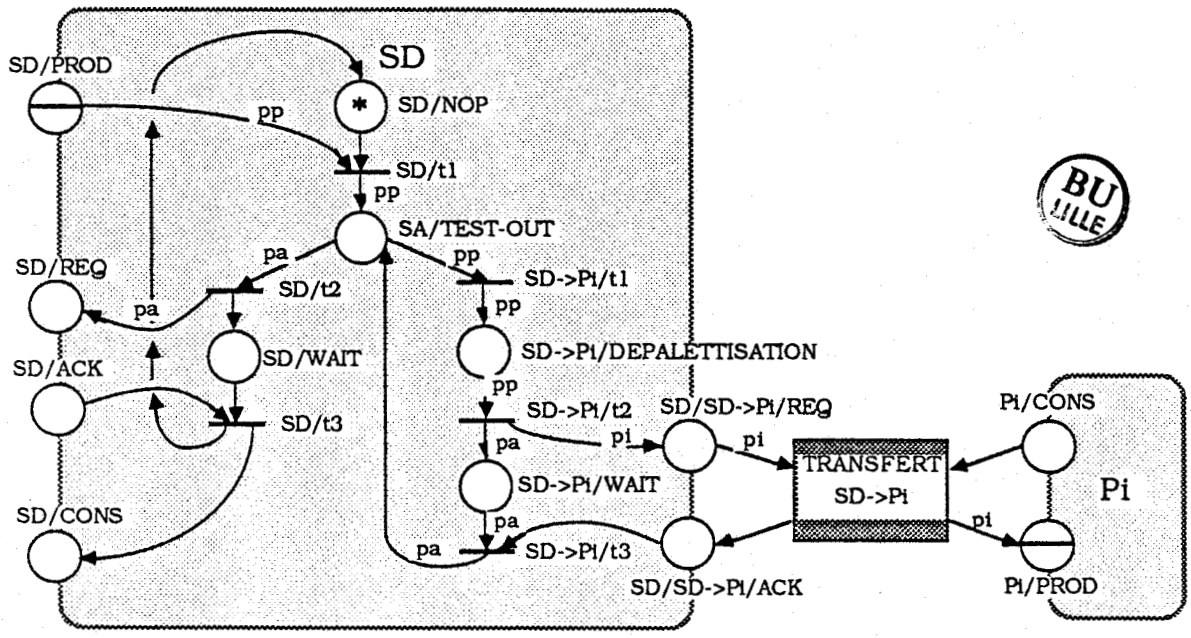


FIGURE 3.36

De même que pour les palettisations, notre logiciel permet de construire des graphes qui prennent en charge :

- des structures "répétitive-multiple" (Fig. 3.37 et 3.38),
- des structures "répétitive-multiple" pour des désassemblages successifs (Fig. 3.39 et 3.40),
- des dépalettisations de plusieurs pièces (Fig. 3.41 et 3.42),
- des variables de couleurs.

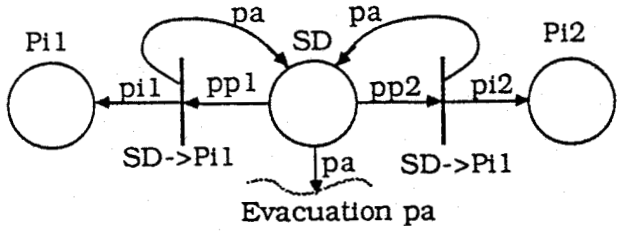


FIGURE 3.37

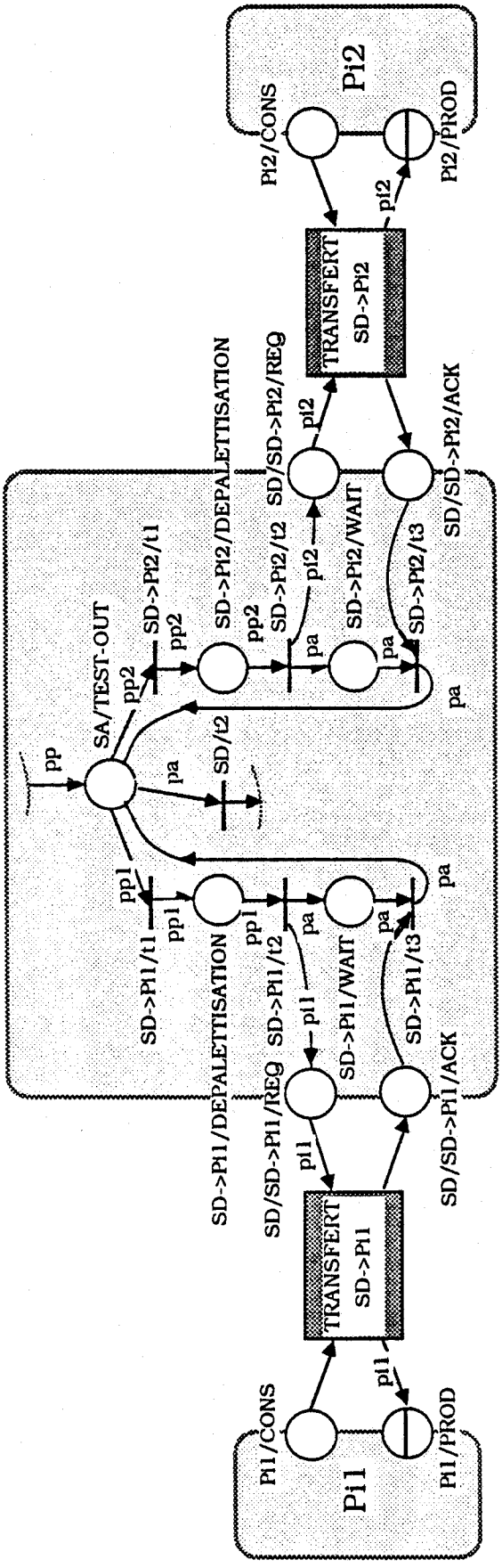

Pi2
Pi2/CONS
Pi2/PROD
pi2
TRANSFERT
SD->Pi2
SD/SD->Pi2/REQ
SD/SD->Pi2/ACK
SD->Pi2/DEPALETTISATION
SA/TEST-OUT
SD->Pi2/t1
SD->Pi2/t2
SD->Pi2/WAIT
SD->Pi2/t3
pp
pp2
pa
SD/t2
pp1
SD->Pi1/t1
SD->Pi1/t2
SD->Pi1/WAIT
SD->Pi1/t3
SD->Pi1/DEPALETTISATION
SD/SD->Pi1/REQ
SD/SD->Pi1/ACK
pi1
TRANSFERT
SD->Pi1
Pi1/CONS
Pi1
Pi1/PROD

FIGURE 3.38

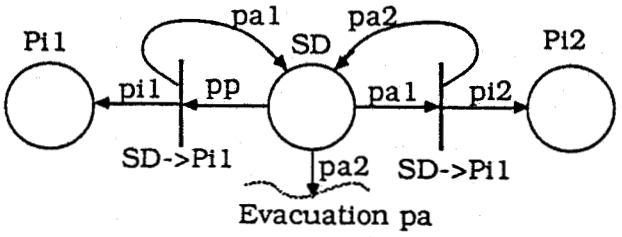


FIGURE 3.39

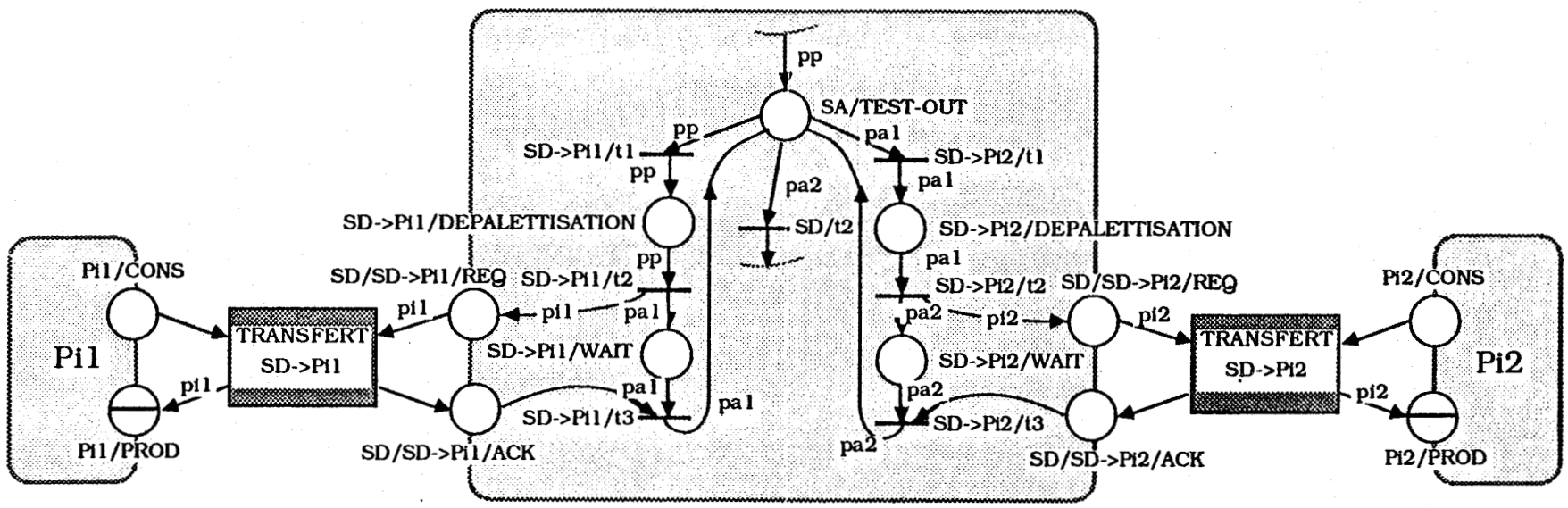

pp
SA/TEST-OUT
pa1
pa2
SD->Pi1/t1
SD->Pi2/t1
SD->Pi1/DEPALETTISATION
SD/t2
SD->Pi2/DEPALETTISATION
Pi1/CONS
SD/SD->Pi1/REQ
SD->Pi1/t2
SD->Pi2/t2
SD/SD->Pi2/REQ
Pi2/CONS
pi1
pi2
TRANSFERT
SD->Pi1
SD->Pi1/WAIT
SD->Pi2/WAIT
TRANSFERT
SD->Pi2
Pi1
Pi2
SD->Pi1/t3
SD->Pi2/t3
Pi1/PROD
SD/SD->Pi1/ACK
SD/SD->Pi2/ACK
Pi2/PROD

FIGURE 3.40

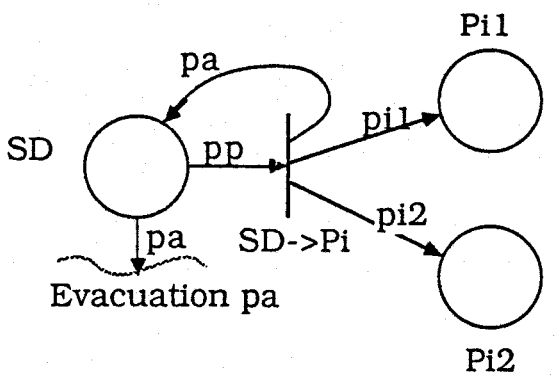


FIGURE 3.41

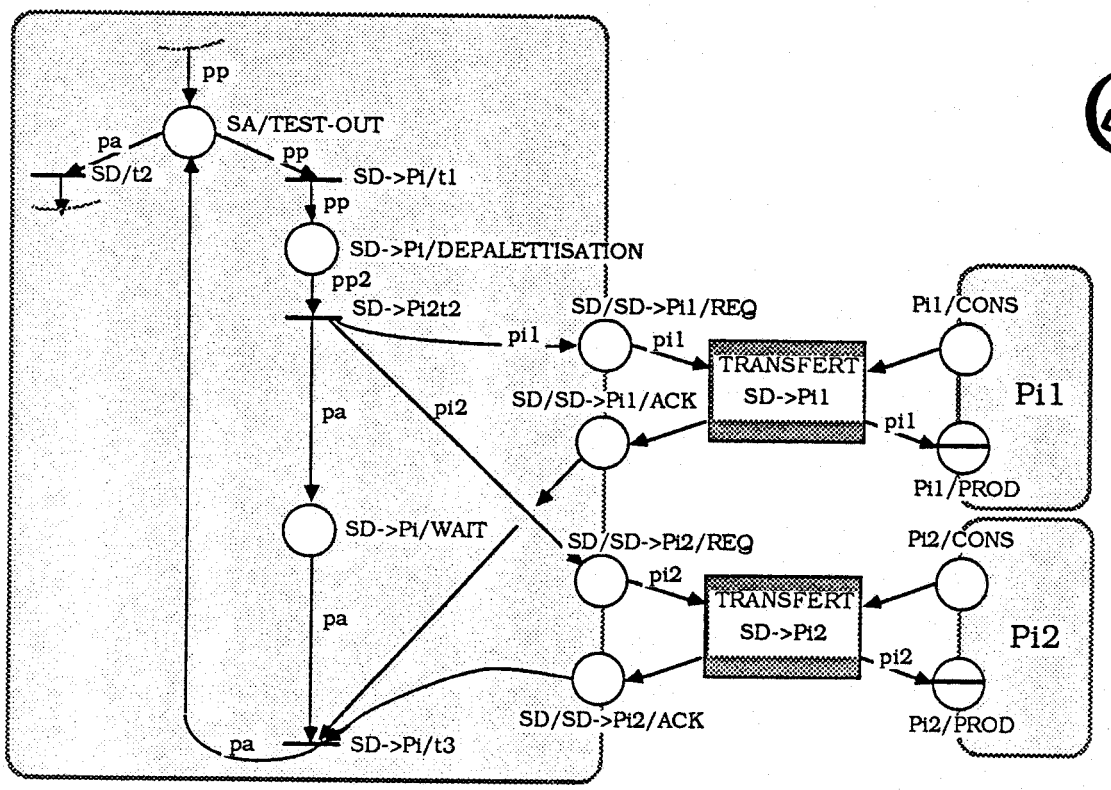


FIGURE 3.42

II.3.6 - Les positionnements

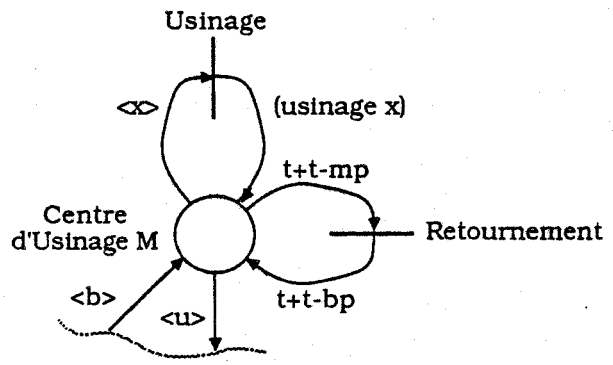
Nous appelons "positionnements", des opérations qui sont représentées sur le prégraphe par une transition implicite dont l'information associée est positionnelle (cf. Chapitre II, § II.3). Considérons, à titre d'exemple, une pièce brute dont la gamme opératoire est la suivante :

- usinage,
- retournement,
- usinage.

Appelons :

t^-t^-, la pièce brute,
t^+t^-mp, la pièce avant le retournement et après le premier usinage,
t^+t^-bp, la pièce après le retournement,
t^+t^+, la pièce totalement usinée.

Ces différentes opérations sont modélisées sur le prégraphe par la Figure 3.43 :



avec : dom (<x>) = { t^-t^- , t^+t^-bp }

et :
```
(de usinage (x)
    (cond ((equal x 't-t-) 't+t-mp)
          ((equal x 't+t-bp) 't+t+)))
```

FIGURE 3.43

L'opération de retournement est une opération de positionnement.

Dans le cadre de notre étude, nous avons été amenés à distinguer deux types de positionnement, selon que l'objet libère ou non la zone opératoire.

Nous appellerons ces deux types de positionnement :
- "positionnements externes" (Fig. 3.44), et
- "positionnements internes" (Fig. 3.45).

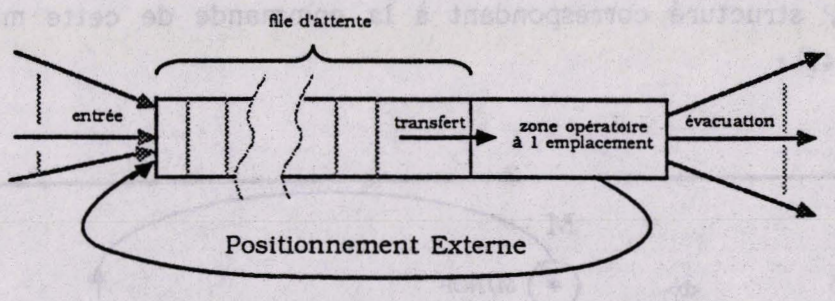


FIGURE 3.44

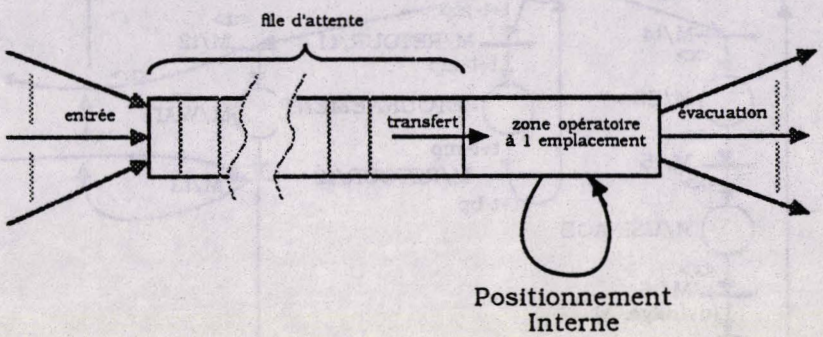


FIGURE 3.45

a) Les positionnements internes

De la même façon que pour les opérations de palettisation, nous créons une structure "répétitive" au niveau de la place "TEST-OUT" du lieu considéré.

Reprenons l'exemple du début du Paragraphe II.5, et supposons que le centre d'usinage soit une machine avec un tampon d'entrée/sortie. Le positionnement interne correspond, par exemple, à une rotation de la platine d'usinage supportant la pièce (Fig. 3.46).

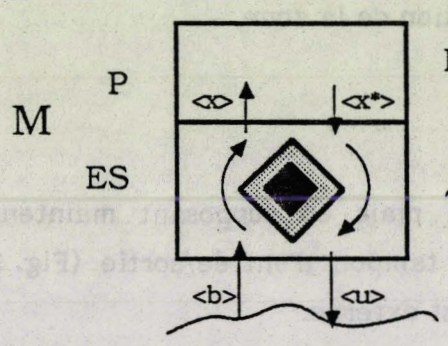


FIGURE 3.46

Le RdP structuré correspondant à la commande de cette machine est alors (Fig. 3.47) :

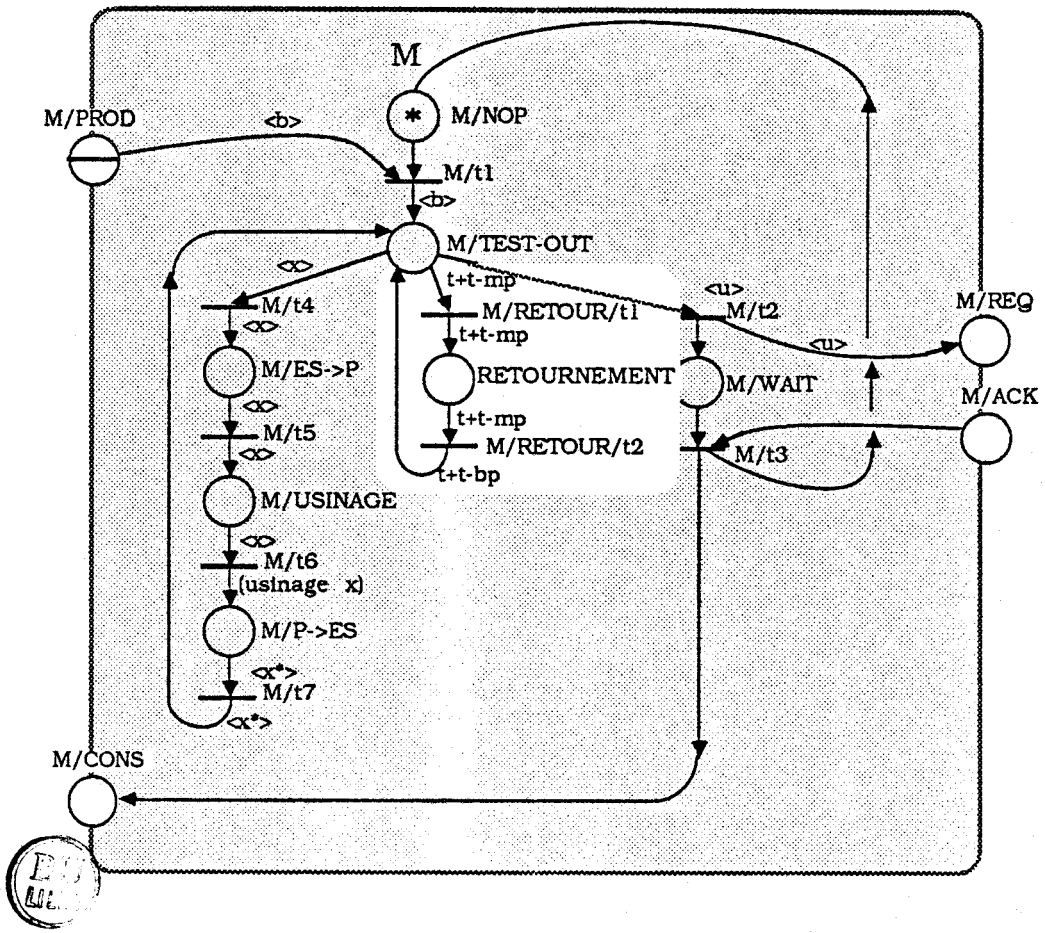


FIGURE 3.47

b) Les positionnements externes

Pour ces positionnements, la pièce libère le lieu considéré. Il est alors nécessaire de créer un processus extérieur qui gère :

- la prise de la pièce et la libération de la zone,
- le positionnement,
- le dépôt de la pièce.

Reprenons l'exemple précédent, mais en supposant maintenant que le centre d'usinage ne possède pas de tampon d'entrée/sortie (Fig. 3.48). Dans ce cas, le positionnement de la pièce est externe.

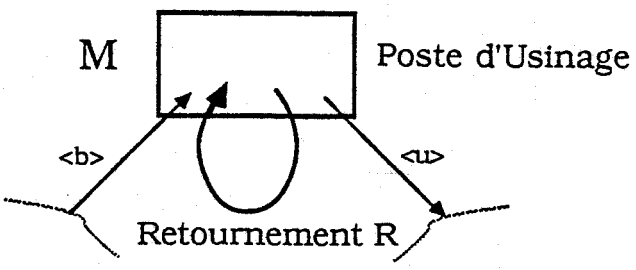


FIGURE 3.48

Le RdP structuré correspondant est alors :

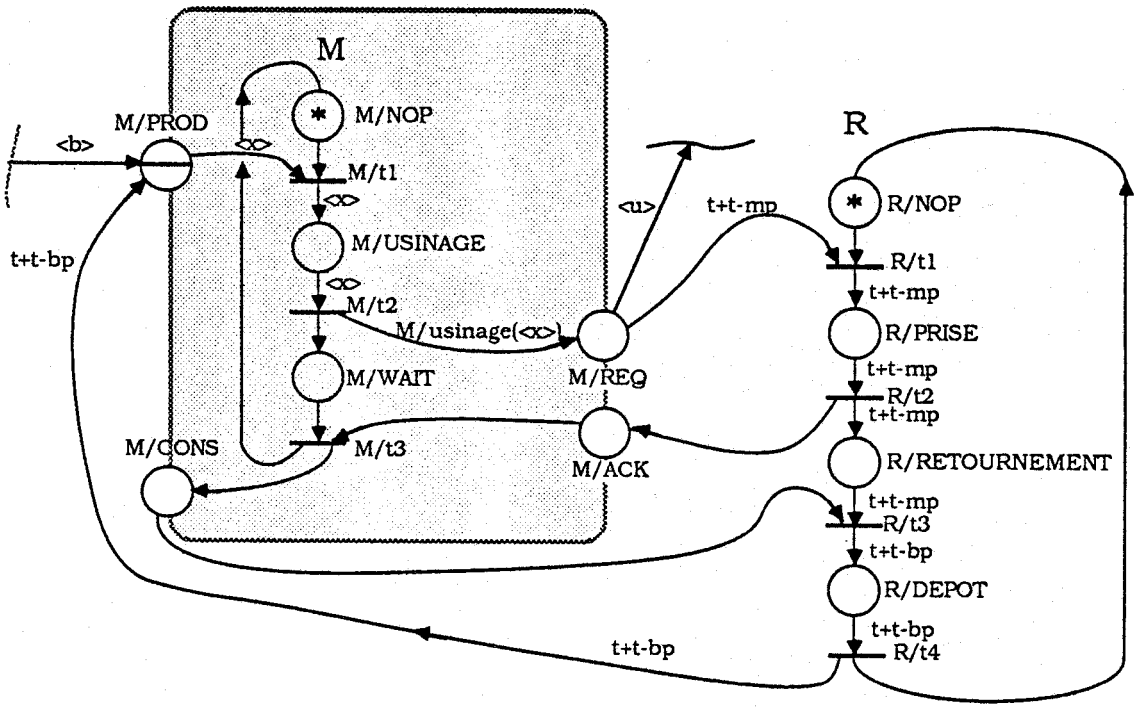


FIGURE 3.49

II.3.7 - Conclusion

Nous avons présenté dans ce paragraphe, les principales primitives utilisées pour le développement et la structuration du prégraphe. Notre souci a été d'offrir la plus grande flexibilité possible pour la création du graphe de commande.

Dans ce sens, nous avons choisi une méthode de conception qui permet de générer un graphe de commande sur lequel le parallélisme et la concurrence du système sont mis en évidence. En effet, tel un jeu de construction, nous assemblons différents modules pré-établis (machines, opérations d'assemblage, de positionnements, ...) après une analyse des fonctionnalités du prégraphe. Cette grande modularité de la partie commande est d'autant plus intéressante qu'elle accélère les phases de mise au point et d'implantation mais aussi et surtout simplifie les procédures de prise en compte d'éventuelles modifications en cours d'exploitation.

De plus, l'automatisation de la démarche de structuration apporte un gain de temps non négligeable dans la phase de conception et une meilleure fiabilité due à la **nécessaire** cohérence logique du modèle produit avec le prégraphe.

III - REPARTITION DES SYSTEMES DE COMMANDE

III.1 - Introduction

Le but de cette étape est de faire apparaître sur le graphe développé, les contraintes relevant de l'architecture matérielle de l'atelier et plus particulièrement des systèmes de manutention et d'assemblage/désassemblage.

En effet, de par sa conception, le réseau de Petri développé issu de l'étape de structuration met en évidence le parallélisme maximal du système et ne permet pas de prendre en compte directement le fait que, par exemple, un robot effectue deux tâches de transfert de manière exclusive ou, au contraire, synchronisée.

Dans ce sens, nous proposerons des solutions permettant de prendre en compte de telles problématiques.

III.2 - Les transferts synchronisés

Il s'agit ici de synchroniser deux ou plusieurs opérations de transfert. Considérons, à titre d'exemple, les deux opérations de transfert schématisés sur le prégraphe par la Figure 3.50.

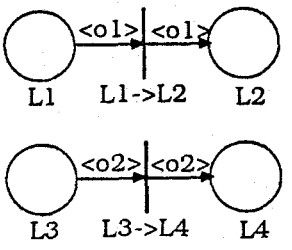


FIGURE 3.50

Le développement structuré de ces deux opérations de transfert est donné Figure 3.51 :

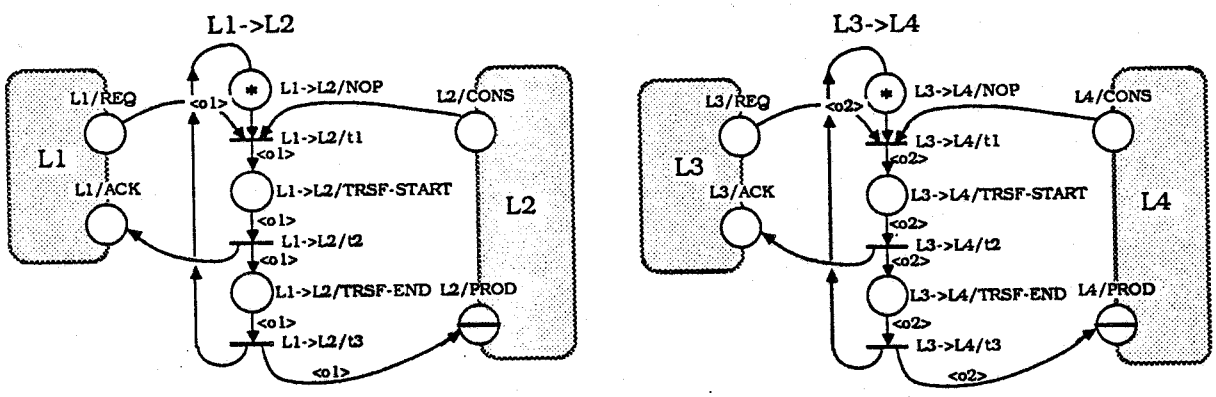


FIGURE 3.51

Tels qu'ils sont représentés sur la Figure 3.51, les deux processus de transfert commandent chacun leur propre système de manutention (des robots, par exemple). Les deux transferts sont alors indépendants.

Supposons maintenant que le concepteur désire étudier l'influence sur les performances du système de la synchronisation des deux transferts dans le cas, par exemple, d'un robot de manutention représenté Figure 3.52 :

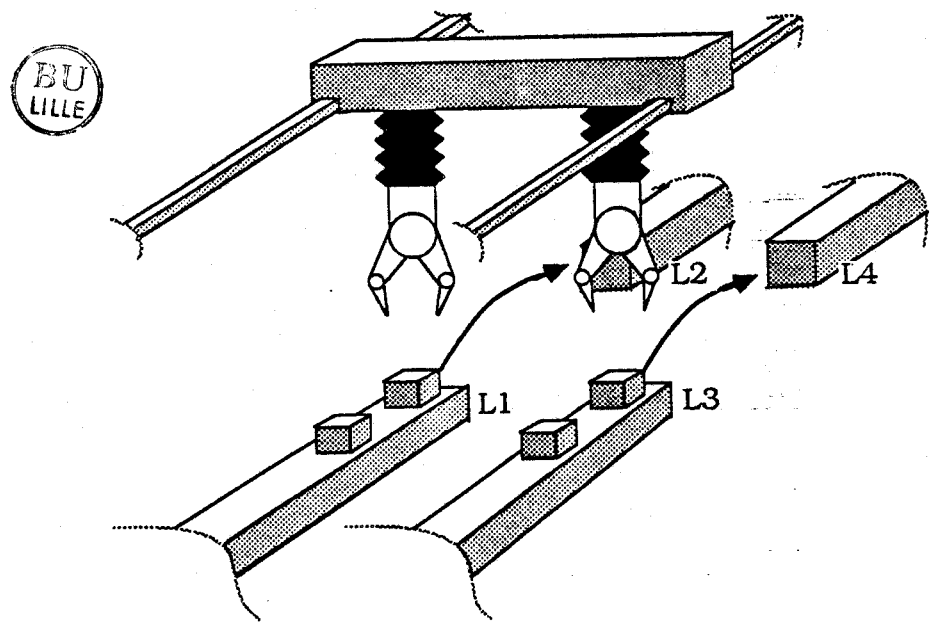


FIGURE 3.52

Il est alors nécessaire d'introduire sur le graphe de la Figure 3.51, un processus permettant la synchronisation des deux transferts. Nous aboutissons au schéma de la Figure 3.53 :

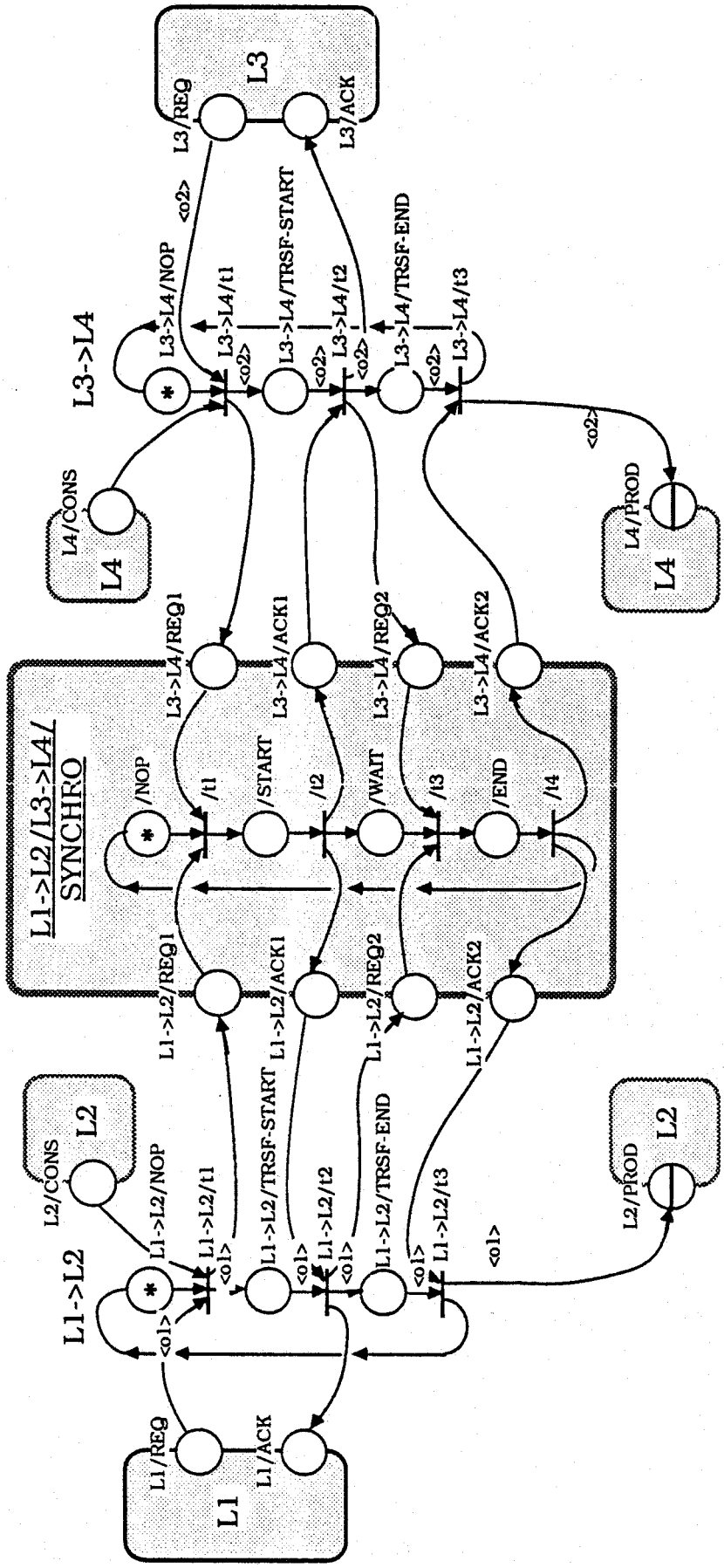

L3
L3/REQ
L3/ACK
L3->L4
L3->L4/NOP
L3->L4/t1
L3->L4/TRSF-START
L3->L4/t2
L3->L4/TRSF-END
L3->L4/t3
<o2>
L4/CONS
L4
L4/PROD
L4
L3->L4/REQ1
L3->L4/ACK1
L3->L4/REQ2
L3->L4/ACK2
L1->L2/SYNCHRO
/NOP
/t1
/START
/t2
/WAIT
/t3
/END
/t4
L1->L2/REQ1
L1->L2/ACK1
L1->L2/REQ2
L1->L2/ACK2
L2/CONS
L2
L1->L2
L1->L2/NOP
L1->L2/t1
L1->L2/TRSF-START
L1->L2/t2
L1->L2/TRSF-END
L1->L2/t3
<o1>
L2/PROD
L2
L1/REQ
L1/ACK
L1
BU LILLE

FIGURE 3.53

La création d'un processus de synchronisation alourdit la représentation mais permet néanmoins :

i) d'étudier la synchronisation de plus de deux processus en ajoutant les liaisons avec le processus de synchronisation,

ii) d'étudier les conséquences d'une synchronisation qui n'a pas été prévue au départ de la conception. En effet, si tel en avait été le cas, la structure du prégraphe aurait été :

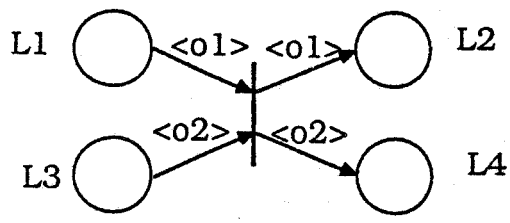


FIGURE 3.54

iii) de revenir à des transferts non synchronisés si les performances obtenues après simulation ne correspondent pas aux résultats escomptés. Ce retour est réalisé par suppression des liaisons et du processus de synchronisation,

iv) d'envisager différentes possibilités de synchronisation à partir d'un même schéma prégraphe.

Exemple : Considérons le prégraphe de la Figure 3.55 qui modélise un système de convoyage circulaire au sein d'une cellule.

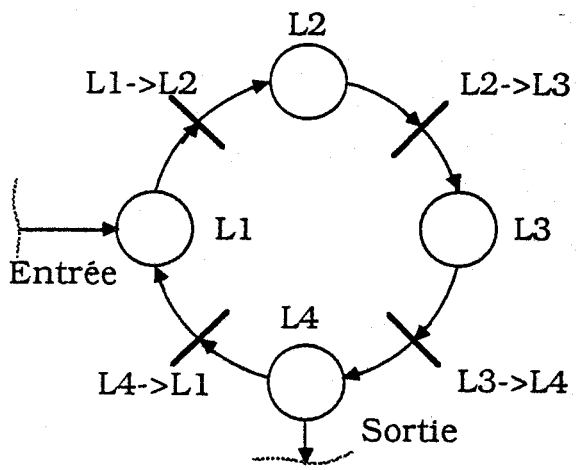


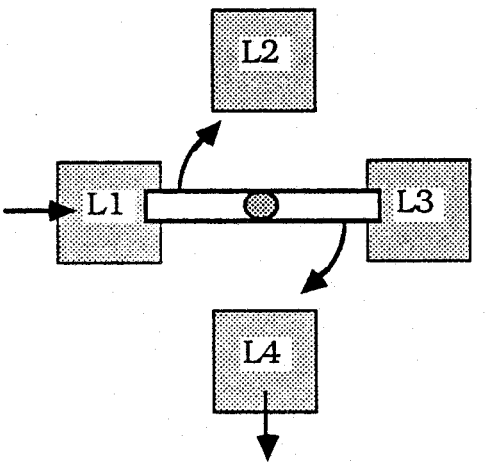
FIGURE 3.55

Ce système de convoyage comporte quatre zones de stockage intermédiaires.

Après la structuration de ce prégraphe, il est possible d'étudier le système en supposant, par exemple, que :

a) tous les transferts sont indépendants ; c'est le cas d'un convoyeur à chaîne débrayable,

b) les transferts sont synchronisés deux par deux ; c'est le cas d'un bras manipulateur à deux positions, schématisé par la Figure 3.56 :



Vue de dessus

FIGURE 3.56

c) tous les transferts sont synchronisés ; il s'agit alors d'un manipulateur à quatre positions (Fig. 3.57) ou d'un carrousel (Fig. 3.58) :

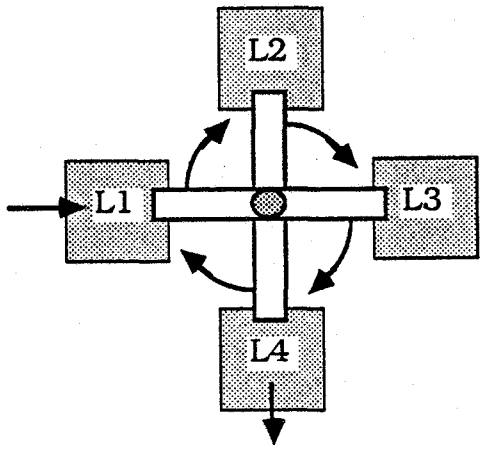


FIGURE 3.57

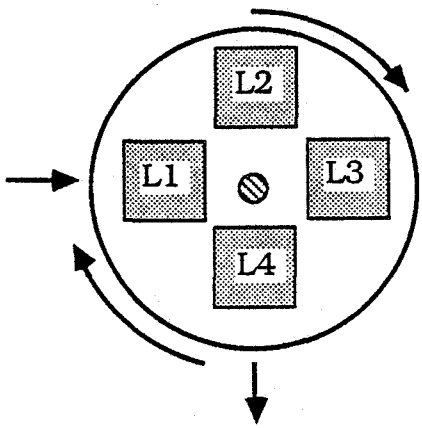


FIGURE 3.58

Le graphe développé fait alors apparaître les quatre processus de transfert synchronisés par un processus de synchronisation analogue à celui de la Figure 3.53.

III.3 - Les robots

III.3.1 - Introduction

Dans ce paragraphe, nous nous intéressons aux systèmes qui réalisent deux ou plusieurs opérations de manière exclusive telles que transferts, (dés)-assemblages, positionnements, etc...

Par abus de langage, nous regrouperons de tels systèmes sous le nom générique de "robot", qu'il s'agisse ou non de robots, au sens propre du terme, ou de chariots filoguidés par exemple.

Nous aborderons donc successivement la traduction sur le RdPSC de la partie commande des contraintes liées aux robots de transferts, d'assemblage/ désassemblage, de positionnement.

III.3.2 - Les robots de transfert

Afin de bien préciser la méthode employée, reprenons l'exemple des deux transferts de pièces décrit sous forme de prégraphe par la Figure 3.50 et sous forme développée par la Figure 3.51.

Supposons maintenant que les deux transferts sont **indépendants** et assurés par le **même** robot R. Ces deux opérations de transfert sont donc exclusives et, de fait, les graphes de commande correspondants ne peuvent évoluer simultanément.

Afin de traduire cette contrainte sur le RdPSC, nous introduisons une liaison d'exclusion mutuelle matérialisée par une place ressource notée R qui représente l'état (libre ou occupé) du robot.

Nous aboutissons alors au graphe de la Figure 3.59 :

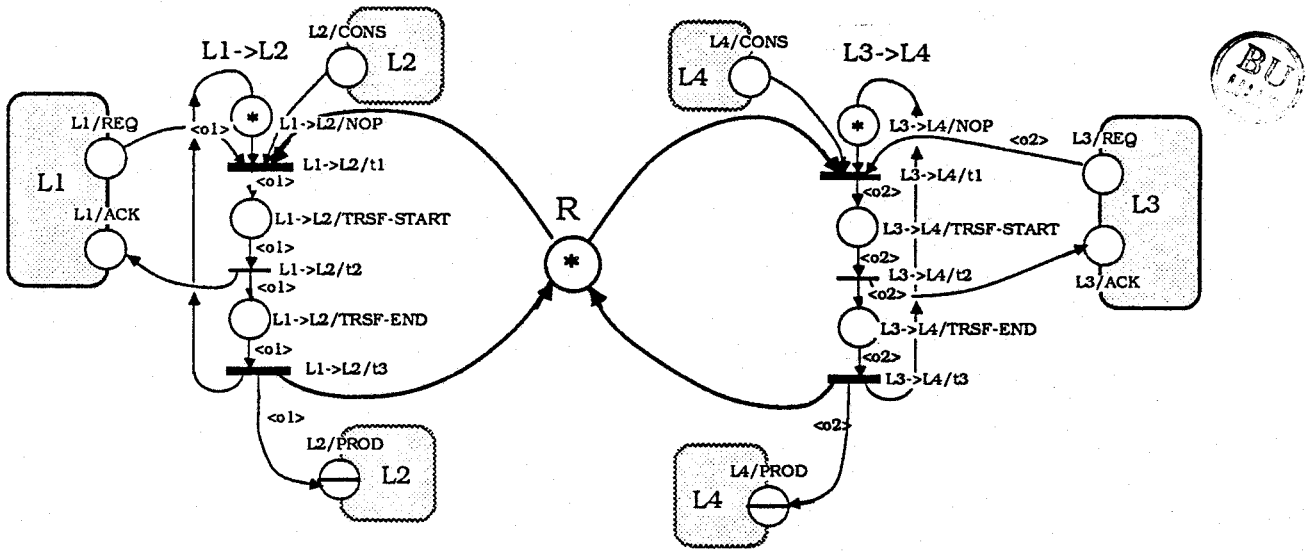


FIGURE 3.59

Ainsi, toutes les contraintes relevant du partage d'un robot entre plusieurs tâches de transfert exclusives seront représentées par une ressource critique et des liaisons d'exclusion mutuelle vers les processus de commande correspondants. Le nombre de marques initiales dans la place R permet de modéliser le nombre de ressources identiques et disponibles pour effectuer les transferts.

III.3.3 - Les robots d'assemblage et de désassemblage, de positionnement

Le principe de représentation des contraintes de partage de ces robots est identique au précédent ; nous représentons :

- l'état du robot par une ressource,
- les contraintes de partage par des liaisons d'exclusion mutuelle sur les opérations ou séquences d'opérations exclusives concernées.

Exemple : Considérons les deux opérations d'assemblage représentées sur un prégraphe par le schéma de la Figure 3.60.

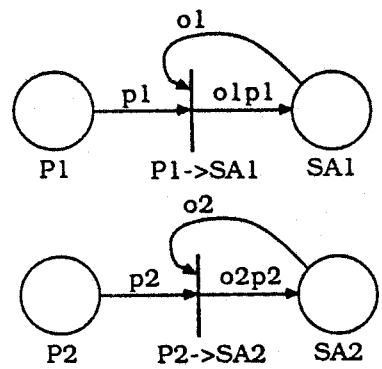


FIGURE 3.60

Supposons que les deux opérations de palettisation sont assurées par le **même** robot d'assemblage "RA". le graphe développé devient alors :

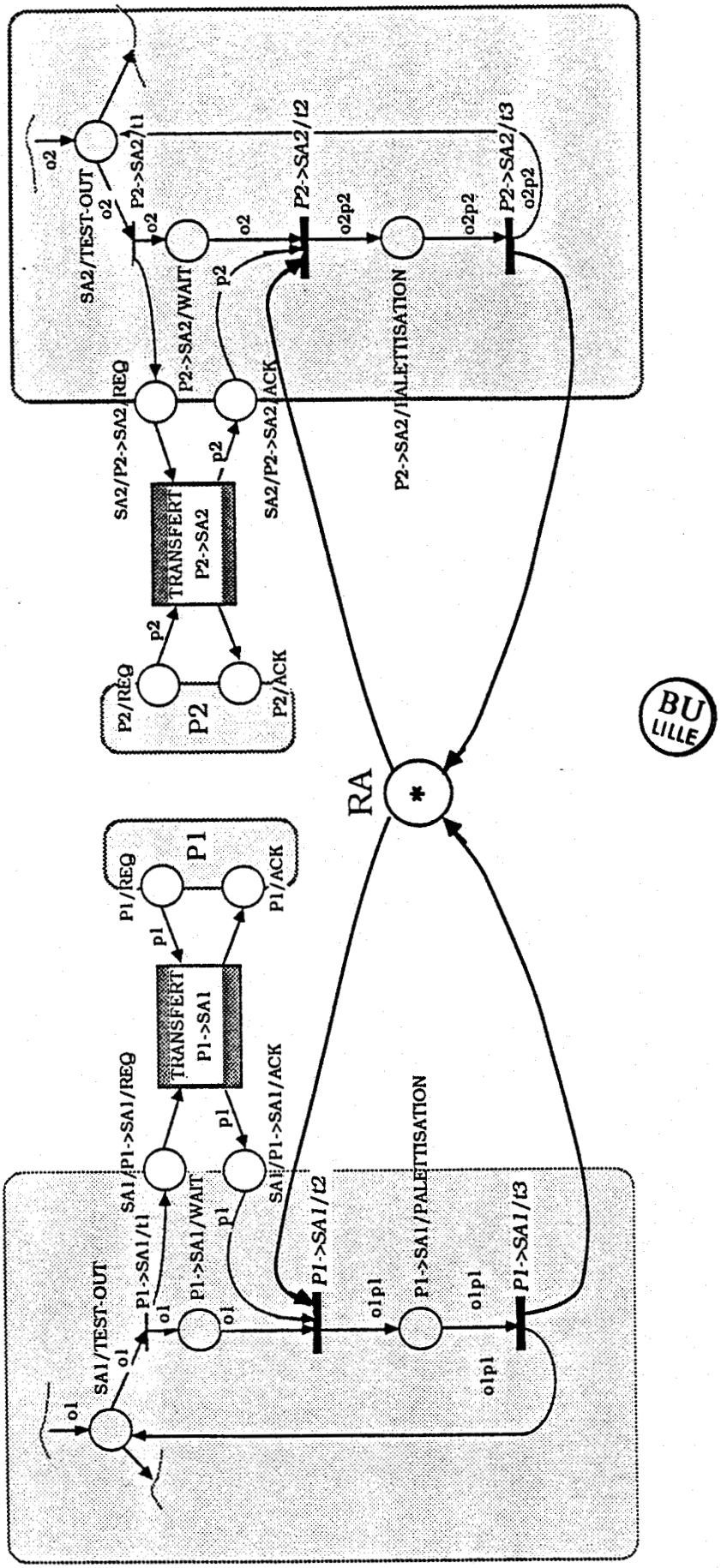

SA2/TEST-OUT
o2
P2->SA2/t1
P2->SA2/t2
P2->SA2/t3
o2p2
P2->SA2/WAIT
p2
REQ
ACK
SA2/P2->SA2
TRANSFERT
P2->SA2
P2->SA2/PALETTISATION
P2/REQ
P2
P2/ACK
RA
*
P1/REQ
P1
P1/ACK
p1
TRANSFERT
P1->SA1
SA1/P1->SA1/REQ
SA1/P1->SA1/ACK
SA1/TEST-OUT
o1
P1->SA1/t1
P1->SA1/t2
P1->SA1/t3
o1p1
P1->SA1/WAIT
P1->SA1/PALETTISATION
BU
LILLE

FIGURE 3.61

III.3.4 - Conclusion

Dans ce paragraphe, nous avons présenté la prise en compte des contraintes liées à l'utilisation de ressources critiques sur des opérations de même type (transferts, assemblages, etc...). Il est bien évident que nous prenons également en charge le cas des "robots polyvalents", c'est-à-dire, capables de réaliser, par exemple, des transferts et des assemblages après éventuellement un changement d'outil. Les liaisons d'exclusion mutuelle seront alors placées sur des opérations de type différent.

III.4 - Conclusion

Nous avons proposé un ensemble de solutions permettant d'assurer la prise en compte des différentes possibilités de répartition des systèmes de commande.

Sur le plan de la réalisation, nous avons écrit un module spécifique qui permet au concepteur de préciser de manière interactive la répartition des "robots".

Dans une phase ultérieure, nous pensons prendre également en compte certaines spécifications relevant de cet aspect, qui pourraient être directement issues du cahier des charges.

IV - INTEGRATION DU NIVEAU HIERARCHIQUE ET DU PROCEDE

IV.1 - Introduction

Nous nous intéressons dans cette partie, aux étapes de création des modèles du niveau hiérarchique et du procédé qui, associés au modèle de la partie commande, permettront d'envisager une étude complète du système par simulation.

Comme nous l'avons précisé dans le Chapitre II, ces deux dernières phases de modélisation constituent une part importante des travaux actuellement en cours au L. A. I. I.. De ce fait, nous nous limiterons à une présentation des étapes prises en charge à ce jour par notre logiciel, c'est-à-dire les étapes 6, 7, 9, 10 et 11 (cf Fig. 3.1).

IV.2 - Intégration du niveau hiérarchique

IV.2.1 - Introduction

Dans ce paragraphe, nous abordons les problèmes liés à la détection des conflits structurels et à l'interfaçage du niveau hiérarchique avec la partie commande. Si le premier module est indépendant du "type" de règle du niveau hiérarchique (règles de requête ou règles de surveillance), il n'en est pas de même du deuxième. Il convient en effet de rappeler que nous devrons prendre ultérieurement en considération ces deux types de règles, ce qui impliquera des modifications du module d'interfaçage PC-NH.

IV.2.2 - Détection des conflits

a) Introduction

Il s'agit ici de détecter les conflits inhérents à la **structure** du graphe : ce sont les conflits structurels /BRA 83/. Contrairement aux conflits effectifs qui font appel à la notion de marquage et qui, de ce fait, ne peuvent être détectés, dans le cas de RdP non autonomes, que par simulation, les conflits structurels peuvent être détectés dans leur intégralité par une scrutation systématique des transitions du graphe.

Dans ce sens, nous avons écrit un module qui fournit automatiquement la liste de tous les conflits structurels, ainsi que les marquages pour lesquels ces conflits deviennent effectifs. Cependant, il est possible que certains conflits structurels que nous détectons ne deviennent jamais effectifs lors de l'évolution du graphe. En effet, le marquage donnant lieu à ces conflits effectifs peut correspondre à un état que le système n'atteindra jamais au cours de son évolution, compte tenu par exemple, de l'ordonnancement des pièces ou même des temps opératoires. La liste des conflits structurels s'avère néanmoins intéressante car elle permet au concepteur, avant même la première simulation, d'écrire les règles du niveau hiérarchique permettant de les résoudre. Il pourra alors consacrer les différentes simulations à :

- la mise en évidence de conflits n'incombant pas à la structure du graphe que nous appelons "'conflits dynamiques",
- l'étude de cohérence des règles introduites,
- la recherche d'une politique d'ordonnancement évitant les conflits dynamiques,
- etc, ...

La détection des conflits structurels s'effectue par scrutation systématique du graphe de commande. Etant donné que ce graphe est issu directement des étapes de structuration et de répartition des systèmes de commande, nous connaissons toutes les structures apparaissant dans le graphe et donc les sous-réseaux dans lesquels les conflits apparaissent, à savoir :

- les partages de ressources,
- les répétitives et alternatives multiples,
- les sorties de zones.

Nous avons donc limité la scrutation du graphe à ces trois structures, ce qui permet d'accélérer notablement la recherche des conflits.

Pour chacune de ces structures, nous donnons ci-après un exemple du type de conflits rencontrés.

b) Le partage de ressources

Reprenons l'exemple du paragraphe III.3.2 qui concerne un robot assurant deux transferts de manière exclusive et dont le graphe de commande est donné Figure 3.62 :

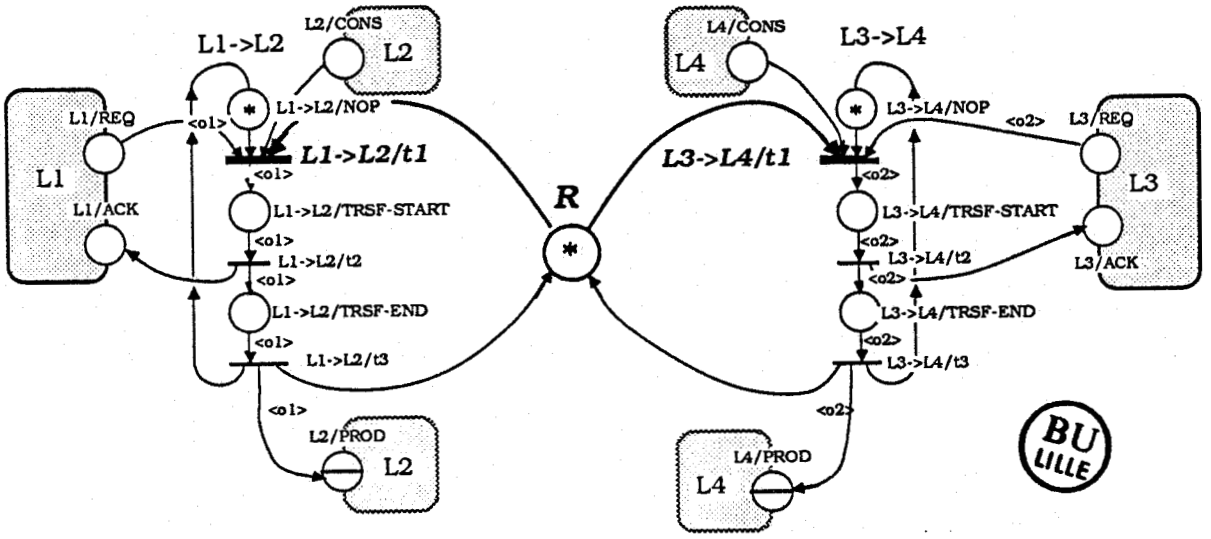


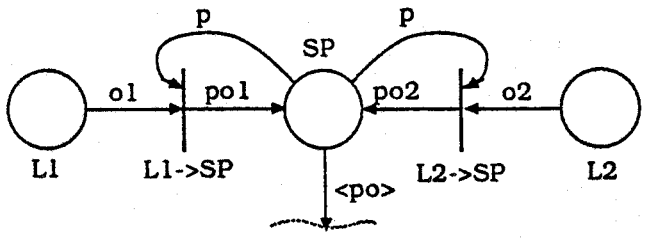
FIGURE 3.62

Le conflit se détecte trivialement en prenant la liste des transitions en aval de la ressource. Dans notre exemple, il s'agit des transitions L1 ⟶ L2/t1 et L3 ⟶ L4/t1.

c) Les répétitives et alternatives multiples

Considérons une station de palettisation SP où des objets o1 et o2 sont assemblés sur une palette notée p pour donner po1 et po2. Supposons de plus, que o1 et o2 proviennent de deux lieux distincts L1 et L2.

La représentation au niveau d'un prégraphe est donnée Figure 3.63 :



avec : dom (<po>) = {po1 , po2 }

FIGURE 3.63

La représentation développée sous forme de RdPSC est donnée Figure 3.64 :

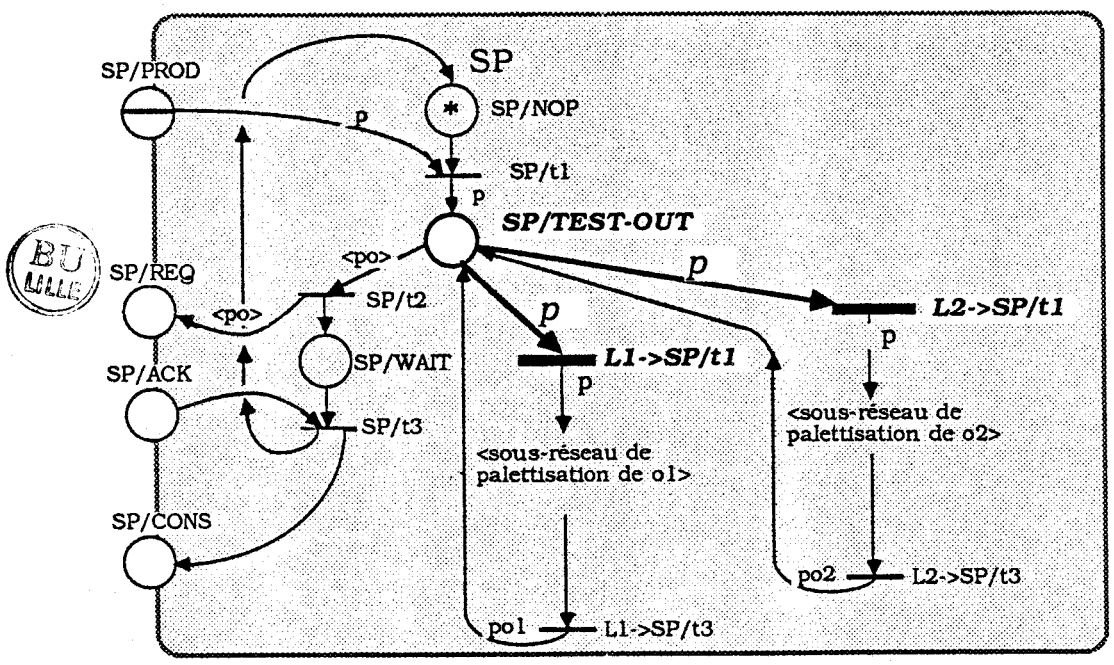


FIGURE 3.64

Sur un tel graphe, nous avons introduit une structure répétitive multiple au niveau de la place SP/TEST-OUT. Du fait de la structure utilisée, les transitions t2, t4 et t5 sont en conflit structurel. De plus, lorsque SP/TEST-OUT est marquée par une marque de couleur p, les transitions t4 et t5 sont en conflit effectif. ceci correspond au fait que, lorqu'une palette vide se trouve sur la zone opératoire de SP, il est possible d'effectuer une palettisation d'un objet o1 **ou** o2, le choix dépend alors de la stratégie de pilotage envisagée.

De manière générale, les structures "répétitives multiples" et "alternatives multiples" génèrent des conflits effectifs au niveau de la place de choix lorsque celui-ci n'est pas déterministe.

d) Les sorties de zones

Considérons, à titre d'exemple, une zone de stockage intermédiaire "ZS" à la sortie de laquelle une pièce peut être aiguillée vers un lieu L1 ou L2. Les modélisations sous forme de prégraphe et de RdPSC sont données respectivement Figures 3.65 et 3.66 :

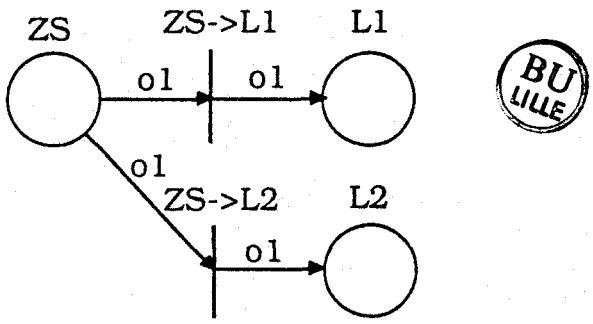


FIGURE 3.65

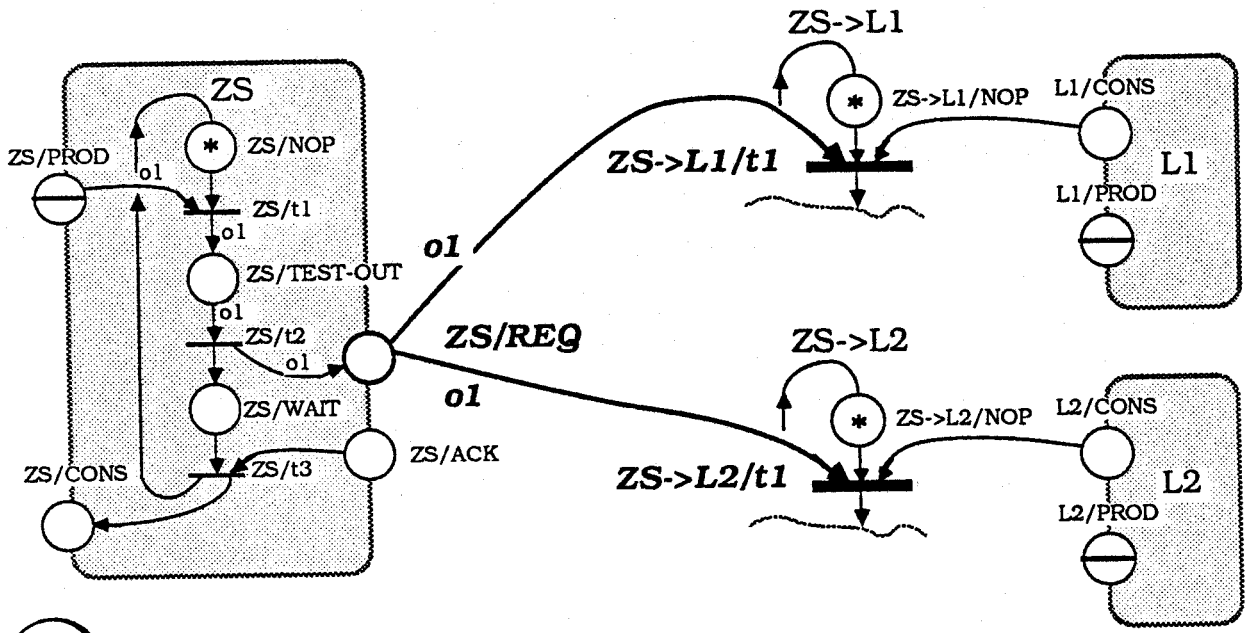


FIGURE 3.66

Sur un tel exemple, nous voyons qu'il y a conflit structurel entre les transitions ZS ⟶ L1/t1 et ZS ⟶ L2/t1 qui devient effectif lorsque la place ZS/REQ est marquée par une marque de couleur o1.

Ce type de conflit se produit chaque fois qu'il y a indéterminisme de choix à la sortie d'une zone (machines, station d'assemblage, etc...) comme par exemple, pour des systèmes de convoyage flexibles dans lesquels les objets identiques peuvent emprunter des itinéraires **différents** pour aller d'un point à un autre de l'installation.

e) Conclusion

Compte tenu de la méthode de développement utilisée, les trois types de configurations que nous avons présentés sont les seuls qui génèrent des conflits. La détection s'effectue donc simplement par :

i) une étude des transitions situées en aval des ressources critiques,

ii) une étude de l'intersection des ensembles de couleurs étiquetant les arcs reliant les places de **requête** (<nom-processus>/REQ) et de **choix** (<nom-processus>/TEST-OUT) avec les transitions situées en aval.

Il en résulte une liste exhaustive des conflits effectifs possibles, relatifs à la structure du graphe.

IV.2.3 - Interfaçage avec la partie commande

Le but de cette étape (étape 7 du projet C.A.S.P.A.I.M.) est de créer l'interface entre le niveau hiérarchique et la partie commande.

Cette interface est actuellement réalisée par un ensemble de couples (arc-adaptatif ; place d'interface) permettant de bloquer ou non l'évolution de certains sous-ensembles du graphe développé en fonction du déclenchement des règles du niveau hiérarchique (cf Chapitre I, § II).

Afin de résoudre les conflits possibles et de pouvoir jouer sur la flexibilité des installations étudiées, nous avons choisi de créer systématiquement un couple (arc-adaptatif ; place d'interface)

i) à l'entrée de chaque processus (Figure 3.67),

ii) au niveau de chaque opération de choix (Figure 3.68).

Les places d'interface sont notées avec le suffixe "/CHOIX-NH".

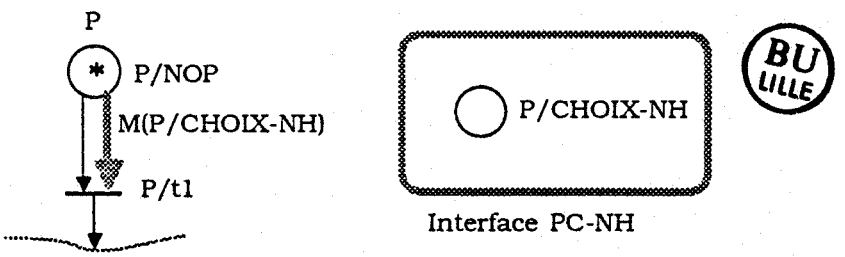


FIGURE 3.67

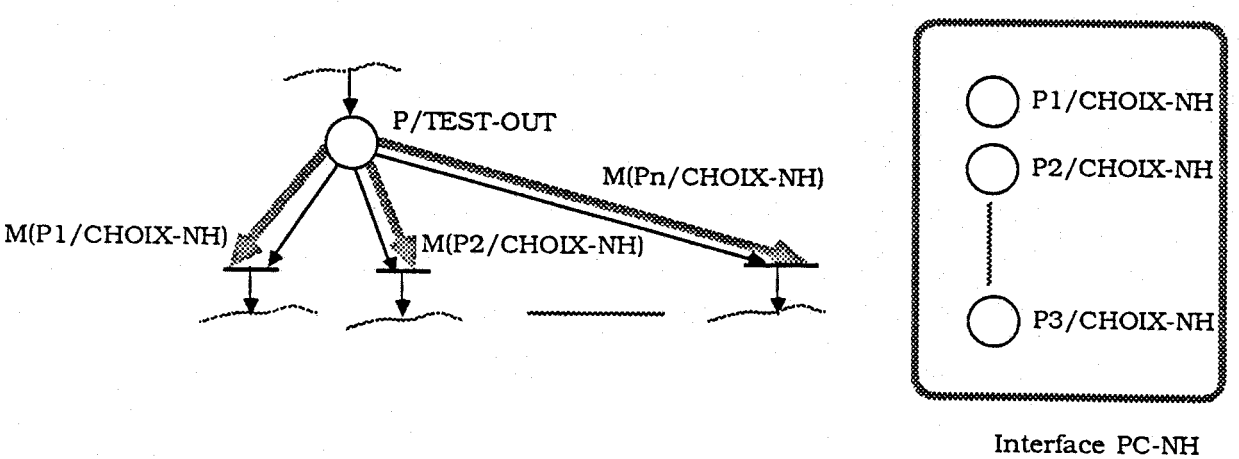


FIGURE 3.68

Ce type d'interfaçage n'est valable que pour des règles de surveillance. Ce choix qui n'est pas optimal au sens des performances du simulateur, présente l'avantage de ne pas augmenter de façon considérable la taille du graphe et donc l'espace mémoire occupé en machine.

IV.3 - Intégration du procédé

IV.3.1 - Introduction

Cette phase de modélisation est composée de trois étapes indépendantes :
- le choix des entrées,
- le dimensionnement,
- la temporisation.

La mise en œuvre de ces trois étapes est assurée par trois modules distincts qui permettent au concepteur de préciser interactivement les données qui seront utilisées par le simulateur.

IV.3.2 - Choix des entrées

a) Introduction

L'ordre et la fréquence d'entrée des pièces dans le système est un facteur déterminant de son évolution. L'apparition d'un blocage, par exemple, résulte généralement de cette séquence d'entrée et son origine relève plutôt de considérations d'ordonnancement que de la notion de vivacité en terme de validation des RdP. En effet, les méthodes de validation actuelles s'appliquent aux réseaux de Petri **autonomes** et prennent surtout en considération la structure du graphe. Ici, en plus de la structure, interviennent la séquence et la fréquence d'entrée. En effet, aucun outil méthodologique ne permet aujourd'hui de valider formellement le bon fonctionnement du modèle pour toutes les séquences possibles d'entrées.

C'est pourquoi, l'émulation des entrées est un point particulièrement important dans l'évaluation d'un système de production.

b) Spécification de la fonction d'entrée

La spécification de la fonction d'entrée dépend de la **séquence** des pièces pénétrant dans le système. Nous distinguons trois types de séquences : la séquence interactive, la séquence prédéfinie et la séquence aléatoire.

*** La séquence interactive :**

L'opérateur décide du type de la pièce, à l'instant où une pièce pénétre dans le système. Une telle fonction peut être définie en Le_Lisp de la manière suivante :

```
(de  entrée / génération-pièces ( )
     (prog  (rep)
            (print  "Veuillez entrer la pièce suivante :")
            (setq  rep  (read))
            (ajouter  1  rep  ' place-entrée)))
```

où "place-entrée" est la place du RdP modélisant l'entrée du système.

*** La séquence prédéfinie :**

L'opérateur construit la liste des pièces fournissant la séquence d'entrée.

```
(setq  entrée / réserve  ' (t⁻ t⁻f⁻ f⁻  ...  t⁻t⁻ t⁻t⁻))
(de  entrée / génération-pièces ( )
     (ajouter  1  (nextl  entrée / réserve)  ' place-entrée))
```

*** La séquence aléatoire :**

Le type de la pièce est défini aléatoirement parmi les différents types de pièces possibles. Si le domaine d'entrée des pièces est (t^-, t^-t^-, f^-, f^-t^-, t^-f^-), l'utilisateur pourra définir la fonction d'entrée de la manière suivante :

```
(de  entrée / génération-pièces ( )
     (ajouter  1   (nth  (random  0 5)  ' (t⁻, t⁻t⁻, f⁻, f⁻t⁻, t⁻f⁻))
                   ' place-entrée))
```

c) Intégration de la fonction d'entrée au modèle

La fonction d'entrée peut être associée à une règle, à un évènement, à l'étiquetage d'un arc, etc.... Le choix du point d'intégration dépend essentiellement de la **fréquence** d'entrée des pièces. Nous en distinguons ici trois types : l'itération interactive, l'itération au plus tôt et l'itération évènementielle.

*** L'itération interactive :**

L'opérateur gère l'entrée des pièces dans le système par le biais de l'interruption "intervention opérateur".

INTERVENTION OPERATEUR > **(entrée)**
Veuillez entrer la pièce suivante : **t⁻**
INTERVENTION OPERATEUR > **suite**

*** L'itération au plus tôt :**

Elle modélise un flux continu de pièce en entrée. Pour cela, nous créons un processus dont l'arc de sortie est étiqueté par la fonction de génération de pièces :

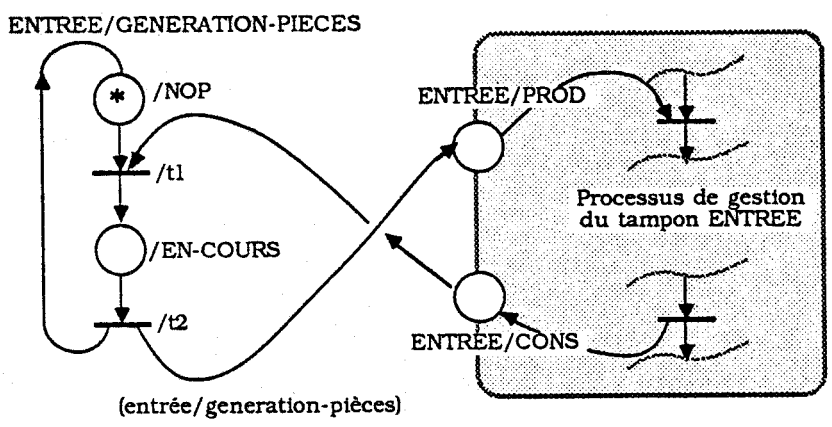


FIGURE 3.69

*** L'itération évènementielle :**

L'occurence d'entrée des pièces est liée au cycle "par évènement" du simulateur. Ceci peut être réalisé pratiquement de plusieurs façons : les évènements d'entrée de pièces sont prévus initialement dans l'échéancier ou plus simplement générés par temporisation de la structure émulant l'entrée (temporisation de la place Entrée/Prod).

d) Conclusion

Le choix de la séquence d'entrée (définition de la fonction d'entrée) et de sa fréquence (mode d'intégration au modèle) permettent de définir la politique d'entrée adéquate aux résultats de simulation désirés.

Seules les trois possibilités de séquence d'entrée et les trois possibilités de fréquence sont offertes à l'utilisateur dans le logiciel réalisé. L'utilisateur peut néanmoins définir facilement d'autres politiques d'entrée en redéfinissant la séquence et la fréquence selon la génération de pièces qu'il désire pour son étude.

IV.3.3 - Le dimensionnement

Cette étape consiste à préciser le marquage initial du réseau de Petri modélisant la partie commande. Le module chargé de la mise en œuvre de cette étape initialise automatiquement tous les processus en positionnant une marque dans les places Pi/NOP.

De plus, ce module permet de saisir interactivement la taille maximale de toutes les zones tampon du système, ainsi que l'état initial de ces zones, c'est-à-dire le nombre d'emplacements libres, la nature et le nombre d'objets initialement présents. Ceci permet à l'utilisateur de démarrer la simulation du système à partir d'une configuration donnée qui peut être, par exemple, le résultat d'une simulation précédente.

La recherche de la taille optimale des zones de stockage dans les systèmes flexibles de production est un problème qui n'est résolu actuellement que par des méthodes d'essais/erreurs. Dans notre cas, nous avons conçu notre module de dimensionnement de telle sorte qu'il soit utilisable directement à partir du simulateur.

IV.3.4 - La temporisation

Comme nous l'avons précisé dans le Chapitre I, § III.2, nous avons choisi d'utiliser le modèle Réseau de Petri **Temporisé** afin de prendre en compte la dynamique imposée par le procédé.

Dans ce sens, nous avons écrit un module permettant, pour chaque place correspondant à une action effective (usinage, transfert, etc...) de saisir une valeur de temporisation. Cette valeur peut être identique pour toutes les marques transitant par la place mais peut également être spécifique à une ou plusieurs couleurs par exemple, pour prendre en compte des temps d'usinage différents pour des pièces différentes.

De la même façon que pour le module de dimensionnement, le module de temporisation peut être exécuté à partir du simulateur afin de modifier, en cours de simulation, les temporisations d'une ou plusieurs places.

IV.4 - Conclusion

Les différents modules que nous avons présentés dans ce paragraphe constituent une base essentielle à l'étude par simulation des systèmes flexibles de production. Ils permettent en effet de construire les modèles qui sont associés au modèle de la partie commande et de définir les éléments indispensables à une étude du comportement dynamique du système tels que l'émulation des entrées de pièces.

CONCLUSION

Nous avons présenté dans ce chapitre, le logiciel chargé de l'élaboration de trois modèles nécessaires à une simulation globale des systèmes flexibles de production.

Outre le gain de temps apporté par l'automatisation de l'élaboration du modèle développé de la partie commande, ce logiciel s'est révélé être un véritable outil de conception assisté permettant une approche par affinements successifs. De plus, notre souci a été de conserver un degré de flexibilité maximal tant sur le plan des modèles créés que sur le plan du logiciel de conception lui-même afin d'en permettra toute reconfiguration ultérieure, par exemple par la prise en compte :

- d'une description du procédé par langage objet,
- de nouvelles architectures de machines,
- de nouveaux types de synchronisation.

CHAPITRE IV

INTRODUCTION

Dans ce dernier chapitre, nous proposons d'illustrer, à partir d'un exemple concret de cellule flexible, la méthodologie proposée et plus particulièrement l'outil de structuration présenté dans ce mémoire.

Dans ce sens, nous présenterons dans un premier temps, le site initial servant de support à l'exemple. Nous décrirons ensuite les différentes phases de la méthodologie C.A.S.P.A.I.M. appliquées à l'exemple étudié. Enfin, nous présenterons l'étude plus complète d'un projet d'extension de la cellule flexible initialement présentée.

I - PRESENTATION DE LA CELLULE INITIALE

I.1 - Architecture de la cellule /MAY 87/

La cellule flexible qui sert de support à notre exemple, est implantée dans le Département Génie Mécanique de l'I. D. N.. Cette cellule, dont le schéma d'implantation physique est donné sur la Figure 4.1, est articulée autour des éléments suivants :

i) Un tour (noté T), de type HES400 (H. ERNAULT-SOMUA), piloté par un directeur de commande NUM 760T.

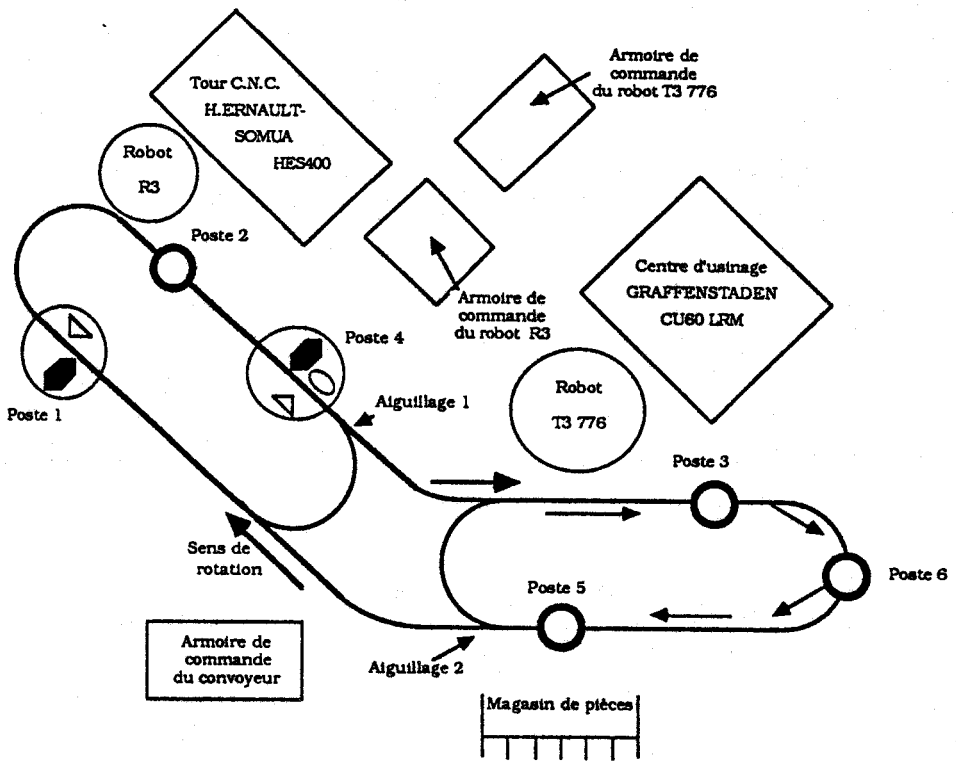
ii) Un centre d'usinage (noté F), de type CU60 LRM (GRAFFENSTADEN), piloté également par un directeur de commande NUM 760T. Ce centre d'usinage est équipé de deux tables équivalentes permettant d'effectuer les entrées/sorties de pièces.

iii) Un convoyeur à chaîne sur lequel six butées définissent trois postes de chargement/déchargement de pièces (P2, P3, P5) et trois files d'attente (P1, P4, P6). Le convoyeur, chargé du transport des pièces palettisées, est piloté par un automate programmable TE 08.

iv) Un robot manipulateur AFMA de type R3, piloté par un directeur de commande ROBO NUM 700. Ce robot est chargé de l'alimentation du tour T.

v) Un robot industriel T^3-776 (CINCINNATI) assure les entrées/sorties de pièces ainsi que le chargement/déchargement du centre d'usinage F.

vi) Un tampon d'entrée E et un tampon de sortie S.



Eléments pouvant entrer dans la composition d'un poste

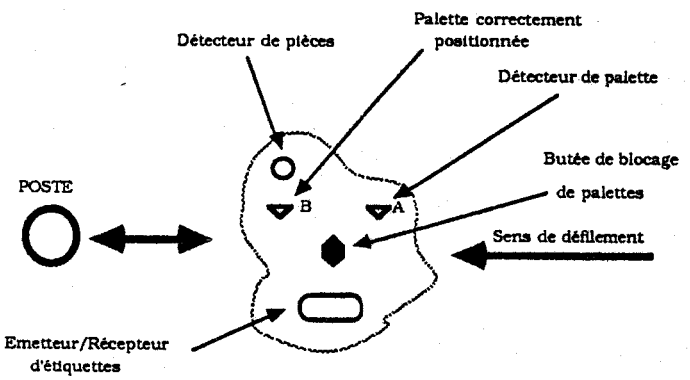


FIGURE 4.1

Dans la suite de ce chapitre, nous utiliserons les notations précisées sur le schéma simplifié de la Figure 4.2.

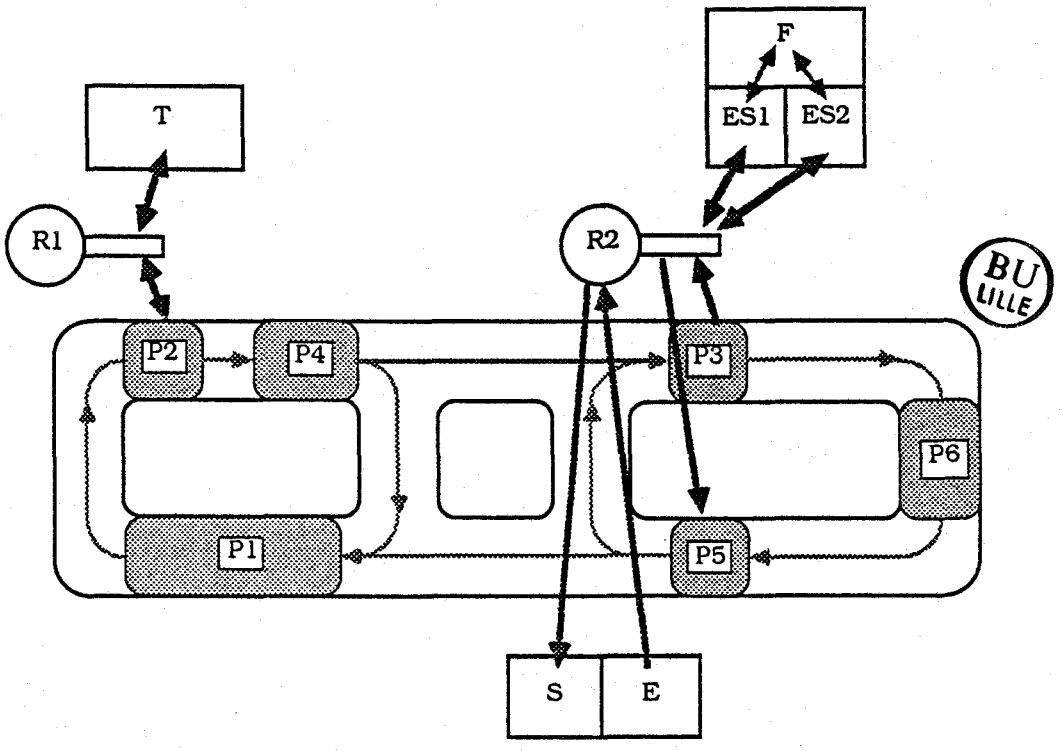


FIGURE 4.2

Sur ce schéma, les six zones définies sur le convoyeur grâce à des butées, ont les fonctions suivantes :

P1 est une zone tampon de pièces destinées au tournage,

P2 est la zone de dépalettisation/palettisation des pièces destinées au tournage,

P3 est la zone de dépalettisation des pièces destinées au centre d'usinage ou à la sortie,

P4 est une zone tampon de pièces provenant du tour,

P5 est la zone de palettisation des pièces provenant de l'entrée ou du centre d'usinage,

P6 est la zone de stockage des palettes vides.

I.2 - Les gammes opératoires

Dans cette cellule, circulent cinq types de pièces. Ces pièces sont usinées :
- soit sur le tour T, une ou deux fois,
- soit sur le centre d'usinage F,
- soit sur F puis sur T,
- soit, enfin, sur T puis sur F.

Afin de simplifier les notations, nous appellerons :
- "t" les pièces à tourner,
- "f" les pièces à fraiser,
- "-" les pièces brutes,
- "+" les pièces usinées.

Ces notations nous permettront d'identifier chaque type de pièces ainsi que son état d'avancement dans la gamme opératoire (identificateur de l'état courant). Ainsi, les cinq pièces brutes suivent les gammes opératoires suivantes :

$$t^- \longrightarrow t^+$$
$$t^-t^- \longrightarrow t^+t^- \longrightarrow t^+t^+$$
$$f^- \longrightarrow f^+$$
$$t^-f^- \longrightarrow t^+f^- \longrightarrow t^+f^+$$
$$f^-t^- \longrightarrow f^+t^- \longrightarrow f^+t^+$$

De plus, nous appellerons "p" les palettes vides et nous noterons les pièces palettisées par "p" concaténé à l'identificateur d'état courant de la pièce.

Avec les conventions précédentes, pt^+f^- désignera, par exemple, une pièce :

i) qui est palettisée,

ii) qui a été tournée,

iii) qui doit être fraisée.

Pour les cinq types de pièces brutes, nous donnons sur les Figures 4.3 à 4.7, les chemins qu'elles suivent dans le système.

t^- : - transfert et palettisation en P5 par le robot R2,
- transfert vers P1 puis P2,
- dépalettisation et transfert sur le tour par R1
  (la palette vide attend la fin d'usinage),
- usinage (transformation en t^+),
- transfert et palettisation de t^+ en P2 par R1,
- transfert vers P4 puis P3,
- dépalettisation et transfert vers la sortie par R2.

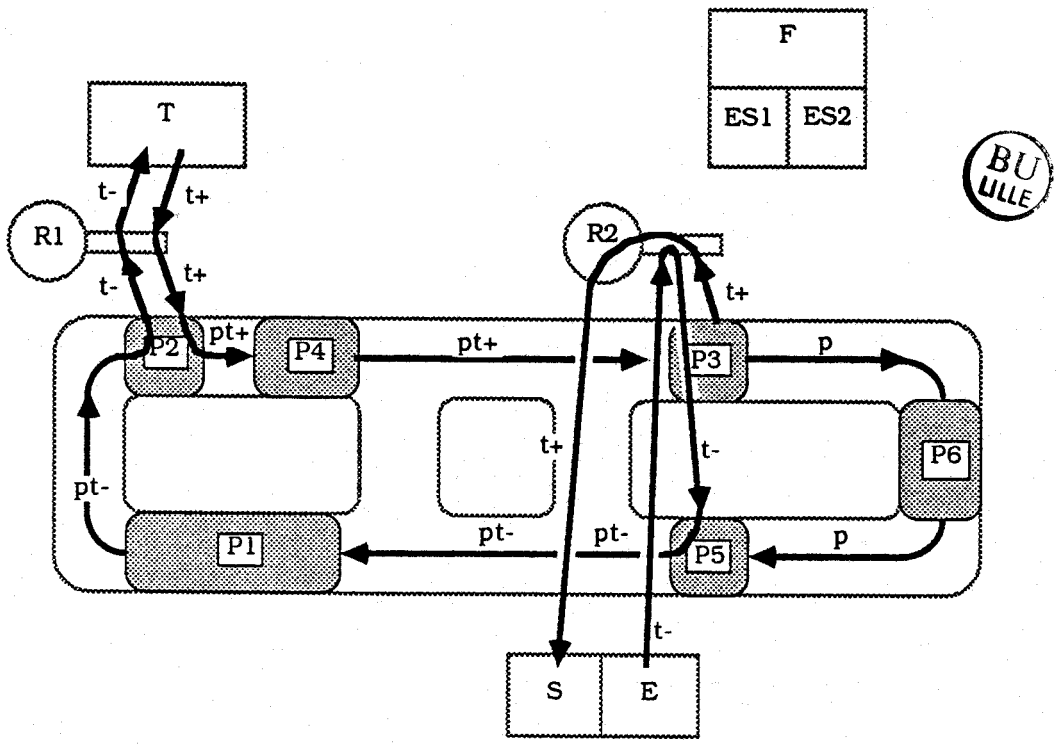


FIGURE 4.3

t^-t^- : - même début de gamme que pour la pièce t^+ jusqu'au premier usinage,
- retournement de la pièce t^+t^- par R1,
- deuxième usinage (transformation en t^+t^+),
- même fin de gamme que pour la pièce t^+.

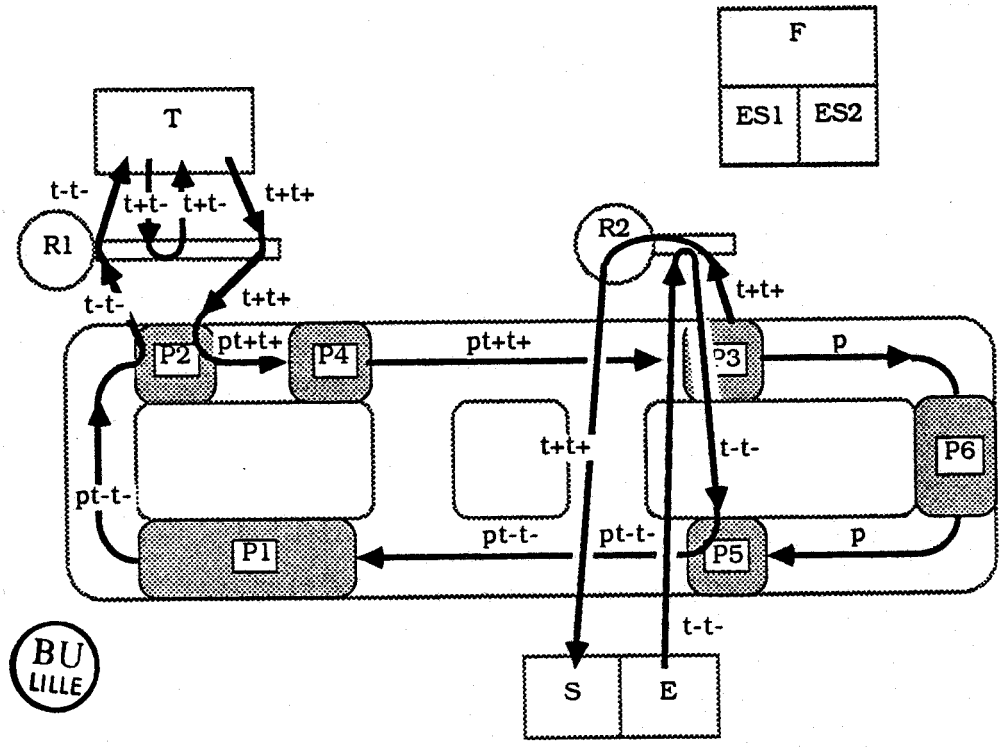


FIGURE 4.4

t^-f^- : - même début de gamme que pour la pièce t^- jusqu'à la dépalettisation en P3 de t^+f^-,
- dépalettisation et transfert par R2 sur l'un des deux tampons d'entrée/sortie du centre d'usinage,
- usinage (transformation en t^+f^+),
- transfert de t^+f^+ sur la sortie par R2.

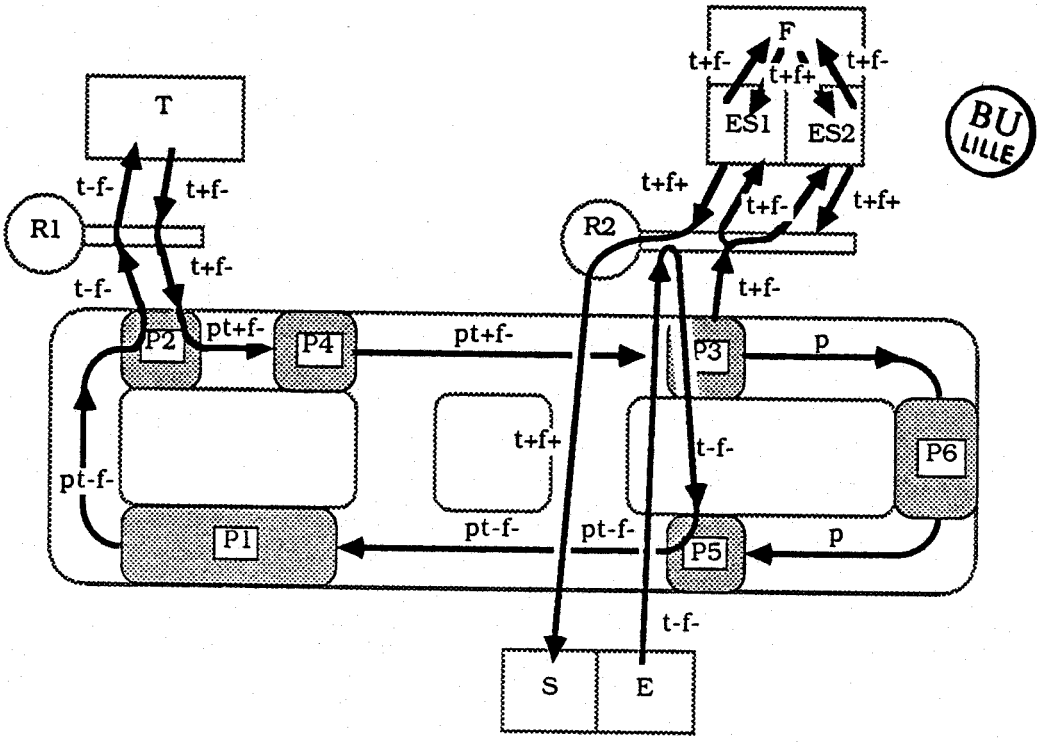


FIGURE 4.5

f^- : - transfert et palettisation par R2 sur P5,
- transfert vers P3,
- dépalettisation et transfert par R2 sur l'un des deux tampons d'entrée/ sortie du centre d'usinage,
- usinage (transformation en f^+),
- transfert de f^+ sur la sortie par R2.

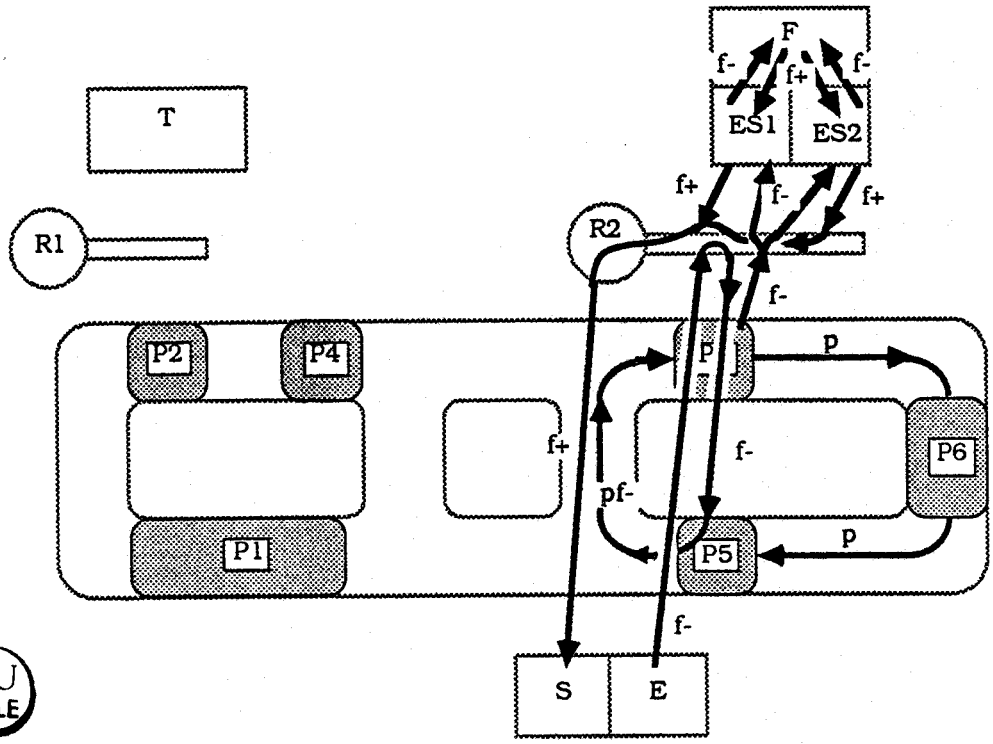


FIGURE 4.6

f^-t^- : - même début de gamme opératoire que pour la pièce f^- jusqu'à la fin d'usinage (transformation en f^+t^-),
- transfert et palettisation de f^+t^- en P5,
- même fin de gamme opératoire que pour la pièce t^-.

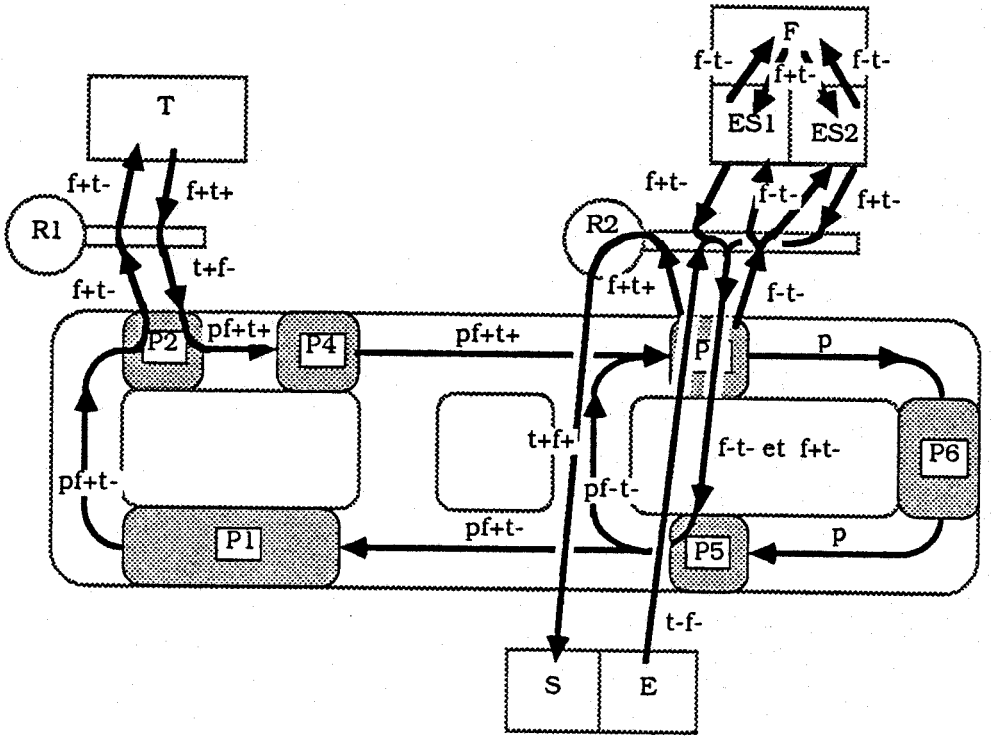


FIGURE 4.7

II - APPLICATION DE LA METHODOLOGIE C.A.S.P.A.I.M.

II.1 - Le prégraphe

La première phase de l'élaboration du prégraphe, consiste à décrire l'ensemble des gammes opératoires sous forme de règles de production et à en vérifier la complétude et la cohérence (cf. Etape 2 du projet, Chapitre II, § I.3.1). Cette première phase nous a permis de mettre en évidence un indéterminisme au niveau de la gamme t^-t^-. En effet, lorsque ce type de pièce a subi son premier usinage et a donc été transformée en t^+t^-, le système ne peut la distinguer d'une pièce de type t^+t^- qui a déjà subi le retournement par le robot R1. De ce fait, nous avons complété les informations sur l'état d'avancement de la gamme t^-t^- en distinguant les pièces de type t^+t^- **avant** retournement et **après** retournement, qui seront notées désormais t^+t^-mp et t^+t^-bp respectivement.

Cette gamme ayant été complétée, nous effectuons une agrégation des règles sous forme de réseau de Petri coloré. Nous obtenons ainsi le prégraphe de la Figure 4.8 pour lequel nous avons défini sept fonctions et neuf variables de couleurs.

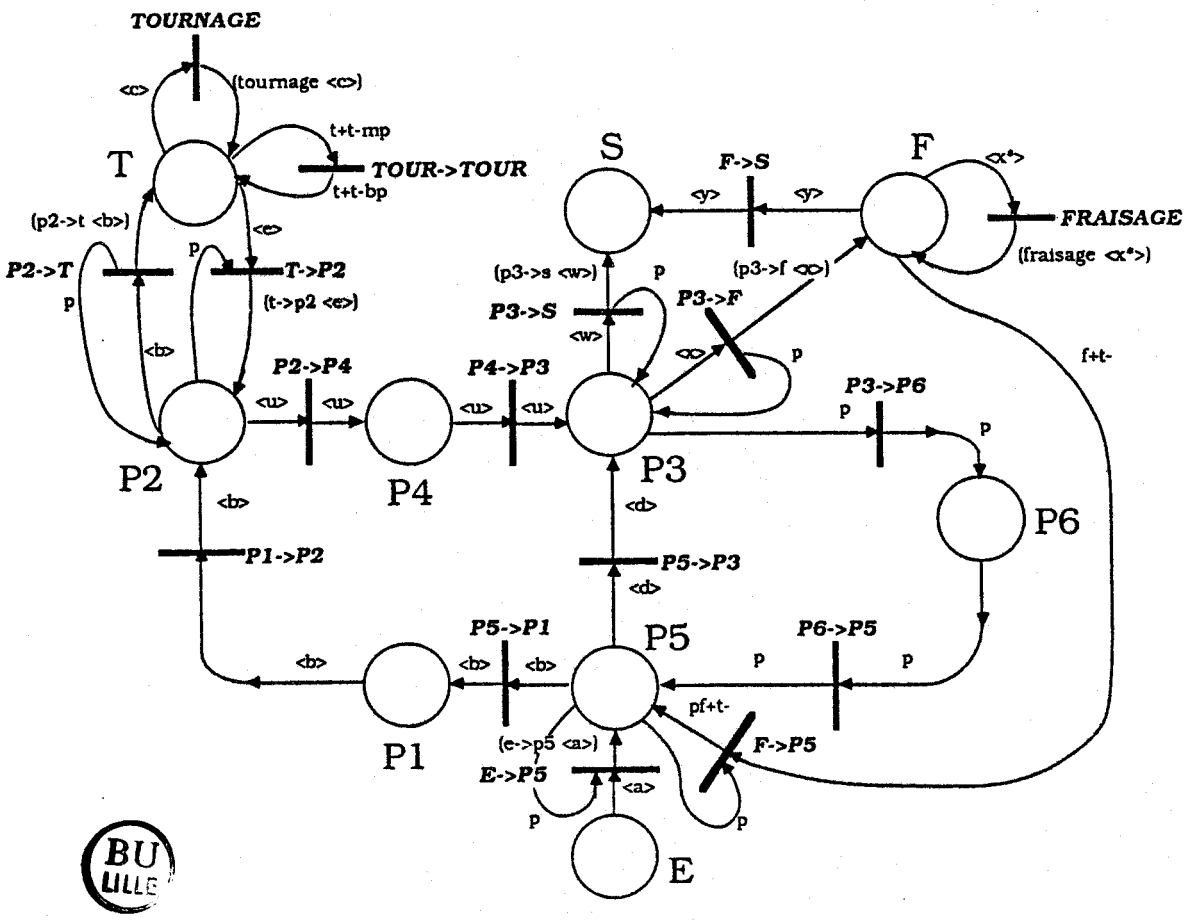


FIGURE 4.8

Nous donnons ci-après les domaines des variables de couleurs et, pour chaque fonction, son rôle et sa définition en Le_Lisp.

dom (<a>) = { t^-, t^-t^-, t^-f^-, f^-t^-, f^- }

dom (<b>) = { pt^-, pt^-t^-, pt^-f^-, pf^+t^- }

dom (<c>) = { t^-, t^-t^-, t^+t^-bp, f^+t^-, t^-f^- }

dom (<d>) = { pf^-, pf^-t^- }

dom (<e>) = { t^+, t^+t^+, f^+t^+, t^+f^- }

dom (<u>) = { pt^+, pt^+t^+, pt^+f^-, pf^+t^+ }

dom (<w>) = { pt^+, pt^+t^+, pf^+t^+ }

dom (<x>) = { pf^-, pf^-t^-, pt^+f^- }

dom (<x*>) = { f^-, f^-t^-, t^+f^- }

dom (<y>) = { f^+, t^+f^+ }

Fonction e ⟶ p5

Rôle : cette fonction définit la palettisation d'une pièce de type a (pièces brutes) correspondant au domaine de couleur $\{t^-, t^-t^-, t^-f^-, f^-t^-, f^-\}$.
Si a est égal à t^-, alors la fonction retourne la pièce palettisée pt^-.
Si a est égal à f^-, la fonction retourne la pièce palettisée pf^-, etc....

Définition :
```
(de e⟶p5 (a)
    (cond ((equal a 't-) 'pt-)
          ((equal a 'f-) 'pf-)
          ((equal a 't-t-) 'pt-t-)
          ((equal a 'f-t-) 'pf-t-)
          ((equal a 't-f-) 'pt-f-)))
```

Fonction p2 ⟶ t

Rôle : dépalettisation des pièces à tourner

Définition :
```
(de p2⟶t (b)
    (cond ((equal b 'pt-) 't-)
          ((equal b 'pt-t-) 't-t-)
          ((equal b 'pt-f-) 't-f-)
          ((equal b 'pf+t-) 'f+t-)))
```

Fonction t ⟶ p2

Rôle : palettisation des pièces tournées

Définition :
```
(de t⟶p2 (e)
    (cond ((equal e 't+) 'pt+)
          ((equal e 't+t+) 'pt+t+)
          ((equal e 'f+t+) 'pf+t+)
          ((equal e 't+f-) 'pt+f-)))
```

Fonction tournage

Rôle : usinage des pièces sur le tour

Définition :
```
(de tournage (c)
    (cond ((equal c 't-) 't+)
          ((equal c 't-t-) 't+t-mp)
          ((equal c 't+t-bp) 't+t+)
          ((equal c 't-f-) 't+f-)
          ((equal c 'f+t-) 'f+t+)))
```

Fonction p3 → s

Rôle : dépalettisation des pièces achevées

```
Définition : (de  p3 → s  (w)
                  (cond  ((equal  w  'pt+)  't+)
                         ((equal  w  'pt+t+)  't+t+)
                         ((equal  w  'pf+t+)  'f+t+)))
```

Fonction p3 → f

Rôle : dépalettisation des pièces à fraiser

```
Définition : (de  p3 → f  (x)
                  (cond  ((equal  x  'pf-)  'f-)
                         ((equal  x  'pf-t-)  'f-t-)
                         ((equal  x  'pt+f-)  't+f-)))
```

Fonction fraisage

Rôle : usinage des pièces sur le centre d'usinage

```
Définition : (de  fraisage  (x*)
                  (cond  ((equal  x*  'f-)  'f+)
                         ((equal  x*  'f-t-)  'f+t-)
                         ((equal  x*  't+f-)  't+f+)))
```

Grâce à ces définitions, le prégraphe décrit complètement les fonctionnalités de la cellule.

II.2 - L'élaboration du graphe de commande structuré

II.2.1 - Structuration du prégraphe

Dans cette phase de structuration, nous étudions pour chaque place les transitions qui y sont reliées et nous effectuons la structuration selon les primitives décrites dans le chapitre précédent.

Cette phase débute par la structuration des machines, afin de prendre en compte les contraintes imposées par les machines à plusieurs tampons d'entrée/sortie équivalents.

En effet, ces machines étant décomposées sous forme de n machines équivalentes, il est nécessaire de dupliquer toutes les transitions reliées à la place "prégraphe" représentant la machine, afin de traduire les transferts sur les différents tampons d'entrée/sortie.

De ce fait, nous avons dupliqué les transitions f⟶S, P3⟶f et f⟶P5 pour les deux tampons d'entrée/sortie du centre d'usinage que nous noterons f-1 et f-2.

Compte tenu des architectures respectives du tour et du centre d'usinage, nous développons donc tout d'abord ces machines et leur transition d'usinage (Fig. 4.9 et 4.10).

Cette étape étant achevée, nous effectuons successivement la structuration :

- des zones tampons P1, P4, P6,
- des stations d'assemblages et/ou de désassemblages P2, P3, P5,
- des transitions d'assemblages
  T ⟶ P2, E ⟶ P5, f-1 ⟶ P5 et f-2 ⟶ P5,
- des transitions de désassemblages
  P2 ⟶ T, P3 ⟶ S, P3 ⟶ f-1 et P3 ⟶ f-2,
- de la transition de positionnement Tour ⟶ Tour,
- des transitions de transfert
  P5 ⟶ P1, P1 ⟶ P2, P2 ⟶ P4, P5 ⟶ P3,
  P3 ⟶ P6, P6 ⟶ P5, f-1 ⟶ S et f-2 ⟶ S.

Ces RdP structurés sont donnés sur les Figures 4.9 à 4.14.

La Figure 4.9 représente le développement en RdP Structurés :

- du tour,
- de la transition Tour ⟶ Tour,
- des transferts des opérations de dépalettisation et palettisation représentées sur le prégraphe par les transitions
  P2 ⟶ T et T ⟶ P2.

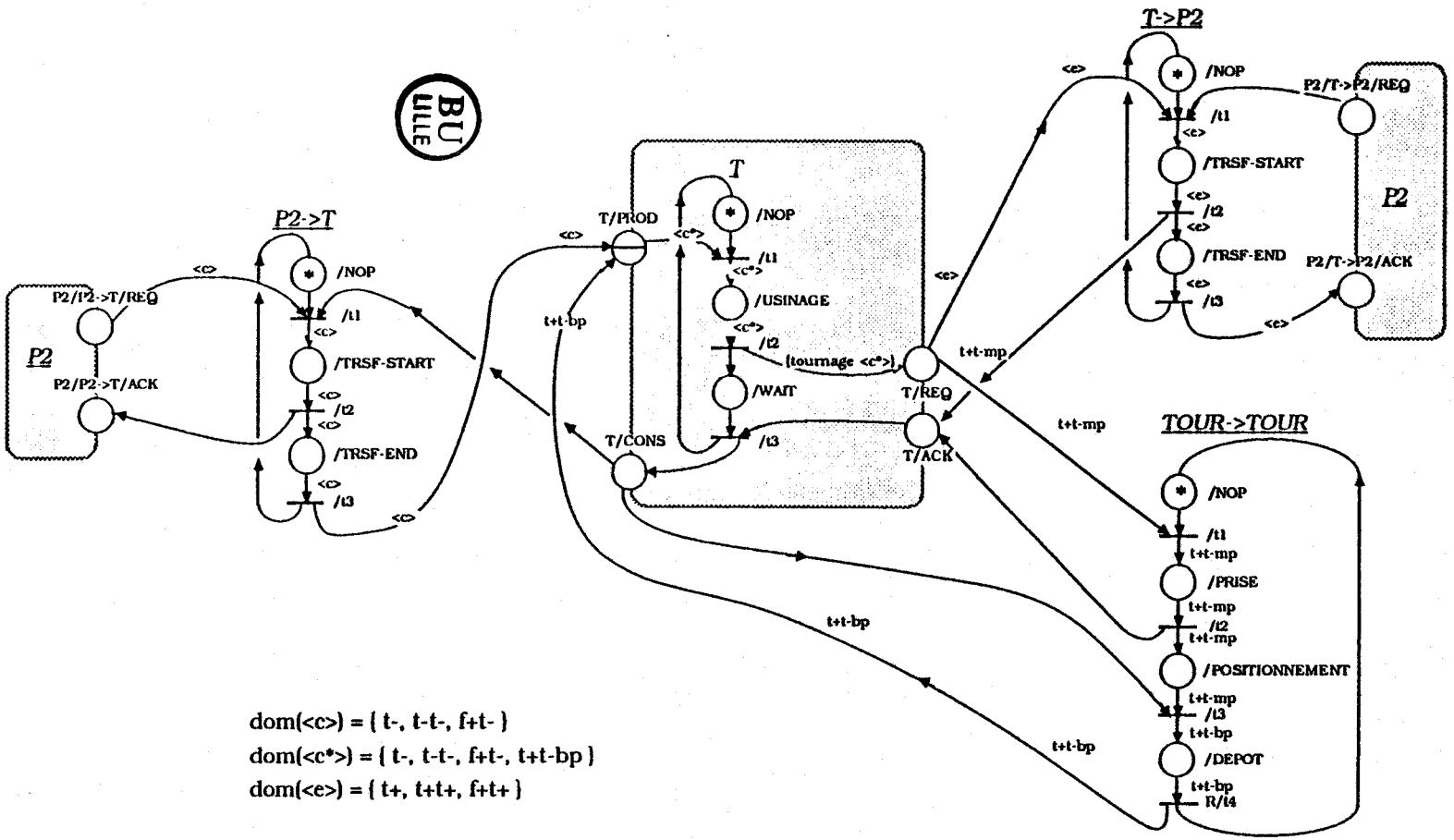


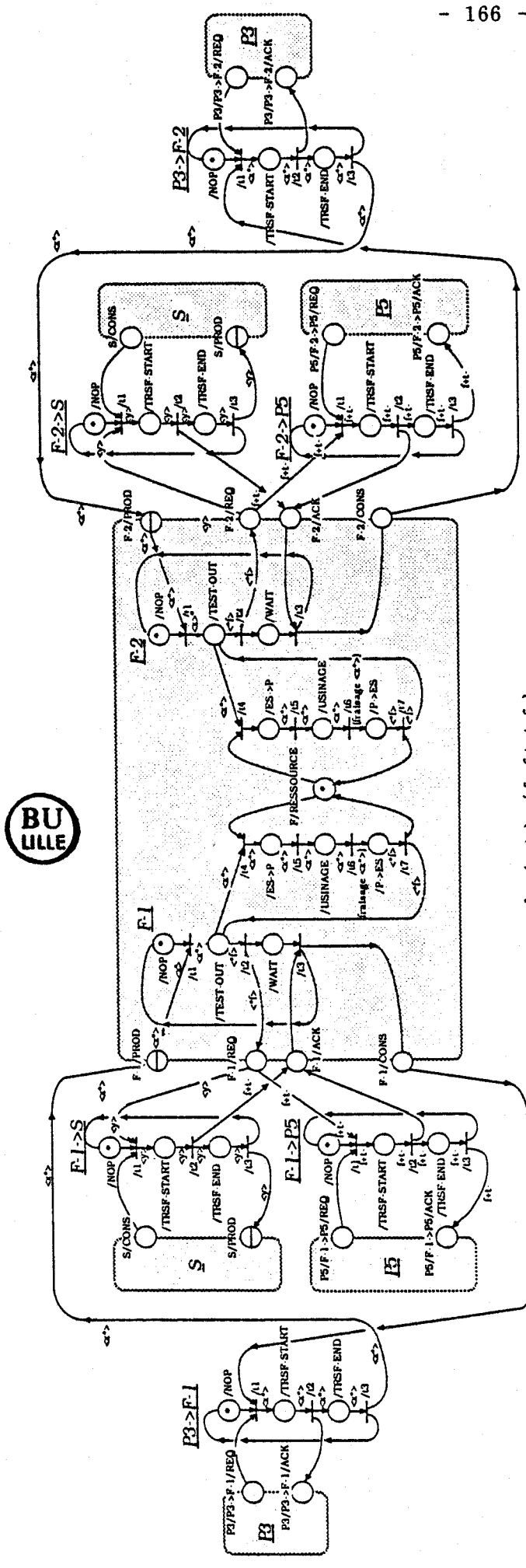
FIGURE 4.9

dom(<c>) = { t-, t-t-, f+t- }
dom(<c*>) = { t-, t-t-, f+t-, t+t-bp }
dom(<e>) = { t+, t+t+, f+t+ }

```
(de tournage (c)
    (cond  ((equal c 't-) 't+)
           ((equal c 't-t-) 't+t-mp)
           ((equal c 't+t-bp) 't+t+)
           ((equal c 't-f-) 't+f-)
           ((equal c 'f+t-) 'f+t+)))
```

La Figure 4.10 représente le développement en RdP Structurés :

- du centre d'usinage,
- des transitions de transfert F-1 $\longrightarrow$ S et F-2 $\longrightarrow$ S,
- des transferts des opérations de palettisation représentées par les transitions F-1 $\longrightarrow$ P5 et F-2 $\longrightarrow$ P5,
- des transferts des opérations de dépalettisation représentées par les transitions P3 $\longrightarrow$ F-1 et P3 $\longrightarrow$ F-2.



dom(<x*>) = { f-, f-t-,t+f- }
dom(<*f>) = { f+, f+t-, t+f+ }
dom(<y>) = { f+, t+f+ }

```
(de  fraisage (x)
     (cond ((equal x 'f-) 'f+)
           ((equal x 'f-t-) 'f+t-)
           ((equal x 't+f-) 't+f+)))
```

FIGURE 4.10

La Figure 4.11 représente le développement en RdP Structurés :

- des zones tampons P1, P4, P6,
- des différents transferts correspondant aux transitions du pré-graphe P1 ⟶ P2, P2 ⟶ P4, P4 ⟶ P3, P3 ⟶ P6,
  P6 ⟶ P5, P5 ⟶ P1, P5 ⟶ P3.

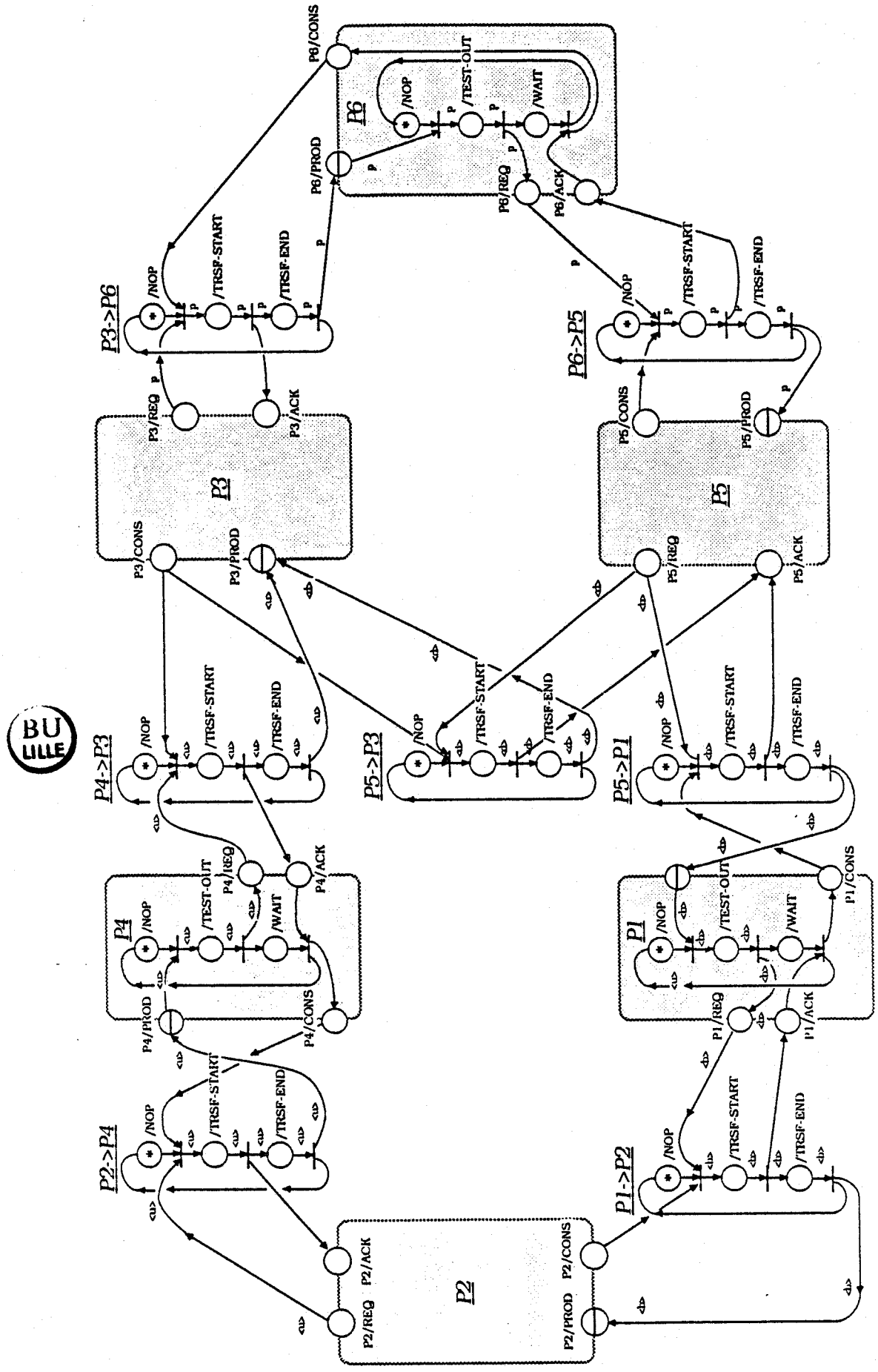
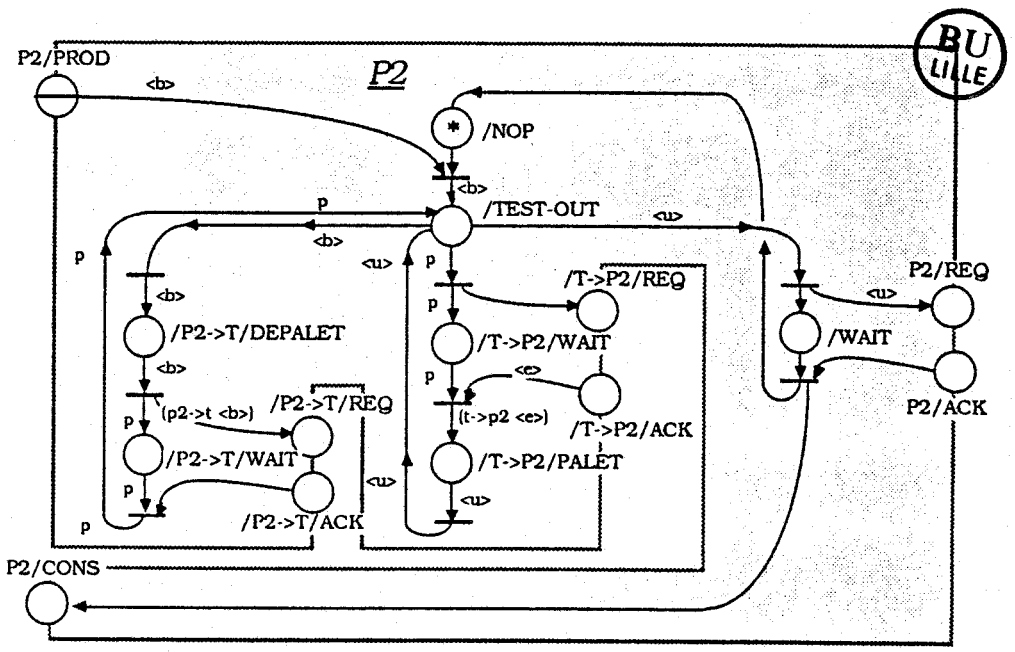

P6
P6/CONS
P6/PROD
P6/REQ
P6/ACK
/NOP
/TEST-OUT
/WAIT
P3->P6
P6->P5
P3
P3/CONS
P3/PROD
P3/REQ
P3/ACK
P5
P5/CONS
P5/PROD
P5/REQ
P5/ACK
/TRSF-START
/TRSF-END
P4->P3
P5->P3
P5->P1
P4
P4/PROD
P4/CONS
P4/REQ
P4/ACK
P1
P1/CONS
P1/REQ
P1/ACK
P2->P4
P1->P2
P2
P2/CONS
P2/PROD
P2/REQ
P2/ACK

FIGURE 4.11

Les Figures 4.12, 4.13 et 4.14 représentent respectivement, le développement en RdP Structurés :

- de la place P2,
  de la transition de dépalettisation P2 ⟶ T,
  et de la transition de palettisation T ⟶ P2,
- de la place P3,
  et des transitions de dépalettisation P3 ⟶ F-1,
  P3 ⟶ F-2 et P3 ⟶ S,
- de la place P5,
  et des transitions de palettisation F-1 ⟶ P5,
  F-2 ⟶ P5, E ⟶ P5.

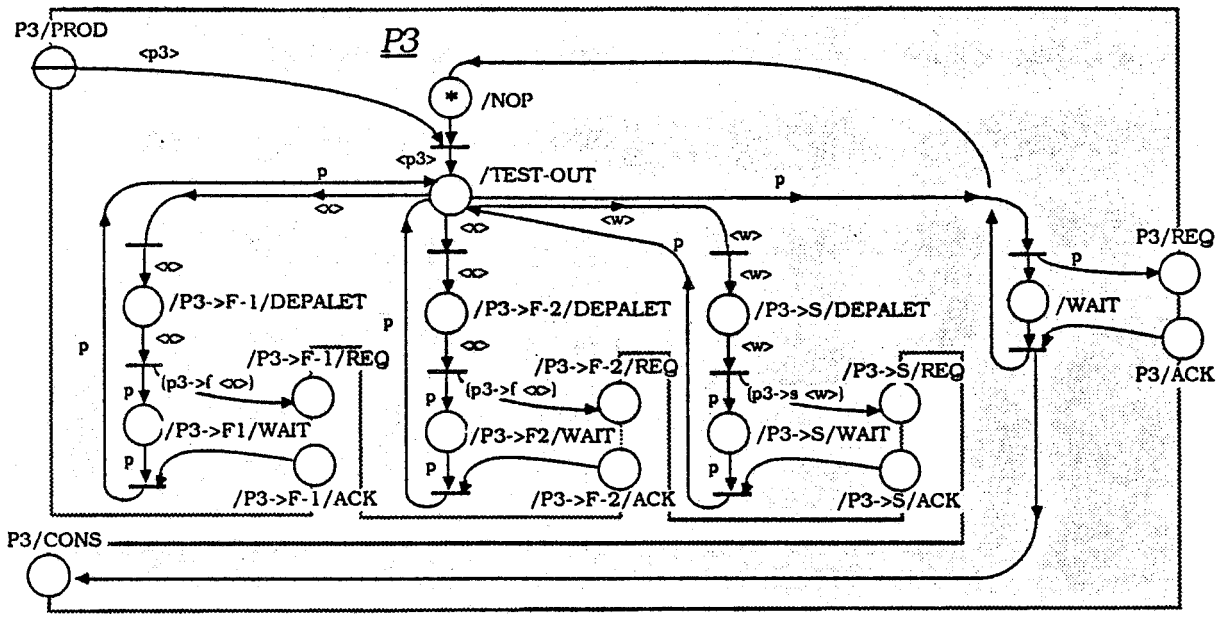


dom(<b>) = { pt-, pt-t-, pt-f-, pf+t- }
dom (<e>) = { t+, t+t+, t+f-, f+t+}
dom (<u>) = { pt+, pt+t+, pt+f-, pf+t+}

```
(de p2->t (b)
    (cond ((equal b 'pt-) 't-)
          ((equal b 'pt-t-) 't-t-)
          ((equal b 'pt-f-) 't-f-)
          ((equal b 'pf+t-) 'f+t-)))
```

```
(de t->p2 (e)
    (cond ((equal e 't+) 'pt+)
          ((equal e 't+t+) 'pt+t+)
          ((equal e 't+f-) 'pt+f-)
          ((equal e 'f+t+) 'pf+t+)
```

FIGURE 4.12

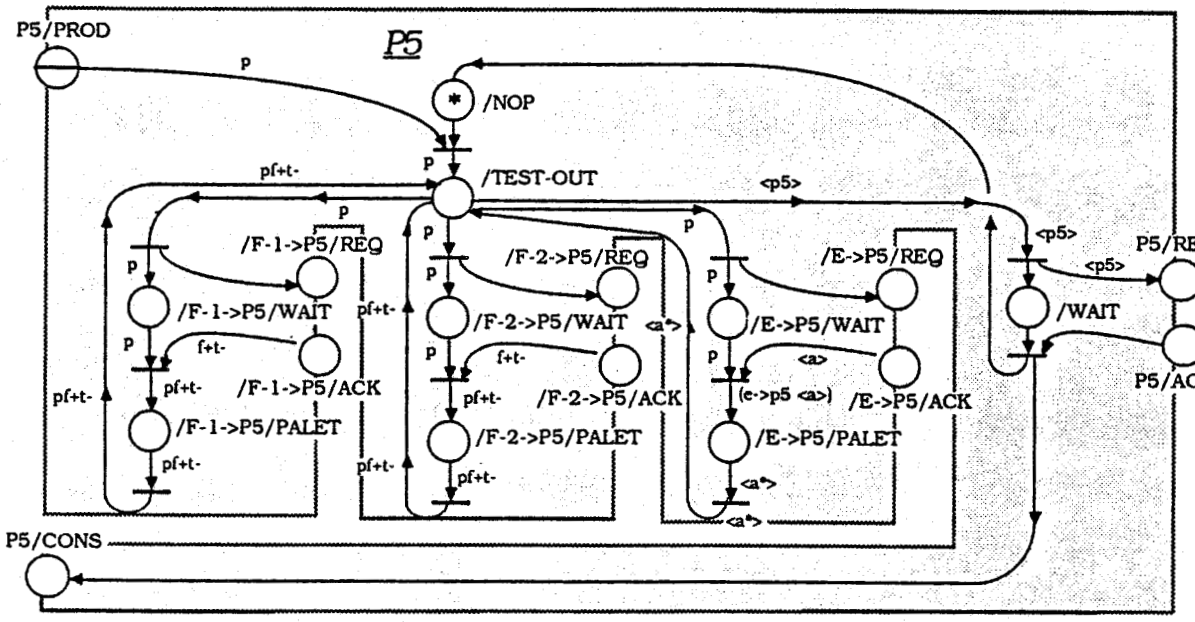


dom(<p3>) = { pt+, pt+t+, pt+f-, pf+t+ , pf-, pf-t- }
dom(<>) = { pt+f-, pf-, pf-t- }
dom(<w>) = { pt+, pt+t+, pf+t+ }

```
(de p3->f (x)
    (cond ((equal x 'pf-) 'f-)
          ((equal x 'pf-t-) 'f-t-)
          ((equal x 'pt+f-) 't+f-)))
```

```
(de p3->s (w)
    (cond ((equal w 'pt+) 't+)
          ((equal w 'pt+t+) 't+t+)
          ((equal w 'pf+t+) 'f+t+)))
```

FIGURE 4.13



dom(<a>) = { t-, t-t-, t-f-, f-t-, f- }
dom(<a*>) = { pt-, pt-t-, pt-f-, pf-t-, pf- }
dom(<p5>) = { pt-, pt-t-, pt-f-, pf-t-, pf-, pf+t- }

```
(de e->p5 (a)
    (cond ((equal a 't-) 'pt-)
          ((equal a 't-t-) 'pt-t-)
          ((equal a 't-f-) 'pt-f-)
          ((equal a 'f-t-) 'pf-t-)
          ((equal a 'f-) 'pf-)))
```


FIGURE 4.14

Note : Afin de ne pas surcharger les figures, nous n'indiquerons pas systématiquement tous les noms de places et transitions. Une description complète du graphe global est donnée en Annexe, sous la forme textuelle éditée par notre logiciel.

Nous ajoutons ensuite automatiquement un sémaphore noté mutex-entrée/p3 sur les deux processus de transfert correspondant au développement des transitions P5 ⟶ P3 et P4 ⟶ P3 pour éviter les collisions de palettes à l'entrée de la zone P3.

Cette dernière étape clôture la phase de structuration du prégraphe. A ce stade de la conception, le graphe global est composé de 215 places, 126 transitions formant 31 processus et leurs interconnexions.

II.2.2 - Répartition des processus de commande

Pour prendre en compte les contraintes imposées par le cahier des charges, nous créons de manière interactive deux ressources critiques R1 et R2 correspondant respectivement aux deux robots. Ces deux ressources critiques sont connectées aux processus et places correspondant aux actions que ces robots auront à effectuer de manière exclusive.

Ainsi, la ressource R1 sera connectée :
- au processus de positionnement Tour ⟶ Tour,
- aux processus de transfert p2 ⟶ t et t ⟶ p2,
- aux actions
  p2 ⟶ t / dépalettisation et t ⟶ p2 / palettisation.

La ressource R2 sera connectée :
- aux processus de transfert
  e ⟶ p5, f-1 ⟶ p5, f-2 ⟶ p5, p3 ⟶ S,
  p3 ⟶ f-1, p3 ⟶ f-2, f-1 ⟶ S, f-2 ⟶ S,
- aux actions
  e ⟶ p5 / palettisation, f-1 ⟶ p5 / palettisation,
  f-2 ⟶ p5 / palettisation, p3 ⟶ S / dépalettisation,
  p3 ⟶ f-1 / dépalettisation, p3 ⟶ f-2 / dépalettisation.

La liste des conflits générés par toutes les liaisons de type exclusion mutuelle mais également par les configurations présentées dans le Chapitre III § IV.2.2 est donnée en Annexe. Cette liste est celle fournie par notre logiciel.

II.2.3 - Les règles du niveau hiérarchique

Afin de résoudre les conflits, nous avons choisi une politique de prudence visant à éviter le blocage du système par engorgement des organes de transfert et des tampons de stockage. Ainsi, c'est l'état d'avancement des pièces dans leur gamme opératoire qui détermine la pièce prioritaire en cas de conflit.

La liste des règles est donnée de manière condensée en Annexe.

II.2.4 - Le dimensionnement et la temporisation

Nous affectons aux différents tampons, une taille maximale ainsi qu'un état initial en accord avec le système réel. Pour chaque zone tampon, la taille maximale est la suivante :

P1 : 5
P2 : 1
P3 : 1
P4 : 4
P5 : 1
P6 : 10

Toutes les zones tampon sont vides sauf P6 qui contient initialement 10 palettes vides "p".

De la même façon, nous affectons une temporisation à chaque action. Ces temporisations sont les suivantes :

Pour le convoyeur :

- 51 s pour le trajet depuis P5 vers P1
  (30 entre P1 et l'aval de l'aiguillage venant de P4 et 21 pour le reste),
- 14 s pour le trajet de P1 vers P2,
- 21 s pour le trajet de P2 vers P4,
- 35 s pour le trajet de P4 vers P3,
- 23 s pour le trajet de P5 vers P3,
- 28 s pour le trajet des palettes vides de P3 vers P6,
- 18 s pour le trajet des palettes vides de P6 vers P5.

Pour les actions de palettisation et de dépalettisation : 10 s

Pour les transferts de pièces par les robots Afma et Cincinnati : 20 s

Pour la fraiseuse :

- 20 s pour les transferts entre les tampons d'entrée/sortie et le point d'usinage,
- 600 s pour l'usinage $f^-t^- \longrightarrow f^+t^-$
  550 s pour l'usinage $f^- \longrightarrow f^+$
  500 s pour l'usinage $t^+f^- \longrightarrow t^+f^+$

Pour le tour :

- 30 s pour la prise, le retournement et la dépose s'il y a lieu (pièces t^-t^- après le 1er usinage),
- 400 s pour l'usinage $t^+t^- \longrightarrow t^+t^+$
  350 s pour l'usinage $f^+t^- \longrightarrow f^+t^+$
  300 s pour l'usinage $t^-f^- \longrightarrow t^+f^-$
  250 s pour l'usinage $t^-t^- \longrightarrow t^+t^-$
  200 s pour l'usinage $t^- \longrightarrow t^+$

II.3 - Simulation du graphe de l'atelier

II.3.1 - Introduction

Par la simulation, nous essayons d'atteindre un double objectif. D'une part, on cherche à vérifier la bonne définition du niveau hiérarchique qui pilote le graphe de commande. D'autre part, on se propose de fournir des résultats quantitatifs sur le fonctionnement du système en associant une durée aux places du graphe associées aux actions temporisées du système.

Cette session de simulation est décomposée selon les étapes suivantes :

1) Simulation non temporisée et choix d'une politique d'entrée aléatoire et au plus tôt.
2) Simulation temporisée et introduction sélective d'une pièce à la fois.
3) Simulation temporisée et choix d'une politique d'entrée au plus tôt.

II.3.2 - Simulation non temporisée et entrée aléatoire au plus tôt

Le but de cette phase de simulation consiste à vérifier que les règles choisies pour le niveau hiérarchique assurent la sécurité d'accès aux ressources critiques du système. Ces ressources critiques sont constituées par :

- les points de palettisation ou de dépalettisation (P5, P2, P3),
- les aiguillages du convoyeur,
- les points d'usinage du tour et de la fraiseuse,
- les tampons d'entrée/sortie de la fraiseuse.

La simulation du graphe non temporisé où les entrées sont aléatoires et au plus tôt (flux continu de pièces disponibles en entrée), n'a mis en évidence aucun blocage ni aucune situation anormale, ce qui tend à confirmer la bonne définition du niveau hiérarchique choisi.

II.3.3 - Simulation temporisée et introduction sélective des pièces

Le but de cette phase de simulation est de donner une première idée des performances du système.

La simulation avec entrée unitaire permet de connaître le temps minimal de conditionnement d'une pièce dans le système. La pièce étant seule dans le système, elle a donc accès immédiatement à toutes les ressources du système. Le temps de conditionnement fourni par la simulation est donc bien minimal. Afin de stopper la simulation à la sortie de la pièce étudiée, nous avons défini un point d'arrêt sur la fifo modélisant la sortie (place s/req).

Entrée d'une pièce de type t^- :

Il y a blocage du graphe à la date 459.

Sur ces 459 s, la pièce a attendu 18 s en entrée qu'une palette arrive de P6 vers P5, a été usinée 200 s et transférée 241 s.

Ce temps se décompose donc en :

44 % d'usinage,

52 % de transfert,

4 % d'attente.

Entrée d'une pièce de type f^- :

Le graphe se bloque à la date 711.

Ce temps minimal de conditionnement se décompose en :

77 % d'usinage (550 s),

20 % de transfert (143 s),

3 % d'attente (18 s).

Entrée d'une pièce de type t^-t^- :

Le graphe se bloque à la date 939, soit :

69 % d'usinage (650 s),

29 % de transfert (271 s),

2 % d'attente (18 s).

Entrée d'une pièce de type t^-f^- :

Le graphe se bloque à la date 1119, soit :

71 % d'usinage (800 s),

27 % de transfert (301 s),

2 % d'attente (18 s).

Entrée d'une pièce de type f^-t^- :

Le graphe se bloque à la date 1228, soit :

77 % d'usinage (950 s),

21 % de transfert (260 s),

2 % d'attente (18 s).

II.3.4 - Simulation temporisée et entrée au plus tôt

Cette phase de simulation a pour but de déterminer, en fonctionnement normal, l'influence de la séquence d'entrée des pièces sur le système et de détecter d'éventuels cas de blocage. Il s'agit ici de blocages de type "deadlock" (ou "étreintes fatales"), chaque processus attendant pour évoluer, la libération de "ressources de production" par un autre processus en aval. Nous avons détecté, grâce à la simulation, 2 situations bloquantes de ce genre.

a) 1ère Situation

La séquence aléatoire gérant l'entrée des pièces fournie au système (soit 11 pièces) est la suivante :

t^-f^-	f^-t^-	t^-	t^-t^-	t^-t^-	t^-t^-	f^-t^-	t^-t^-	t^-	f^-t^-	t^-f^-
1re	2e	3e	4e	5e	6e	7e	8e	9e	10e	11e

Dans de telles conditions, la simulation s'interrompt par blocage du graphe à la date 4037. L'étreinte fatale est alors la suivante :

- La 1re pièce, après usinage sur le tour et transfert devant la fraiseuse, attend en P3 (marque pt^+f^- dans la place p3/test-out) que l'un des 2 tampons d'entrée/sortie de la fraiseuse se libère. Or, ceux-ci sont occupés par la 2e et la 7e pièce.

- La 2e pièce, après usinage sur la fraiseuse, attend sur le tampon d'entrée/sortie f-2 (marque f^+t^- dans la place f-2/req) que la zone de palettisation P5 occupée par la 10e pièce le libère afin d'être acheminée vers le tour.

- La 7e pièce est dans la même situation sur le tampon f-1 (marque f^+t^- dans la place f-1/req).

- La 10e pièce, après entrée et palettisation en P5 (marque pf^-t^- dans la place p5/req) attend que la zone P3 occupée par la 1re pièce soit libre.

Ces 4 pièces forment un deadlock au niveau des zones P3 et P5 qui paralyse tout le système. Les autres pièces sont elles aussi bloquées, puisque la zone P3 par où doivent passer toutes les pièces participe à cette étreinte fatale.

- La 3e pièce qui est complètement usinée, attend en tête de la fifo P4 (marque pt^+ dans la place p4/req) la libération de P3 afin d'y être dépalettisée et évacuée. Elle bloque derrière elle les pièces 4, 5, 6 et 8 (marques pt^+t^+ dans la place p4/prod). La fifo P4, déclarée de taille 5, est donc pleine.

- La 9e pièce, après usinage sur le tour et palettisation en P2 (marque pt^+ dans la place p2/req) attend une place libre dans la fifo P4.

- Enfin, la 11e pièce (marque t^-f^- dans la place e/req) attend dans l'organe d'entrée que le tampon d'entrée P5 occupé par la pièce 10 soit libre.

b) 2ème Situation

f^-t^-	f^-t^-	t^-f^-	t^-	t^-	t^-	t^-	t^-	t^-	t^-	t^-	t^-
1re	2e	3e	4e	5e	6e	7e	8e	9e	10e	11e	12e

Dans de telles conditions, la simulation s'interrompt par blocage du graphe à la date 2253. L'étreinte fatale est alors la suivante :

- La 1re pièce, après usinage sur la fraiseuse, attend sur le tampon d'entrée/sortie f-1 (marque f^+t^- dans la place f-1/req) qu'une palette soit présente dans la zone de palettisation P5 afin d'être acheminée vers le tour.

- La 2e pièce est dans la même situation sur le tampon f-2 (marque f^+t^- dans la place f-2/req).

- La 3e pièce, après usinage sur le tour et transfert devant la fraiseuse, attend en P3 (marque pt^+f^- dans la place p3/test-out) que l'un des 2 tampons d'entrée/sortie de la fraiseuse se libère. Or, ceux-ci sont occupés par la 1re et la 2e pièce.

- La 4e pièce qui est complètement usinée, attend en tête de la fifo P4 (marque pt^+ dans la place p4/req) la libération de P3 afin d'y être dépalettisée et évacuée. Elle bloque derrière elle les pièces 5, 6, 7 et 8 (marques pt^+ dans la place p4/prod). La fifo P4, déclarée de taille 5, est donc pleine.

- La 9e pièce, après usinage sur le tour et palettisation en P2 (marque pt^+ dans la place p2/req) attend une place libre dans la fifo P4.

- La 10e et la 11e pièce (marque pt^- dans les places p1/req et p1/prod) attendent dans la fifo P1 la libération de P2 occupée par la pièce 4.

- Enfin, la 12e pièce attend dans le tampon d'entrée qu'une palette arrive en P5 afin d'y être palettisée.

C'est le manque de palettes vides qui engendre ici le blocage du système. Les pièces présentes sur les tampons de la fraiseuse attendent une palette vide en P5. Or, toutes les palettes sont bloquées dans la portion du convoyeur desservant le tour.

II.3.5 - Conclusion sur la simulation

La simulation nous a permis ici de valider d'une part la logique de commande du système associé au niveau hiérarchique de base permettant de résoudre les indéterminismes de ce graphe, c'est-à-dire de déterminer les priorités d'accès aux diverses ressources du système.

Par contre, le dernier jeu de simulation (§ II.3.4) a mis en évidence l'existence d'un ensemble de blocages (dont 2 seulement ont été présentés en détail dans ce mémoire) inhérents à la séquence d'entrée des pièces dans le système.

Notons cependant, que ces blocages peuvent être résolus dans le cadre de modélisation que nous avons choisi. En effet, toutes les informations nécessaires sont disponibles au sein du graphe de commande (nature et localisation des pièces présentes) et peuvent donc être utilisées par une couche supplémentaire du niveau hiérarchique (à définir) apte à résoudre ce genre de problème. Un des moyens d'action de ces règles spécialisées serait, par exemple, d'interdire momentanément l'introduction d'une nouvelle pièce dans le système si celle-ci risque d'engendrer, en se combinant avec les pièces déjà entrées, une séquence bloquante. Ce genre de considérations s'apparente aux techniques de Gestion de Production classique.

Une fois définie et validée (par simulation) cette couche hiérarchique de supervision de la séquence d'entrée, l'étape suivante consisterait à optimiser les performances du système. Divers jeux d'essais jouant sur la taille des diverses fifo définies au niveau du convoyeur (P1, P4 et P6), permettraient de maximiser le temps d'engagement des machines tout en minimisant les en-cours, quelle que soit la séquence d'entrée. Rappelons qu'une telle optimisation nécessite la description du procédé afin de recueillir les statistiques de fonctionnement indispensables à cette approche.

III - PROJET D'EXTENSION DE LA CELLULE INITIALE

Nous nous intéressons dans ce paragraphe à un projet d'extension de la cellule flexible présentée dans ce chapitre au § I. Ce projet étant encore actuellement dans une phase de pré-définition, nous nous limiterons à une description de la nouvelle architecture et du prégraphe correspondant.

III.1 - Présentation de la nouvelle architecture

La nouvelle architecture à laquelle nous nous intéressons ici est composée de deux parties essentielles.

La première est constituée par la cellule dont les différentes composantes ont été présentées dans le paragraphe I et auxquelles nous avons adjoint un poste de bridage/débridage (noté PDB) des pièces à fraiser.

La seconde est constituée :

i) d'un second convoyeur connecté au premier par une traversée de jonction double (notée A3),
ii) d'une station de nettoyage (notée SN),
iii) d'un poste de métrologie (noté PM),
iv) d'un robot R3 assurant l'évacuation des pièces achevées.

L'ensemble des deux cellules constituant le nouvel atelier flexible est représenté sur le schéma de principe de la Figure 4.15.

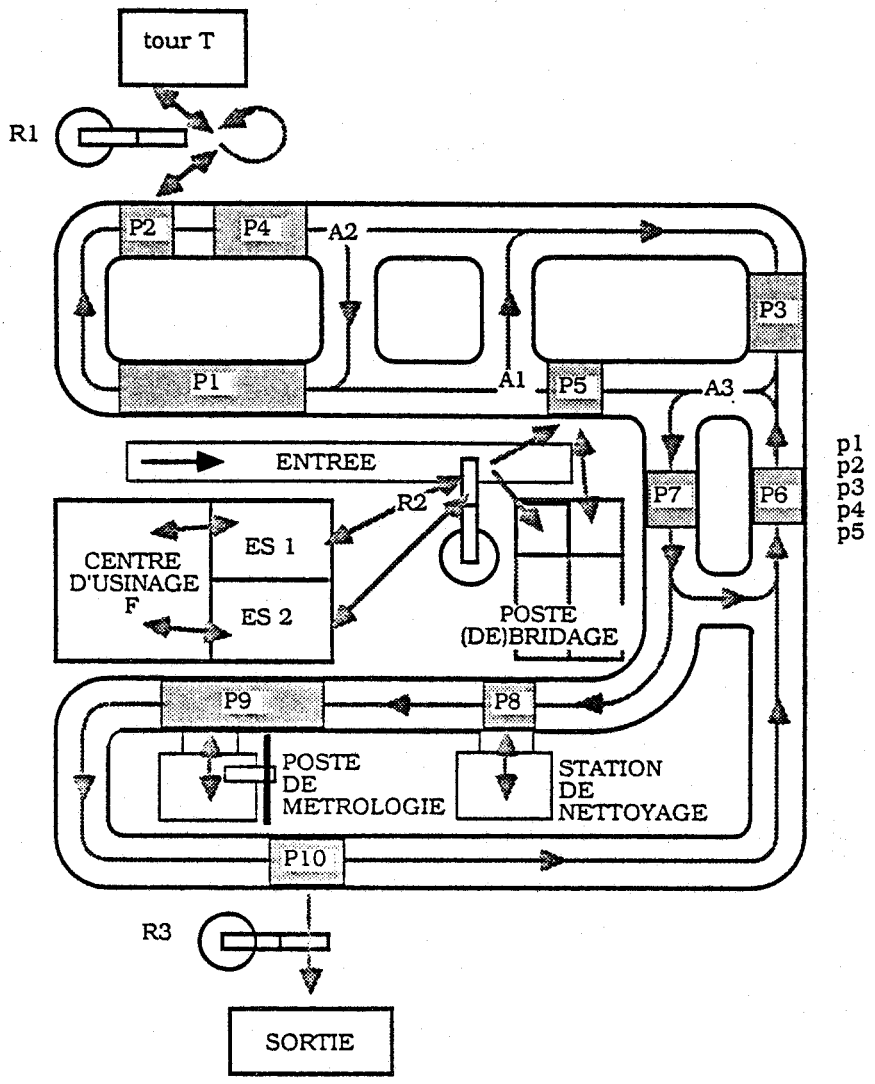


FIGURE 4.15

Dans cet atelier, seront usinés sept types de pièces qui seront transférées sur cinq types de palettes. En conservant les notations du premier paragraphe, et en appelant "l" et "m" les pièces qui passeront respectivement sur la station de nettoyage et sur le poste de métrologie, nous donnons ci-après, de manière simplifiée, les sept gammes opératoires et les palettes correspondantes (notées p1, ..., p5).

$\rightarrow t^-l^-m^- \rightarrow t^+l^-m^- \rightarrow t^+l^+m^- \rightarrow t^+l^+m^+$

avec une palette p1

$\rightarrow t^-t^-l^-m^- \rightarrow t^+t^-l^-m^- \rightarrow t^+t^+l^-m^- \rightarrow t^+t^+l^+m^- \rightarrow t^+t^+l^+m^+$

avec une palette p1

$\rightarrow t^-f^-l^-m^- \rightarrow t^+f^-l^-m^- \rightarrow t^+f^+l^-m^- \rightarrow t^+f^+l^+m^- \rightarrow t^+f^+l^+m^+$

avec une palette p2

$\rightarrow f^-f^-l^-m^- \rightarrow f^+f^-l^-m^- \rightarrow f^+f^+l^-m^- \rightarrow f^+f^+l^+m^- \rightarrow f^+f^+l^+m^+$

avec une palette p3

$\rightarrow f^-t^-l^-m^- \rightarrow f^+t^-l^-m^- \rightarrow f^+t^+l^-m^- \rightarrow f^+t^+l^+m^- \rightarrow f^+t^+l^+m^+$

avec une palette p4

$\rightarrow f^-t^-f^-l^-m^- \rightarrow f^+t^-f^-l^-m^- \rightarrow f^+t^+f^-l^-m^-$
$\rightarrow f^+t^+f^+l^-m^- \rightarrow f^+t^+f^+l^+m^- \rightarrow f^+t^+f^+l^+m^+$

avec une palette p5

III.2 - Elaboration du graphe de commande

L'analyse des différentes gammes nous donne le prégraphe de la Figure 4.16 que nous avons structuré avec notre logiciel.

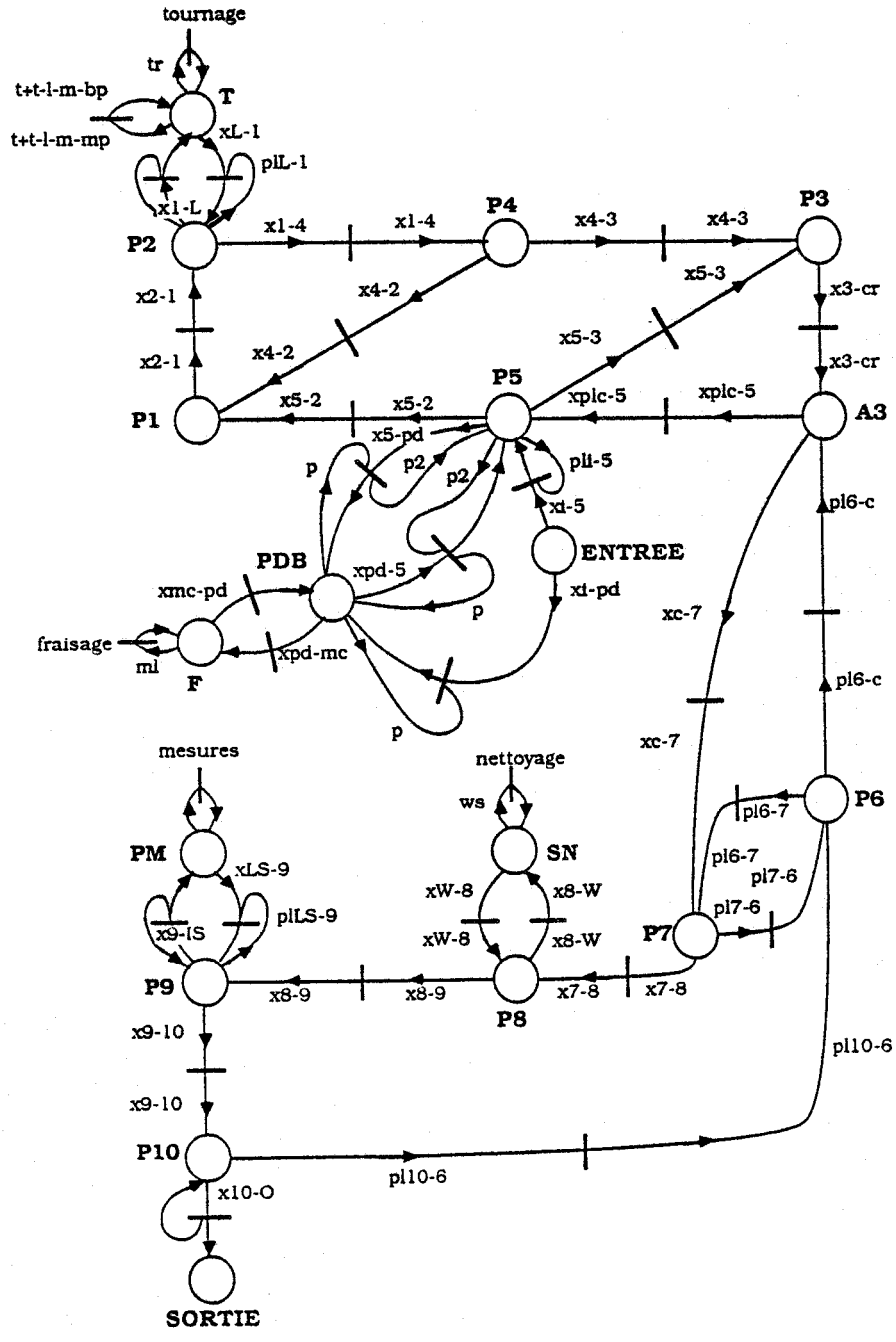


FIGURE 4.16

Le graphe développé obtenu comporte 340 places et 201 transitions formant ainsi 51 processus. La structuration d'un tel prégraphe ne pose aucune difficulté d'ordre méthodologique. La seule limitation provient de la taille du graphe qui ralentit considérablement la simulation : il faut en effet environ 15 minutes de temps CPU sur Macintosh pour qu'une pièce de type $t^-l^-m^-$ termine sa gamme opératoire.

Il paraît donc intéressant, voire nécessaire, de décomposer de telles installations en sous-systèmes relativement indépendants, de simuler isolément chaque sous-système en utilisant ses sorties comme entrées du sous-système suivant.

Dans notre exemple, une décomposition posible est donnée sur le schéma de la Figure 4.17.

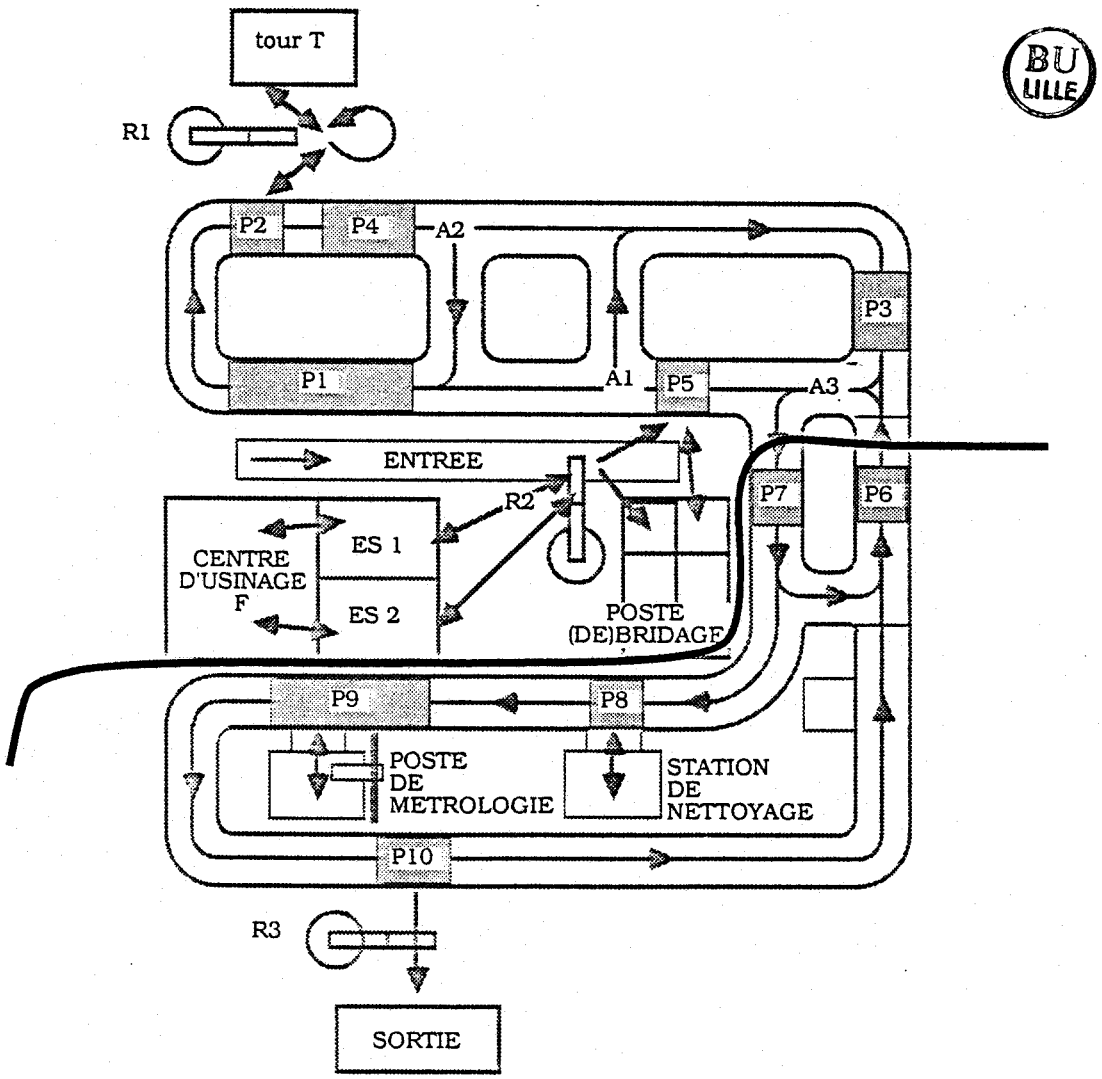


FIGURE 4.17

CONCLUSION

Nous avons présenté dans ce chapitre, un premier exemple d'application de taille industrielle. Cet exemple a permis de mettre en évidence l'intérêt de l'automatisation de la démarche de structuration tant sur le plan du gain de temps apporté que sur la fiabilité et la modularité du modèle généré.

Le second exemple présenté a fait apparaître la nécessité de décomposer les installations importantes en sous-systèmes. Cette limitation n'incombe pas aux différents logiciels de notre méthodologie, mais simplement au fait qu'il est difficile d'appréhender, dans leur globalité, des systèmes de taille importante.

CONCLUSION GENERALE

Nous avons présenté dans ce mémoire une méthodologie de conception assistée et plus particulièrement un outil de structuration automatique de la partie commande des systèmes de production flexible.

Cet outil de structuration est basé sur l'utilisation des Réseaux de Petri Structurés Adaptatifs et Colorés comme modèle de description de la partie commande.

Ce modèle original permet :

i) de préciser l'architecture parallèle de l'ensemble des systèmes de commande (**structuration**),

ii) de prendre en compte les différentes classes d'objet au moyen de marques de couleur (**coloration**),

iii) d'assurer une représentation non interprétée des paramétrages en mode automatique (**adaptativité**).

Afin de préserver la flexibilité maximale du système de production que l'on désire concevoir, nous avons proposé une approche originale qui permet de définir un modèle de base à degré de parallélisme optimal. Notre souci a été de préciser les modalités de définition automatique du graphe structuré, à partir d'un prégraphe d'ordonnancement des tâches ne spécifiant, à ce niveau, ni les modes de transport, ni la taille des stocks et ne pouvant de ce fait mettre en évidence les conflits possibles.

Il s'agit donc beaucoup plus d'une conception assistée au sens de l'analyse descendante que de la simple transformation d'un réseau de Petri en un réseau de Petri **structuré** équivalent. C'est dans ce sens que nous avons proposé des primitives de structuration interprétant, en fait, les schémas relativement peu nombreux qui peuvent apparaître sur un tel prégraphe dans le domaine de la production manufacturière.

Le RdPSAC constitue une dernière étape particulièrement bien adaptée à une implantation répartie des tâches parallèles mises en évidence, ainsi que les modes de communication qu'elles utilisent.

Dans ce sens, il paraît intéressant d'envisager le développement d'outils graphiques augmentant la convivialité de la méthodologie tant sur le plan de la conception du graphe de commande que sur le plan de sa simulation.

A un autre niveau, nous pensons qu'il serait intéressant de créer des interfaces avec le système de gestion de base de données de RdP développé par A. KARFIA /KAR 87/ afin de pouvoir enrichir notre logiciel par des primitives de structuration non encore étudiées.

Ces deux derniers points constituent des axes de recherches sur lesquels nous avons engagé des développements prospectifs.

BIBLIOGRAPHIE

INTRODUCTION GENERALE

/DEF 85/ DEFAUX M.,LOREAL A.
"Atelier flexible : commencez petit"
L'Usine Nouvelle, pp 47-59, Avril 1985.

CHAPITRE I

/ATA 87/ ATABAKHCHE H., SIMONETTI-BARBALHO D., VALETTE R., COURVOISIER M.
"Commande d'ateliers : un compromis est-il possible entre une approche graphique et une approche intelligence artificielle"
APII, Vol 21, n° 4, pp 377-394, 1987.

/SAH 87/ SAHRAOUI A.
"Contribution à la surveillance et à la commande d'atelier"
Thèse de Doct. de lUniversité Paul SABATIER, TOULOUSE , 21 Octobre 1987

MODELISATION DE LA PARTIE COMMANDE

/COR 79/ CORBEEL D.
"Schéma de cablage et schéma de contrôle. Application à la simulation et à la gestion des processus industriels"
Thèse de Doct. de Spécialité, LILLE, 1979.

/COR 80/ CORBEEL D.
"Formal description of processes systems and exception handling"
Mini & Micro, Proc. pp 335-339, BUDAPEST, 9-11 Septembre 1980.

/KAR 87/ KARFIA A.
"Présentation d'une base de connaissances adaptée à la modélisation par réseaux de Petri structurés du contrôle des processus de production discrétisés"
Thèse de Doct. de l'Université, LILLE, 9 Juillet 1987.

/COR 85/ CORBEEL D., GENTINA J.C., VERCAUTER C.
"Application of an extension of Petri nets to modelization of control and production processes"
6th European workshop on application and theory of Petri nets, Proc pp 53-74, ESPOO (FINLANDE), 26-28 Juin 1985.

/COR 81/ CORBEEL D., GENTINA J.C., VERCAUTER C.
"Méthodologie de description des systèmes de processus et de gestion d'erreurs"
IASTED Modelling, Identification and control, DAVOS, 1981.

/BRA 83/ G.W. BRAMS
"Réseaux de Petri : théorie et pratique"
Tome 1: théorie et analyse, Ed. MASSON, 1983.

/CAS 87/ CASTELAIN E.
"Modélisation et simulation interactive de cellules de production flexibles dans l'industrie manufacturière"
Thèse de Doct. de l'Université, LILLE, 26 Février 1987.

/HAC 75a/ HACK M.
"Petri-nets languages"
MIT, Computation Structures Group Memo 124, Project MAC, 1975.

/HAC 75b/ HACK M.
"Decision problem for Petri nets and vector addition systems"
Mass.:MIT, Project Mac, TR 59, Cambridge, Mars 1975

/VAL 78/ VALK R.
"Self modifying nets, a natural extension of Petri-nets"
Lect. notes in computer science, vol 62, pp 464-476, Springer,Verlag BERLIN, 1978.

/COR 84/ CORBEEL D., GENTINA J.C., VERCAUTER C.
"Adaptive Petri-nets for real-time applications"
IMACS Digiteh'84, PATRAS (GRECE), 1984.

/GEN 79/ GENRICH H., LAUTENBACH K., THIAGARAJAN P.
"Elements of general net theory"
Proc of the advnaced course on grneral net theory of processes and systems, HAMBURG, 1979.

/JEN 81/ JENSEN K.
"Coloured Petri-nets and invariant method"
Theorical computer science, n° 14, pp 317-336, North Holland Pub. Comp., 1981.

/PET 80/ PETERSON J.L.
"A note on coloured Petri-nets"
Information processing letters, Vol.11, n° 1, pp 40-43, 1980.

MODELISATION DU NIVEAU HIERARCHIQUE

/FRO 84/ FROMENT B., LESAGE J.J.
"Productique : les techniques de l'usinage flexible"
Ed. DUNOD, Série Génie Mécanique, 1984.

/BEN 86/ BENASSY J., BENCHIMOL G., BLOCH G., FERRE A., PHILIP C., ROSTAN G., SAUVAGE L., VAYSSIERE P.
"L'usine integrée par ordinateur"
Ed. HERMES 1986.

/SAH 87/ SAHRAOUI A., ATABAKHCHE H., COURVOISIER M. VALETTE R.
"Joining Petri-nets and knowledge based systems for monitoring purposes"
IEEE, International conference on robotics and automation, Proc pp 1160-1165, RALEIGH (USA), 31 Mars - 3 Avril 1987.

/COR 85/ CORBEEL D., GENTINA J.C., VERCAUTER C.
"Application of an extension of Petri nets to modelization of control and production processes"
6th European workshop on application and theory of Petri nets, Proc pp 53-74, ESPOO (FINLANDE), 26-28 Juin 1985.

/BOU 86/ BOUREY J.P., CORBEEL D., CRAYE E., GENTINA J.C.
"Adaptive and coloured structured Petri-ntes for description,analisys and synthesis of hierachical control and reliability of flexible cells in manufacturing systems"
1rst European Workshop on Fault Diagnostics, Reliability and related Knowledge-based Approaches, ile de Rhodes, Septembre 1986.
Vol 1, pp 281-295, D. Reidel Publ. Comp. 1987.

/BOU 87/ BOUREY J.P., CORBEEL D., CRAYE E., GENTINA J.C.
"Utilisation des réseaux de Petri structurés adaptatis colorés dans l'analyse et la synthèse du contrôle hiérachisé de processus discontinus. Partie A : les modèles de description."
APII, Vol 21, n° 4, pp 343-362, 1987.

/MAR 87/ MARTINEZ J., MURO P., SILVA M.
"Modelling, validation and software implementation of production systems using high level Petri-nets"
IEEE, International conference on robotics and automation, Proc pp 1180-1185, RALEIGH (USA), 31 Mars - 3 Avril 1987.

/ATA 87/ ATABAKHCHE H., SIMONETTI-BARBALHO D., VALETTE R., COURVOISIER M.
"Commande d'ateliers : un compromis est-il possible entre une approche graphique et une approche intelligence artificielle"
APII, Vol 21, n° 4, pp 377-394, 1987.

/CAS 87/ CASTELAIN E.
"Modélisation et simulation interactive de cellules de production flexibles dans l'industrie manufacturière"
Thèse de Doct. de l'Université, LILLE, 26 Février 1987.

MODELISATION DU PROCEDE

/CAS 87/ CASTELAIN E.
"Modélisation et simulation interactive de cellules de production flexibles dans l'industrie manufacturière"
Thèse de Doct. de l'Université, LILLE, 26 Février 1987.

/BRA 83/ G.W. BRAMS
"Réseaux de Petri : théorie et pratique"
Tome 2: modélisation et applications, Ed. MASSON, 1983.

/CAR 83/ CARRIERE B., CAZALOT C., DUMAT J.M., GROSJEAN P.M., LEROY P., PRUNET F.
"Un système de C.A.O. pour la commande de processus reposant sur un standard"
IFIP 1983.

/DEF 86/ DEFRENNE J., TOULOTTE J.M., HACHEMANI R.
"Validation des logiciels de commande des systèmes industriels à évolutions simultanées"
Convention Automatique Productique, Proc pp 154-158, PARIS, Mai 1986.

/RAM 73/ RAMCHANDI C.
"Analysis of asynchronous concurrent systems by timed Petri nets"
Ph.D Thesis, M.I.T., Septembre 1973.

/MOA 76/ MOALLA M.
"L'approche fonctionnelle dans la vérification des systèmes informatiques. Proposition d'un ensemble de méthodologies"
Thèse de Doct. Ing., INPG-ENSIMAG, GRENOBLE, 1976.

/SIF 77/ SIFAKIS J.
"Use of Petri nets for perfomance evaluation"
Measuring, Modelling and Evaluating Computer Systems, North Holland Pub. Comp., pp 75-93, 1977.

/CHR 83/ CHRETIENNE P.
"Les réseaux de Petri temporisés"
Thèse d'Etat, Université Pierre & Marie CURIE, PARIS VI, 1983.

CHAPITRE II

INTRODUCTION

/MAR 87/ MARTINEZ J., MURO P., SILVA M.
"Modelling, validation and software implementation of production systems using high level Petri-nets"
IEEE, International conference on robotics and automation, Proc pp 1180-1185, RALEIGH (USA), 31 Mars - 3 Avril 1987.

/MOS 87/ MOSER J.M., STEPOURJINE R., VIAL J.
"Méthodologie de conception d'une cellule flexible d'usinage"
2eme Conférence Internationale INRIA, Systèmes de production, Proc. pp 551-567, PARIS 6-10 Avril 1987.

/COU 84/ COURVOISIER M., BIGOU J.M., VALETTE R., DESCLAUX C., BENZAKOUR K
"The SECOIA project"
EUROCON84, Proc. pp 1-4, BRIGHTON, Septembre 1984.

/PRU 87/ PRUNET F., STURLESE J.L., CAZALOT C., GINESTET E., PANAGET D., DECHENAUX G., LLORCA P.
"Méthodologie et implantation automatique de commande d'automatisme à l'aide de la chaîne PIASTRE"
APII, vol 21 n°4, pp 299-322, 1987.

CASPAIM

/BON 85/ BONETTO R.
"Les ateliers flexibles de production"
Ed. HERMES, 1985.

/BOU 84/ BOURJAULT A.
"Contribution à une approche méthodologique de l'assemblage automatisé : élaboration automatique des séquences opératoires"
Thèse d'Etat, Faculté des sciences et techniques de l'Université de Franche-Comté, BESANCON, 12 Novembre 1984.

/AUG 87/ AUGEZ P.
"Une méthodologie intégrée d'aide à la conception des systèmes de fabrication"
Thèse de Doct. Ing, I.D.N, 9 Juin 1987.

/BOU 87a/ BOURJAULT A., CHAPPE D., HENRIOUD J.M.
"Elaboration automatique de gammes d'assemblage à l'aide des réseaux de Petri"
APII, vol 21 n°4, pp 323-342, 1987.

/CAM 85/ CAMPAGNE J.P., PEYRON J., TEMANI M.
"Structuration des bases de données techniques autour de nomenclatures et gammes-mères en vue de l'élaboration automatique de gammes"
APII, vol 19 n°4, pp ,1985.

/BOU 87b/ BOURJAULT A., HENRIOUD J.M.
"Determination des sous-assemblage d'un produit à partir des séquences temporelles d'assemblage"
APII, vol 21 n°2, pp ,1987.

/KAP 87/ KAPUSTA M., GENTINA J.C.
"Introduction to a first step of the aided design of control system of flexible manufacturing cells"
COMPINT'87, MONTREAL, NOVEMBRE 1987.

/THO 85/ THOMESSE J.P.
"Les réseaux locaux industriels"
Ed. E.T.A., collection Novotique, STRASBOURG, 1985.

/BEA 87/ BEAUBOUCHER N.
"L'implantation répartie du graphe d'un système de commande sur des automates programmables reliés par réseau local"
D.E.A de Productique, L.A.I.I, I.D.N., 22 Juin 1987.

/CRU 87/ CRUETTE D.
"L'implantation répartie du graphe d'un système de commande sur des automates programmables reliés par réseau local, et implantation d'un niveau hiérarchique"
D.E.A de Productique, L.A.I.I, I.D.N., 7 Septembre 1987.

LE PREGRAPHE

/GEN 87/ GENTINA J.C., CORBEEL D.
"Coloured adaptive structured Petri-nets : a tool for the automatic synthesis of hierachical control of flexible manufacturing systems (F.M.S.)"
IEEE, International conference on robotics and automation, Proc pp 1166-1173, RALEIGH (USA), 31 Mars - 3 Avril 1987.

/BOU 87/ BOUREY J.P., GENTINA J.C.
"Computer aided design for structuration and representation of control of flexible manufacturing systems"
8th European Workshop on Application and Theory of Petri-nets, Proc pp 117-135 , SARAGOSSE, 24-26 Juin 1987.

/KAP 87/ KAPUSTA M., GENTINA J.C.
"Introduction to a first step of the aided design of control system of flexible manufacturing cells"
COMPINT'87, MONTREAL, NOVEMBRE 1987.

LE SIMULATEUR

/CAS 87a/ CASTELAIN E.
"Modélisation et simulation interactive de cellules de production flexibles dans l'industrie manufacturière"
Thèse de Doct. de l'Université, LILLE, 26 Février 1987.

/MEM 80/ MEMMI G., ROUCAIROL G.
"Linear algebra in net theory"
Proc. of the advanced course on general Net Theory of Processes and Systems, HAMBURG, SPRINGER 1980.

/GIR 84/ GIROD C.
"La conception et la réalisation d'un logiciel de simulation de réseaux de Petri"
Thèse de Doct. Ing., I.D.N., 25 Septembre 1984.

/CAS 85/ CASTELAIN E., CORBEEL D., GENTINA J.C.
"Comparative simulations of control processes described by Petri-nets"
COMPINT'85, MONTREAL, Septembre 1985.

/CAS 87b/ CASTELAIN E., GENTINA J.C.
"Description of manufacturing processes by means of a type frame language"
IMACS, Symposium on A.I, expert systems and languages in modelling and simulation, BARCELONE, Juin 1987.

/BEL 85/ BEL G., DUBOIS D.
"Modélisation et simulation de systèmes automatisés de production"
APII, vol 19 n°1, pp 3-43, 1985.

CHAPITRE III

/CAS 87/ CASTELAIN E.
"Modélisation et simulation interactive de cellules de production flexibles dans l'industrie manufacturière"
Thèse de Doct. de l'Université, LILLE, 26 Février 1987.

/BAL 87/ BALBO G., CHIOLA G. , FRANCCESCHINIS G., MOLINAR ROET G.
"Generalized stochastic Petri nets for the performance evaluation of F.M.S."
IEEE, International conference on robotics and automation, Proc pp 1013-1018, RALEIGH (USA), 31 Mars - 3 Avril 1987.

/DES 85/ DESCOTES-GENON B., LADET P.
"Outils graphiques pour la modélisation et la simulation d'applications de commande séquentielle"
Congrès AFCET , Proc pp 559-570, TOULOUSE, Octobre 1985.

/MAR 87/ MARTINEZ J., MURO P., SILVA M.
"Modelling, validation and software implementation of production systems using high level Petri-nets"
IEEE, International conference on robotics and automation, Proc pp 1180-1185, RALEIGH (USA), 31 Mars - 3 Avril 1987.

CHAPITRE IV

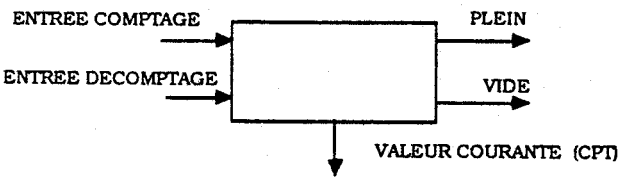
/MAY 87/ MAYET J.
"Sur la conception d'un atelier flexible de fabrication mécanique"
Mémoire Ing. C.N.A.M., Centre Régional Associé de LILLE, 18 Juin 1987.

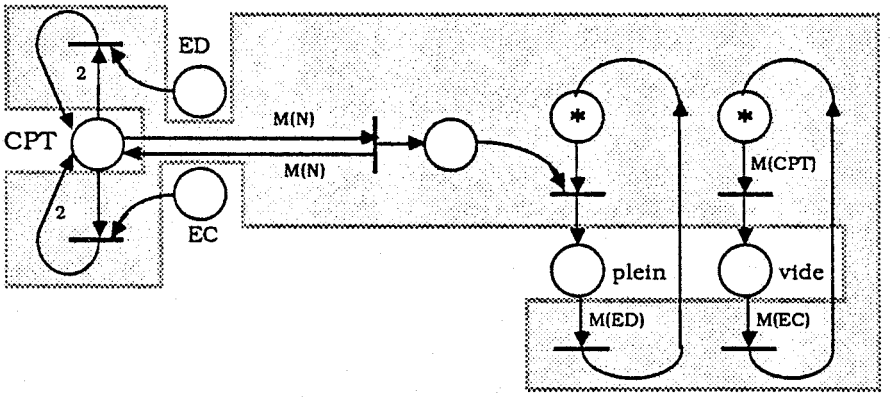
CONCLUSION GENERALE

/KAR 87/ KARFIA A.
"Présentation d'une base de connaissances adaptée à la modélisation par réseaux de Petri structurés du contrôle des processus de production discrétisés"
Thèse de Doct. de l'Université, LILLE, 9 Juillet 1987.

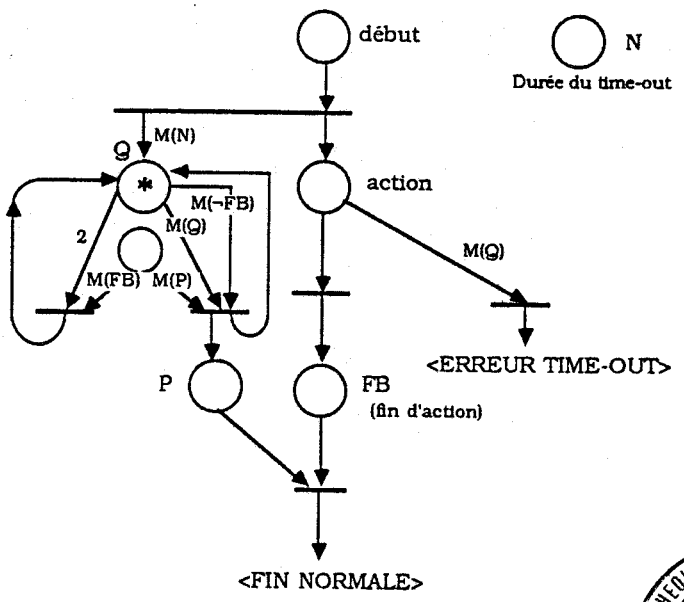
ANNEXE

COMPTEUR/DECOMPTEUR

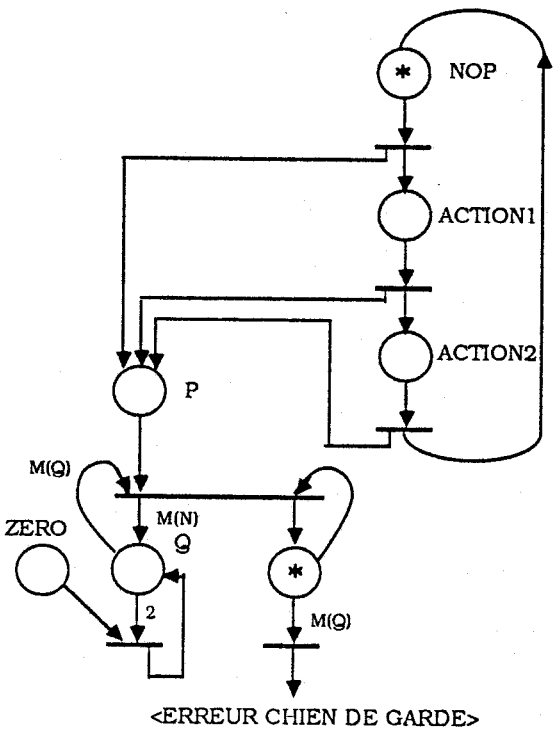




TIME -OUT SUR UNE ACTION



CHIEN DE GARDE SUR UN PROCESSUS



RESUME

Nous présentons dans ce mémoire, un outil de structuration qui permet de construire de manière rigoureuse et systématique la partie procédurale du système de commande de systèmes flexibles de production discontinue.

Cet outil permet de générer un graphe de commande (Réseau de Petri Structuré et Coloré) sur lequel le parallélisme et la concurrence du système sont résolus et mis en évidence. Notre objectif a été d'offrir la plus grande flexibilité pour la création d'un tel graphe. Dans ce sens nous proposons un ensemble de primitives de structuration basées sur une analyse des différentes fonctionnalités d'un modèle initial (appelé "pregraphe") du système de production.

L'automatisation de la démarche de structuration apporte un gain de temps non négligeable dans la phase de conception et une meilleure fiabilité due à la nécessaire cohérence logique du modèle produit avec le prégraphe.

Un exemple de dimension industrielle illustre l'approche proposée.

MOTS-CLEFS :

METHODOLOGIE DE CONCEPTION
SYSTEMES DE PRODUCTION FLEXIBLE
MODELISATION
PARTIE COMMANDE
RESEAUX DE PETRI
STRUCTURATION
LANGAGE LE_LISP