Spécification et conception sûre d'automatismes discrets complexes, basées sur l'utilisation du GRAFCET et des réseaux de PETRI

Mohamed Moalla

THESE

présentée à

l' Université Scientifique et Médicale de Grenoble

et à

l' Institut National Polytechnique de Grenoble

pour obtenir le grade de

DOCTEUR D'ETAT ES SCIENCES
«Informatique - Automatique»

par

Mohamed MOALLA

SPECIFICATION ET CONCEPTION SURE D'AUTOMATISMES DISCRETS COMPLEXES, BASEES SUR L'UTILISATION DU GRAFCET ET DES RESEAUX DE PETRI.

Thèse soutenue le 10 Juillet 1981 devant la commission d'examen.

	L. BOLLIET	Président
Mme	G. SAUCIER	Examinateurs
MM.	A. COSTES	
	R. DAVID	
	G. GIRAULT	
	M. SAKAROVITCH	
	M. DEGUERRY	Invités
	J.F. LEMAITRE	

UNIVERSITE SCIENTIFIQUE ET MEDICALE DE GRENOBLE

Monsieur Gabriel CAU : Président
Monsieur Joseph KLEIN : Vice-Président

MEMBRES DU CORPS ENSEIGNANT DE L'U.S.M.G.

PROFESSEURS TITULAIRES

MM.	AMBLARD Pierre	Clinique de dermatologie
	ARNAUD Paul	Chimie
	ARVIEU Robert	I.S.N.
	AUBERT Guy	Physique
	AYANT Yves	Physique approfondie
Mme	BARBIER Marie-Jeanne	Electrochimie
MM.	BARBIER Jean-Claude	Physique expérimentale
	BARBIER Reynold	Géologie appliquée
	BARJON Robert	Physique nucléaire
	BARNOUD Fernand	Biosynthèse de la cellulose
	BARRA Jean-René	Statistiques
	BARRIE Joseph	Clinique chirurgicale A
	BEAUDOING André	Clinique de pédiatrie et puériculture
	BELORIZKY Elie	Physique
	BARNARD Alain	Mathématiques pures
Mme	BERTRANDIAS Françoise	Mathématiques pures
MM.	BERTRANDIAS Jean-Paul	Mathématiques pures
	BEZES Henri	Clinique chirurgicale et traumatologie
	BLAMBERT Maurice	Mathématiques pures
	BOLLIET Louis	Informatique (I.U.T. B)
	BONNET Jean-Louis	Clinique ophtalmologie
	BONNET-EYMARD Joseph	Clinique hépato-gastro-entérologie
Mme	BONNIER Marie-Jeanne	Chimie générale
MM.	BOUCHERLE André	Chimie et toxicologie
	BOUCHEZ Robert	Physique nucléaire
	BOUSSARD Jean-Claude	Mathématiques appliquées
	BOUTET DE MONVEL Louis	Mathématiques pures
	BRAVARD Yves	Géographie
	CABANEL Guy	Clinique rhumatologique et hydrologique
	CALAS François	Anatomie
	CARLIER Georges	Biologie végétale
	CARRAZ Gilbert	Biologie animale et pharmacodynamie

.../...

MM.	CAU Gabriel	Médecine légale et toxicologie
	CAUQUIS Georges	Chimie organique
	CHABAUTY Claude	Mathématiques pures
	CHARACHON Robert	Clinique ot-rhino-laryngologique
	CHATEAU Robert	Clinique de neurologie
	CHIBON Pierre	Biologie animale
	COEUR André	Pharmacie chimique et chimie analytique
	COUDERC Pierre	Anatomie pathologique
	DEBELMAS Jacques	Géologie générale
	DEGRANGE Charles	Zoologie
	DELORMAS Pierre	Pneumophtisiologie
	DEPORTES Charles	Chimie minérale
	DESRE Pierre	Métallurgie
	DODU Jacques	Mécanique appliquée (I.U.T. I)
	DOLIQUE Jean-Michel	Physique des plasmas
	DREYFUS Bernard	Thermodynamique
	DUCROS Pierre	Cristallographie
	FONTAINE Jean-Marc	Mathématiques pures
	GAGNAIRE Didier	Chimie physique
	GALVANI Octave	Mathématiques pures
	GASTINEL Noël	Analyse numérique
	GAVEND Michel	Pharmacologie
	GEINDRE Michel	Electroradiologie
	GERBER Robert	Mathématiques pures
	GERMAIN Jean-Pierre	Mécanique
	GIRAUD Pierre	Géologie
	JANIN Bernard	Géographie
	KAHANE André	Physique générale
	KLEIN Joseph	Mathématiques pures
	KOSZUL Jean-Louis	Mathématiques pures
	KRAVTCHENKO Julien	Mécanique
	LACAZE Albert	Thermodynamique
	LACHARME Jean	Biologie végétale
Mme	LAJZEROWICZ Janine	Physique
MM.	LAJZEROWICZ Joseph	Physique
	LATREILLE René	Chirurgie générale
	LATURAZE Jean	Biochimie pharmaceutique
	LAURENT Pierre	Mathématiques appliquées
	LEDRU Jean	Clinique médicale B
	LE ROY Philippe	Mécanique (.I.U.T. I)

.../...

MM.	LLIBOUTRY Louis	Géophysique
	LOISEAUX Jean-Marie	Sciences nucléaires
	LONGEQUEUE Jean-Pierre	Physique nucléaire
	LOUP Jean	Géographie
Mlle	LUTZ Elisabeth	Mathématiques pures
MM.	MALINAS Yves	Clinique obstétricale
	MARTIN-NOEL Pierre	Clinique cardiologique
	MAYNARD Roger	Physique du solide
	MAZARE Yves	Clinique Médicale A
	MICHEL Robert	Minéralogie et pétrographie
	MICOUD Max	Clinique maladies infectieuses
	MOURIQUAND Claude	Histologie
	MOUSSA André	Chimie nucléaire
	NEGRE Robert	Mécanique
	NOZIERES Philippe	Spectrométrie physique
	OZENDA Paul	Botanique
	PAYAN Jean-Jacques	Mathématiques pures
	PEBAY-PEYROULA Jean-Claude	Physique
	PERRET Jean	Séméiologie médicale (neurologie)
	RASSAT André	Chimie systématique
	RENARD Michel	Thermodynamique
	REVOL Michel	Urologie
	RINALDI Renaud	Physique
	DE ROUGEMONT Jacques	Neuro-Chirurgie
	SARRAZIN Roger	Clinique chirurgicale B
	SEIGNEURIN Raymond	Microbiologie et hygiène
	SENGEL Philippe	Zoologie
	SIBILLE Robert	Construction mécanique (I.U.T. I)
	SOUTIF Michel	Physique générale
	TANCHE Maurice	Physiologie
	VAILLANT François	Zoologie
	VALENTIN Jacques	Physique nucléaire
Mme	VERAIN Alice	Pharmacie galénique
MM.	VERAIN André	Physique biophysique
	VEYRET Paul	Géographie
	VIGNAIS Pierre	Biochimie médicale

.../...

PROFESSEURS ASSOCIES

MM.	CRABBE Pierre	CERMO
	SUNIER Jules	Physique

PROFESSEURS SANS CHAIRE

Mlle	AGNIUS-DELORS Claudine	Physique pharmaceutique
	ALARY Josette	Chimie analytique
MM.	AMBROISE-THOMAS Pierre	Parasitologie
	ARMAND Gilbert	Géographie
	BENZAKEN Claude	Mathématiques appliquées
	BIAREZ Jean-Pierre	Mécanique
	BILLET Jean	Géographie
	BOUCHET Yves	Anatomie
	BRUGEL Lucien	Energétique (I.U.T. I)
	BUISSON René	Physique (I.U.T. I)
	BUTEL Jean	Orthopédie
	COHEN-ADDAD Jean-Pierre	Spectrométrie physique
	COLOMB Maurice	Biochimie médicale
	CONTE René	Physique (I.U.T. I)
	DELOBEL Claude	M.I.A.G.
	DEPASSEL Roger	Mécanique des fluides
	GAUTRON René	Chimie
	GIDON Paul	Géologie et minéralogie
	GLENAT René	Chimie organique
	GROULADE Joseph	Biochimie médicale
	HACQUES Gérard	Calcul numérique
	HOLLARD Daniel	Hématologie
	HUGONOT Robert	Hygiène et médecine préventive
	IDELMAN Simon	Physiologie animale
	JOLY Jean-René	Mathématiques pures
	JULLIEN Pierre	Mathématiques appliquées
Mme	KAHANE Josette	Physique
MM.	KRAKOWIACK Sacha	Mathématiques appliquées
	KUHN Gérard	Physique (I.U.T. I)
	LUU DUC Cuong	Chimie organique - pharmacie
	MICHOULIER Jean	Physique (I.U.T. I)
Mme	MINIER Colette	Physique (I.U.T. I)

.../...

MM.	PELMONT Jean	Biochimie
	PERRIAUX Jean-Jacques	Géologie et minéralogie
	PFISTER Jean-Claude	Physique du solide
Mlle	PIERY Yvette	Physiologie animale
MM.	RAYNAUD Hervé	M.I.A.G.
	REBECQ Jacques	Biologie (CUS)
	REYMOND Jean-Charles	Chirurgie générale
	RICHARD Lucien	Biologie végétale
Mme	RINAUDO Marguerite	Chimie macromoléculaire
MM.	SARROT-REYNAULD Jean	Géologie
	SIROT Louis	Chirurgie générale
Mme	SOUTIF Jeanne	Physique générale
MM.	STIEGLITZ Paul	Anesthésiologie
	VIALON Pierre	Géologie
	VAN CUTSEM Bernard	Mathématiques appliquées

MAITRES DE CONFERENCES ET MAITRES DE CONFERENCES AGREGES

MM.	ARMAND Yves	Chimie (I.U.T. I)
	BACHELOT Yvan	Endocrinologie
	BARGE Michel	Neuro-chirurgie
	BEGUIN Claude	Chimie organique
Mme	BERIEL Hélène	Pharmacodynamie
MM.	BOST Michel	Pédiatrie
	BOUCHARLAT Jacques	Psychiatrie adultes
Mme	BOUCHE Liane	Mathématiques (CUS)
MM.	BRODEAU François	Mathématiques (I.U.T. B) (Personne étrangère habilitée à être directeur de thèse)
	BERNARD Pierre	Gynécologie
	CHAMBAZ Edmond	Biochimie médicale
	CHAMPETIER Jean	Anatomie et organogénèse
	CHARDON Michel	Géographie
	CHERADAME Hervé	Chimie papetière
	CHIAVERINA Jean	Biologie appliquée (EFP)
	COLIN DE VERDIERE Yves	Mathématiques pures
	CONTAMIN Charles	Chirurgie thoracique et cardio-vasculaire
	CORDONNER Daniel	Néphrologie
	COULOMB Max	Radiologie
	CROUZET Guy	Radiologie

.../...

MM.	CYROT Michel	Physique du solide
	DENIS Bernard	Cardiologie
	DOUCE Roland	Physiologie végétale
	DUSSAUD René	Mathématiques (CUS)
Mme	ETERRADOSSI Jacqueline	Physiologie
MM.	FAURE Jacques	Médecine légale
	FAURE Gilbert	Urologie
	GAUTIER Robert	Chirurgie générale
	GIDON Maurice	Géologie
	GROS Yves	Physique (I.U.T. I)
	GUIGNIER Michel	Thérapeutique
	GUITTON Jacques	Chimie
	HICTER Pierre	Chimie
	JALBERT Pierre	Histologie
	JUNIEN-LAVILLAVROY Claude	O.R.L.
	KOLODIE Lucien	Hématologie
	LE NOC Pierre	Bactériologie-virologie
	MACHE Régis	Physiologie végétale
	MAGNIN Robert	Hygiène et médecine préventive
	MALLION Jean-Michel	Médecine du travail
	MARECHAL Jean	Mécanique (I.U.T. I)
	MARTIN-BOUYER Michel	Chimie (CUS)
	MASSOT Christian	Médecine interne
	NEMOZ Alain	Thermodynamique
	NOUGARET Marcel	Automatique (I.U.T. I)
	PARAMELLE Bernard	Pneumologie
	PECCOUD François	Analyse (I.U.T. B) (Personnalité étrangère habilitée à être directeur de thèse)
	PEFFEN René	Métallurgie (I.U.T. I)
	PERRIER Guy	Géophysique-glaciologie
	PHELIP Xavier	Rhumatologie
	RACHALL Michel	Médecine interne
	RACINET Claude	Gynécologie et obstétrique
	RAMBAUD Pierre	Pédiatrie
	RAPHAEL Bernard	Stomatologie
Mme	RENAUDET Jacqueline	Bactériologie (pharmacie)
MM.	ROBERT Jean-Bernard	Chimie-physique
	ROMIER Guy	Mathématiques (I.U.T. B) (Personnalité étrangère habilitée à être directeur de thèse)
	SAKAROVITCH Michel	Mathématiques appliquées

.../...

MM.	SCHAERER René	Cancérologie
Mme	SEIGLE-MURANDI Françoise	Crytogamie
MM.	STOEBNER Pierre	Anatomie pathologie
	STUTZ Pierre	Mécanique
	VROUSOS Constantin	Radiologie

MAITRES DE CONFERENCES ASSOCIES

MM.	DEVINE Roderick	Spectro Physique
	KANEKO Akira	Mathématiques pures
	JOHNSON Thomas	Mathématiques appliquées
	RAY Tuhina	Physique

MAITRE DE CONFERENCES DELEGUE

M.	ROCHAT Jacques	Hygiène et hydrologie (pharmacie)

Fait à Saint Martin d'Hères, novembre 1977

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Année universitaire 1979-1980

Président : M. Philippe TRAYNARD

Vice-Présidents : M. Georges LESPINARD
M. René PAUTHENET

PROFESSEURS DES UNIVERSITES

MM.	ANCEAU François	Informatique fondamentale et appliquée
	BENOIT Jean	Radioéléctricité
	BESSON Jean	Chimie Minérale
	BLIMAN Samuel	Electronique
	BLOCH Daniel	Physique du Solide - Cristallographie
	BOIS Philippe	Mécanique
	BONNETAIN Lucien	Génie Chimique
	BONNIER Etienne	Métallurgie
	BOUVARD Maurice	Génie Mécanique
	BRISSONNEAU Pierre	Physique des Matériaux
	BUYLE-BODIN Maurice	Electronique
	CHARTIER Germain	Electronique
	CHERADAME Hervé	Chimie Physique Macromoléculaires
Mme	CHERUY Arlette	Automatique
MM.	CHIAVERINA Jean	Biologie, Biochimie, Agronomie
	COHEN Joseph	Electronique
	COUMES André	Electronique
	DURAND Francis	Métallurgie
	DURAND Jean-Louis	Physique Nucléaire et Corpusculaire
	FELICI Noël	Electrotechnique
	FOULARD Claude	Automatique
	GUYOT Pierre	Métallurgie Physique
	IVANES Marcel	Electrotechnique
	JOUBERT Jean-Claude	Physique du Solide - Cristallographie
	LACOUME Jean-Louis	Géographie - Traitement du Signal
	LANCIA Roland	Electronique - Automatique
	LESIEUR Marcel	Mécanique
	LESPINARD Georges	Mécanique
	LONGEQUEUE Jean-Pierre	Physique Nucléaire Corpusculaire
	MOREAU René	Mécanique
	MORET Roger	Physique Nucléaire Corpusculaire
	PARIAUD Jean-Charles	Chimie - Physique
	PAUTHENET René	Physique du Solide - Cristallographie
	PERRET René	Automatique

.../...

MM.	PERRET Robert	Electrotechnique
	PIAU Jean-Michel	Mécanique
	PIERRARD Jean-Marie	Mécanique
	POLOUJADOFF Michel	Electrotechnique
	POUPOT Christian	Electronique - Automatique
	RAMEAU Jean-Jacques	Chimie
	ROBERT André	Chimie Appliquée et des matériaux
	ROBERT François	Analyse numérique
	SABONNADIERE Jean-Claude	Electrotechnique
Mme	SAUCIER Gabrielle	Informatique fondamentale et appliquée
M.	SOHM Jean-Claude	Chimie - Physique
Mme	SCHLENKER Claire	Physique du Solide - Cristallographie
MM.	TRAYNARD Philippe	Chimie - Physique
	VEILLON Gérard	Informatique fondamentale et appliquée
	ZADWORNY François	Electronique

CHERCHEURS DU C.N.R.S. (Directeur et Maître de Recherche)

M.	FRUCHART Robert	Directeur de Recherche
MM.	ANSARA Ibrahïm	Maître de Recherche
	BRONOEL Guy	Maître de Recherche
	CARRE René	Maître de Recherche
	DAVID René	Maître de Recherche
	DRIOLE Jean	Maître de Recherche
	KAMARINOS Georges	Maître de Recherche
	KLEITZ Michel	Maître de Recherche
	LANDAU Ioan-Doré	Maître de Recherche
	MERMET Jean	Maître de Recherche
	MUNIER Jacques	Maître de Recherche

Personnalités habilitées à diriger des travaux de recherche (décision du Conseil Scientifique)

E.N.S.E.E.G.

MM.	ALLIBERT Michel	
	BERNARD Claude	
	CAILLET Marcel	
Mme	CHATILLON Catherine	
MM.	COULON Michel	
	HAMMOU Abdelkader	
	JOUD Jean-Charles	
	RAVAINE Denis	
	SAINFORT	C.E.N.G.

.../...

MM. SARRAZIN Pierre
SOUQUET Jean-Louis
TOUZAIN Philippe
URBAIN Georges — Laboratoire des Ultra-Réfractaires ODEILLO

E.N.S.M.E.E.

MM. BISCONDI Michel
BOOS Jean-Yves
GUILHOT Bernard
KOBILANSKI André
LALAUZE René
LANCELOT Françis
LE COZE Jean
LESBATS Pierre
SOUSTELLE Michel
THEVENOT François
THOMAS Gérard
TRAN MINH Canh
DRIVER Julian
RIEU Jean

E.N.S.E.R.G.

MM. BOREL Joseph
CHEHIKIAN Alain
VIKTOROVITCH Pierre

E.N.S.I.E.G.

MM. BORNARD Guy
DESCHIZEAUX Pierre
GLANGEAUD François
JAUSSAUD Pierre
Mme JOURDAIN Geneviève
MM. LEJEUNE Gérard
PERARD Jacques

E.N.S.H.G.

M. DELHAYE Jean-Marc

E.N.S.I.M.A.G.

MM. COURTIN Jacques
LATOMBE Jean-Claude
LUCAS Michel
VERDILLON André

Les travaux présentés dans ce mémoire ont été réalisés au sein du laboratoire d'Informatique et de Mathématiques Appliquées de Grenoble (Laboratoire Associé au CNRS, n° 7).

Je voudrais exprimer ma reconnaissance à tous ceux, du laboratoire ou de l'extérieur, qui m'ont aidé par leurs conseils, par leurs compétences et aussi par leur amitié.

Je tiens à exprimer toute ma gratitude à Monsieur L. BOLLIET, Professeur à l'Université des Sciences Sociales de Grenoble, qui a guidé mes premiers pas de chercheur et n'a cessé depuis de m'accorder sa confiance et son soutien. Je suis sensible à l'honneur qu'il me fait aujourd'hui de présider mon jury de Thèse.

Ma gratitude va également à Madame SAUCIER, Professeur à l'Institut National Polytechnique de Grenoble, qui m'a accueilli dans son équipe et m'a prodigué aide et conseils tout au long de l'accomplissement de mes travaux.

Je remercie particulièrement :

Messieurs

A. COSTES, Professeur à l'Institut National Polytechnique de Toulouse, et

R. DAVID, Maître de Recherche CNRS au laboratoire d'Automatique de Grenoble,

qui ont bien voulu examiner en détail ces travaux et en être les rapporteurs.

Messieurs

M. DEGUERRY, Directeur de la Division Electronique et Automatisme à la TELEMECANIQUE

C. GIRAULT, Professeur à l'Institut de programmation de Paris VI

J.F. LEMAITRE, Directeur des Produits Nouveaux à la régie RENAULT-RIET

M. SAKAROVITCH, Professeur à l'Université Scientifique et Médicale de Grenoble

qui ont accepté de participer au jury.

Je dois beaucoup à mes amis P. CASPI, N. HALBWACHS, J. SIFAKIS, et J. PULOU dont la collaboration, les critiques et les suggestions m'ont été d'une aide précieuse. Qu'ils soient assurés de ma reconnaissance et de ma sympathie.

Je voudrais mentionner aussi les membres de l'équipe de F. DOLLE – de la Direction des Affaires Scientifiques et Techniques de la régie RENAULT, et ceux de l'équipe de B. SEMPE – de la Société OPTION, avec qui j'ai travaillé dans le cadre de contacts de recherche. Je me suis beaucoup enrichi de cette collaboration. J'ai notamment apprécié les compétences et l'ouverture avec lesquelles ces équipes abordaient la concrétisation de la recherche dans l'industrie.

Je suis très reconnaissant envers mon amie F. VUILLET pour avoir accepté de combler les carences d'un secrétariat constamment "en surcharge" et assuré avec autant de dévouement et de talent la dactylographie de ce mémoire.

Je ne puis enfin oublier de remercier l'équipe de D. IGLESIAS qui a assuré la reprographie de ce document.

- SOMMAIRE -

INTRODUCTION.

PARTIE I : APPROCHE ET OUTILS POUR LA SPÉCIFICATION DU CAHIER DES CHARGES D'UN AUTOMATISME COMPLEXE.

Chap.I : STRUCTURATION ET HIERARCHISATION DU CAHIER DES CHARGES

Chap.II : OUTILS DE REPRESENTATION DES SPECIFICATIONS FONCTIONNELLES

Chap.III : EXTENSION DU GRAFCET POUR LA SPECIFICATION DE SYSTEMES TEMPS REEL COMPLEXES

Annexe : EXEMPLE D'APPLICATION A LA SPECIFICATION DU CAHIER DES CHARGES DU SYSTEME DE COMMANDE D'UN PROCEDE DE DISTILLATION

Bibliographie.

PARTIE II : LES RÉSEAUX DE PETRI ET LEUR UTILISATION POUR LA DESCRIPTION ET L'ANALYSE DES SYSTÈMES À EVOLUTIONS PARALLÈLES.

Chap.IV : RESEAUX DE PETRI AUTONOMES

Chap.V : RESEAUX DE PETRI NON-AUTONOMES

Chap.VI : ANALYSE ET VERIFICATION DES SYSTEMES DECRITS PAR DES RdP

Chap.VII : GRAFCET ET RdPI

Bibliographie.

PARTIE III : MISE EN OEUVRE ASSISTÉE PAR ORDINATEUR.

Chap.VIII : LE SYSTEME M.A.S

Bibliographie.

CONCLUSION.

INDEX.

BIBLIOGRAPHIE.

INTRODUCTION

L'automatisation des fonctions tant dans le domaine de la production que dans celui de la gestion et du contrôle connaît un développement important depuis le début des années 70. D'une part, cette automatisation est considérée comme un point de passage obligé dans la recherche d'une meilleure maîtrise des "moyens de production" et d'une compétitivité maximale. D'autre part, elle est stimulée par les progrès de la technologie micro-électronique offrant sur le marché des composants de plus en plus puissants et fiables (automates programmables, circuits à haut degré d'intégration, microprocesseurs, ...) à des bas prix. Il est devenu possible dès lors de réaliser à faible coût et de façon très souple des systèmes d'automatismes sophistiqués qui n'étaient pas envisageables du temps des réalisations en composants analogiques ou de logique à moyenne intégration.

Cette évolution de la complexité s'est aussi accompagnée d'une évolution des exigences quant à la qualité de service que ces automatismes doivent offrir ; en particulier pour ce qui concerne leur sûreté de fonctionnement. On attache de plus en plus d'importance aux facilités de commande et de contrôle que l'on peut trouver dans l'interface d'exploitation de l'automatisme . En même temps, on cherche à accroître le pouvoir de décision et d'action de l'automatisme. Celui-ci se substitue ainsi en grande partie à l'homme, et son rôle évolue de la simple "aide à la commande" vers une "commande complète à pleine autorité" qui doit posséder les qualités minima de fiabilité, de disponibilité et de sécurité.

La conjugaison de ces trois facteurs : élargissement des champs d'application de l'automatisation, évolution de la complexité et des exigences, et mutation profonde des technologies de réalisation a conduit à une remise en cause quasi-totale des approches de conception existantes, basées pour l'essentiel sur l'intuition et l'expérience. Et il s'est créé un besoin pressant pour le développement de méthodes, de solutions et d'outils :

- Des méthodes pour décomposer progressivement des problèmes complexes en sous-problèmes plus simples, selon un processus qui permet d'assurer une conception aussi "sûre" que possible.

. Des catalogues de solutions qui aident à résoudre ces problèmes compte tenu des nouvelles exigences et des spécificités des nouvelles technologies.

. Des outils CAO pour assister le concepteur dans ce processus de conception de façon à réaliser un produit optimisé, tout en minimisant ses coûts et durée de réalisation.

Pour ce qui concerne particulièrement les automatismes discrets, deux aspects essentiels de ce besoin sont apparus comme étant prioritaires : 1) La définition et la normalisation des outils de spécification du cahier des charges et 2) Le développement d'une gamme d'outils compatibles pour l'aide à la conception et à la synthèse.

1 - Nécessité d'une normalisation des outils de spécification du cahier des charges :

Il dépend étroitement de la clarté et de la rigueur apportées à la spécification des objectifs de fonctionnement et de sécurité attendus d'un automatisme que la conception puisse trouver les meilleures solutions à la réalisation de ces objectifs. Or, force est de constater que les outils de spécification utilisés (le jargon entre autre) varient presque avec chaque client, sont souvent imprécis, mal définis, et sujets à de nombreuses équivoques qui ne se révèlent qu'à des stades avancés de la réalisation ou lors de la recette. Il en résulte alors une série de retouches et de sur-retouches qui, généralement, présentent de grands risques d'introduction d'erreurs, sont coûteuses et détruisent tout effort d'optimisation.

Cette situation se complique avec les nouveaux automatismes qui, en sus de la complexité, amènent à prendre en compte de façon beaucoup plus significative des problèmes de parallélisme et de hiérarchisation dans la commande, ainsi que des considérations subtiles concernant le comportement temps réel et la sûreté de fonctionnement qui sont difficiles à exprimer en l'absence d'un langage technique approprié.

L'intérêt de rechercher une méthodologie et des outils de description des cahiers des charges est alors multiple :

- Il s'agit tout d'abord de dégager les concepts et la termino-

logie qui permettent de comprendre et de maîtriser ces différents problèmes. La réunion de ces concepts dans des outils de description adéquats facilite la tâche du spécificateur qui pourraît disposer ainsi des moyens lui permettant d'exprimer le plus directement et le plus précisément possible ses désidérata, tout en l'obligeant, par la rigueur de ces outils, à approfondir l'analyse de ces désidérata et à faire une autocritique permanente vis-à-vis des impératifs de complétude, de clarté et de cohérence des spécifications fournies.

- L'unification et la normalisation de ces outils permet de clarifier la communication et le dialogue entre les différentes personnes "clients" et "concepteurs" concernées par la conduite du projet d'automatisation. La responsabilité et les niveaux de décision de chacun se trouvent ainsi clairement établis et la conception a de meilleures chances d'être initialisée sur des bases solides.

- Les retombées sont à considérer au niveau même du développement des outils d'aide à la conception et à la synthèse.

La généralité de ces outils (de façon à ne dépendre ni d'un type particulier d'application ni d'une technologie quelconque de réalisation) et les possibilités offertes pour l'aide à l'analyse doivent constituer deux critères primordiaux dans le choix de ces outils. Un troisième critère aussi important, qui apparaît par certains côtés en contradiction avec le premier, est la facilité d'utilisation : Il ne faut pas perdre de vue que cette normalisation s'adresse à des utilisateurs de tous les niveaux (clients non spécialistes, techniciens, ... chefs de projets). Comme il ne faut pas perdre de vue qu'il s'agit d'outils de travail. Il faut éviter de sombrer dans les notations trop formelles qui rebutent et qui rendent les spécifications illisibles ; un graphisme structuré est généralement plus agréable à manipuler. Il y a aussi tout intérêt à limiter le nombre des primitives en focalisant la normalisation sur les concepts essentiels et en laissant une ouverture vers des langages complémentaires qui viendront commenter et adapter l'utilisation de ces outils pour des applications particulières.

Il faut souligner, enfin, que cette normalisation ne peut être efficace que si elle est associée à une véritable méthodologie de présentation du cahier des charges qui sépare les différents aspects techniques qui interviennent dans la spécification d'un automatisme, et définit les critères qui permettent de structurer et de hiérarchiser leur exposé.

Nous reviendrons longuement sur ce problème de la spécification dans le développement de ce mémoire. Si quelques éléments de solution commencent à émerger pour ce qui concerne la spécification des automatismes logiques, la généralisation aux automatismes numériques et aux systèmes de commande temps réel de façon générale n'en est encore qu'à ses balbutiements.

2 - Développement d'une gamme d'outils compatibles pour l'aide à la conception et à la synthèse :

Le processus de conception d'un automatisme comporte un certain nombre d'étapes que l'on peut délimiter comme suit :

A - *Analyse préliminaire de la complétude du cahier des charges et de la faisabilité de l'automatisme* : Le cahier des charges définit les fonctions globales que doit réaliser l'automatisme et les diverses contraintes (ou désidérata) opérationnelles et technologiques auxquelles il doit satisfaire. Il est nécessaire, dans un premier temps, de comprendre ces spécifications, d'étudier les liens qui peuvent exister entre les fonctions globales, de cerner de très près les caractéristiques et besoins essentiels du produit demandé ... . Cette analyse doit permettre de mettre en évidence les éventuelles insuffisances et contradictions des spécifications (incohérence de fonctionnement ou de performances, insuffisance des données d'entrée pour la réalisation de fonctions particulières, indéterminisme, impossibilités ...).

B - *Décomposition fonctionnelle et définition de l'architecture fonctionnelle de l'automatisme* : C'est l'étape où va commencer la conception proprement dite. L'étape A permet de comprendre le pourquoi

de l'automatisme (les raisons qui ont suggéré le projet de l'automatisation, son contexte d'utilisation, ...) et le quoi (quels sont les fonctions que l'on veut que l'automatisme accomplisse, sous quelles conditions et dans quelles limites). L'étape B aborde le comment : Quel est le découpage en "modules" (au sens de composantes fonctionnelles ou de tâches) que l'on peut définir tel que la coordination des activités de ces modules permette de réaliser les fonctions assignées à l'automatisme ? Quel va être le rôle précis de chacun de ces modules ? Quels seront ses interfaces ? De quelle manière la coordination des activités doit-elle s'effectuer ?

C - *Etude des structures possibles d'implantation, tenant compte des performances des technologies de réalisation disponibles ou imposées, des exigences fonctionnelles et opérationnelles formulées et des coûts* : Une fois l'analyse fonctionnelle suffisamment avancée, le concepteur a à résoudre deux problèmes de faisabilité :

- *faisabilité des exigences fonctionnelles* : Quelles sont les solutions techniques qui permettent la réalisation effective de l'architecture fonctionnelle définie en B ? Quel va être, en conséquence, la part du logiciel et du matériel dans la réalisation des modules de l'architecture fonctionnelle ? Quels sont à priori les architectures logicielles et matérielles envisageables ? Quels sont leurs avantages du point de vue du compromis performances-coût ?

- *faisabilité des exigences opérationnelles* : Ces architectures logicielles et matérielles, telles quelles, peuvent ne pas satisfaire aux qualités de sûreté de fonctionnement souhaitées. Quelles sont alors les structures d'implantation "optimales" qui permettent d'y remédier ? (redondances sélectives ou massives à introduire tant au niveau du matériel qu'au niveau du logiciel, dispositifs autonomes de surveillance en ligne, mécanismes de reconfiguration, ...).

D - *Etude de la conception détaillée des modules logiciels et matériels définis par la structure d'implantation retenue, et précision des protocoles de synchronisation et d'échange entre ces modules.*

E - *Réalisation effective de l'automatisme.*

F - *Vérifications finales et tests d'acceptation.*

En fonction de la complexité de l'automatisme et de ses caractéristiques particulières, l'importance de chacune de ces étapes est plus ou moins grande. Il est bien évident aussi que, en pratique, la démarche du concepteur ne peut être strictement linéaire en suivant le séquencement de ces étapes. La connaissance à priori du matériel de réalisation peut influencer, dès le début, la décomposition fonctionnelle. De même, ne serait-ce que pour des raisons d'optimisation, le concepteur est amené à faire des itérations notamment sur les étapes B à D. Enfin, dans le raffinement progressif de la conception, certaines difficultés (ou impossibilités) peuvent remettre en cause des décisions prises à des stades antérieurs. Il faut alors revenir en arrière pour corriger ces décisions, évaluer l'impact de ces modifications sur le comportement global de l'automatisme et réajuster en conséquence les compromis entre les divers critères de sécurité, de performances et de coût.

Pour conduire ce processus, le concepteur a besoin de s'appuyer sur des méthodes et critères qui facilitent une progression correcte de la conception. Il a aussi besoin de disposer d'un ensemble d'outils compatibles qui permettent de modéliser l'automatisme à différents niveaux de détail et simplifient les vérifications et évaluations qui s'imposent à chacun de ces niveaux. Ces vérifications et évaluations sont de deux types :

- *Validation fonctionnelle de la conception :*

  . Validation qualitative : tout ce qui concerne la vérification de la construction correcte des solutions apportées —tant au niveau algorithmique qu'au niveau de l'implantation— et la cohérence de l'ensemble pour la réalisation d'un fonctionnement "sain" de l'automatisme : Il ne suffit pas, par exemple, d'implanter des procédures qui imposent le respect des contraintes de sécurité fonctionnelle, de synchronisation ou de partage de ressources, ... mais encore faut-il que les effets

conjugués de ces procédures ne puissent en aucun cas conduire à des situations de fonctionnement critique ou de blocage.

. Validation quantitative : évaluation du comportement dynamique pour la prise en compte des contraintes de performances et de temps de réponse ; analyse de ce comportement vis-à-vis de certains choix algorithmiques ou technologiques, ...

- *Evaluation des différents paramètres qui touchent plus directement à la sûreté de fonctionnement* : fiabilité, sécurité, disponibilité, etc.

De nombreux outils existants (qu'il s'agisse de méthodes de description, de modèles d'évaluation ou de simulateurs) peuvent contribuer à ces deux types de vérifications. Cependant, leur utilisation en pratique le long d'un processus complet de conception s'avère difficile. La grande majorité de ces outils n'a pas été motivée principalement par le problème de la conception des automatismes, mais par des disciplines externes (spécification et construction de logiciels, recherche opérationnelle, processus stochastiques, ...) d'où une mise en oeuvre qui impose un effort important d'adaptation. D'un autre côté, l'incompatibilité de ces outils tant sur le plan du formalisme que sur le plan des hypothèses d'application oblige le concepteur à procéder à plusieurs modélisations différentes de son système, utilisant à chaque fois l'outil qui permet de mieux approcher ses hypothèses pour la vérification de points particuliers.

Le développement futur de tels outils (ou l'adaptation des outils existants) pour l'aide à la conception des automatismes doit accorder la plus grande importance à ces deux caractéristiques : compatibilité et facilité de mise en oeuvre. La première caractéristique implique une analyse qui prenne en compte globalement le problème de la conception des automatismes, fait le point des véritables besoins pratiques et organise en conséquence les outils proposés pour les résoudre. Pour satisfaire la deuxième caractéristique, l'élaboration de ces outils doit insister autant sur les approches, méthodes et concepts, que sur les pratiques de leur emploi.

Le thème de notre recherche est précisément de faire cette analyse en suivant de façon aussi pragmatique que possible une démarche complète de conception, depuis le cahier des charges jusqu'à la réalisation effective, et en proposant un ensemble de solutions. Le long de cette recherche nous avons choisi de nous appuyer sur deux idées directrices :

. Structuration et hiérarchisation de la démarche de conception ;

. Utilisation, aux différents stades de la conception et de la validation fonctionnelle, d'une gamme d'outils compatibles bâtie autour des réseaux de PETRI et de modèles dérivés.

Une démarche de conception sûre qui prend en compte globalement l'ensemble des aspects fonctionnels, performances et sûreté de fonctionnement ne nous paraît en effet possible que si elle est organisée de façon structurée et progressive. L'approche que nous proposons est une approche descendante, par raffinements successifs, basée sur une double décomposition des problèmes :

- Une *décomposition verticale* qui suit l'évolution progressive de la conception. Nous venons de voir que la conception d'un automatisme passe par au moins trois stades principaux : la définition de l'architecture fonctionnelle, l'étude d'une structure d'implantation adéquate, et la réalisation effective. Même si les études de ces trois aspects de la conception de l'automatisme sont effectuées par des équipes parallèles, la synthèse finale doit reconsidérer les solutions apportées à chacun de ces aspects dans un ordre strictement hiérarchique : fonctionnel, structurel, technique ou technologique. On peut alors parler de trois niveaux de conception de l'automatisme auxquels correspondent des descriptions, modélisations, évaluations, et problèmes à résoudre de types différents. Dans le cas d'un automatisme complexe, on peut envisager de traiter chaque étape en plusieurs sous-étapes donnant lieu à des sous-niveaux progressifs : chaque sous-niveau ayant comme données d'entrée les résultats de la conception issus du sous-niveau supérieur et comme objectif d'apporter des solutions et de nouvelles précisions vers la concrétisation effective de la conception.

- Une *décomposition horizontale* qui, à l'intérieur de chaque niveau (ou sous-niveau), sépare et traite indépendamment les différents types de problèmes fonctionnels, temporels et de sûreté de fonctionnement.

Vu sous cet angle de la double décomposition, le processus de conception apparaît comme une suite de transformations par raffinements successifs de machines abstraites, partant de la première machine abstraite spécifiée par le cahier des charges et progressant vers la machine concrète qui sera l'automatisme.

Il faut noter que le cahier des charges définit la machine abstraite la plus riche ; puisque le but est de décrire tous les comportements admissibles. La conception construit, par niveaux successifs, des machines abstraites qui restreignent progressivement cette classe de comportements admissibles vers un comportement unique qui satisfait à divers compromis.

Ce qui caractérise précisément un niveau de la conception est une définition, à un certain degré de détail, d'une machine abstraite complète. Tout doit y être, que ce soit sous la forme d'une spécification de contraintes qui restent à résoudre ou sous la forme de solutions décidées en réponse à des problèmes posés à des niveaux précédents.

Une transformation consiste à passer d'une machine abstraite complète d'un niveau donné à une autre machine abstraite complète plus proche de la réalisation. A chacun de ces niveaux, la machine est conçue comme un ensemble de modules qui interagissent. La décomposition horizontale conduit à associer à chaque module un ensemble de "propriétés horizontales" : comportement fonctionnel déterminé par les sous-fonctions que le module doit réaliser, interfaces avec les autres modules du même niveau, contraintes dynamiques, propriétés liées à la sûreté de fonctionnement, ... le but étant de répartir sur les modules, au fur et à mesure de leur définition, les propriétés spécifiées au niveau du cahier des charges de l'automatisme. Le concepteur progresse en s'attachant, à chaque pas, à une (ou à un sous-ensemble) de ces propriétés. Il confirme cette

progression en analysant ses répercussions sur la cohérence de l'ensemble.

Procédant de la sorte, il est possible de prendre en compte globalement l'ensemble des aspects fonctionnels, de performances et de sûreté de fonctionnement, et ce dès les premiers pas ; même si, parce que la démarche de conception l'impose, l'étude et la résolution de certains de ces aspects peuvent se trouver par moments beaucoup plus avancées que pour d'autres.

La conception sera d'autant plus sûre que l'on peut disposer de moyens qui permettent :

- d'une part, de progresser dans la décomposition verticale en vérifiant, à chaque pas, que les propriétés horizontales de chaque module sont bien induites par les propriétés horizontales des modules du niveau inférieur ;

- d'autre part, de mettre en confrontation, à l'intérieur d'un même niveau, des choix de solutions. Il est clair que plusieurs solutions à un niveau donné peuvent induire les mêmes propriétés horizontales du niveau supérieur et il incombe au concepteur de choisir parmi ces possibilités celles qui offrent le meilleur compromis.

La tâche de validation de la conception, en exploitant cette relation d'induction entre les divers niveaux, est considérablement simplifiée.

Décomposition horizontale et décomposition verticale traduisent en d'autres termes la structuration et la hiérarchisation dont nous avons déjà souligné l'intérêt pour ce qui concerne la rédaction du cahier des charges.

Le choix des réseaux de PETRI comme modèle de base nous a été dicté par le souci de disposer d'un outil général pour la description des systèmes discrets, qui soit proche des outils classiques utilisés par les automaticiens (en maintenant les notions d'état et de relation de succession entre états) et qui puisse prendre en compte de façon claire la représentation et l'étude du parallélisme

et de la synchronisation dans le fonctionnement de ces systèmes. Les notions et concepts introduits par les "réseaux de PETRI ordinaires" nous ont paru constituer un bon noyau pour la définition d'un outil répondant à cet objectif.

Les réseaux de PETRI et plusieurs variantes de ce modèle ont déjà été utilisés pour la description et l'étude des propriétés caractéristiques des systèmes à évolutions parallèles, et l'on dispose actuellement de nombreux résultats intéressants à ce sujet. Cependant, la plupart de ces travaux se sont situés dans le contexte d'un fonctionnement "autonome" ; c'est-à-dire où les réseaux de PETRI ne servent qu'à représenter des contraintes fonctionnelles (séquencement, partage de ressources), laissant de côté tous les problèmes liés à la prise en compte du facteur temps, de l'interaction du système avec son environnement (synchronisation sur occurrence d'événement), du déterminisme du comportement du système ... . L'exploitation pratique des résultats obtenus s'en trouve dès lors très limitée.

Peu de travaux ont abordé le problème de l'utilisation des réseaux de PETRI pour la représentation et l'étude des fonctionnements "non-autonomes". De façon particulière en France, on s'est intéressé à l'utilisation des réseaux de PETRI en tant que modèle de description et d'implantation des automatismes séquentiels : Une commission du groupe "Systèmes Logiques" de l'AFCET a travaillé à la définition d'un outil pour la normalisation de la représentation du cahier des charges d'un automatisme logique, le GRAFCET, très inspiré des réseaux de PETRI. D'autres travaux ont étudié la possibilité du passage systématique d'une description dans ce modèle ou dans des modèles dérivés à une réalisation câblée, microprogrammée ou programmée.

Outre le fait que ces travaux sont restés au niveau des automatismes logiques simples, il ne nous semble pas que les caractéristiques fondamentales des systèmes à fonctionnement non-autonome aient été suffisamment mises en évidence et explorées.

C'est le deuxième objectif que nous avons fixé à cette recherche, à savoir la formalisation rigoureuse des extensions qui permettent d'adapter l'utilisation des réseaux de PETRI à la description des systèmes à fonctionnement non-autonome, puis la définition, à partir du modèle obtenu, de méthodes et outils qui puissent répondre de façon directe aux besoins de la méthodologie proposée pour la conception de systèmes discrets complexes.

Nous allons maintenant introduire les chapitres de ce mémoire, présenté en trois parties.

La première partie (chapitres I à III) est consacrée à l'étude du problème de la spécification du cahier des charges :

Le chapitre I pose et développe le problème de la structuration et hiérarchisation de la présentation du cahier des charges d'un automatisme complexe. Une méthodologie est proposée.

Dans les chapitres II et III, nous nous intéressons plus particulièrement aux outils de représentation des spécifications fonctionnelles d'un cahier des charges. Une analyse des besoins est présentée qui permet de préciser les qualités que doivent posséder de tels outils et de situer, par rapport à ces qualités, les avantages et limites des principaux outils existants. Nous rappelons ensuite les définitions de base du GRAFCET et nous proposons des extensions à ce modèle qui permettent de sortir son utilisation du strict cadre des automatismes logiques vers les systèmes de traitements numériques et de commande temps réel de façon générale.

Un exemple d'application à la spécification du cahier des charges du système de commande d'un procédé de distillation est présenté en annexe.

La deuxième partie (chapitres IV à VII) est consacrée à l'étude des réseaux de PETRI (RdP) :

Dans le chapitre IV, nous rappelons les définitions des "RdP ordinaires" (modèle autonome primitif), ainsi que leurs propriétés

caractéristiques. Nous étudions ensuite les principales extensions apportées à ce modèle dans le contexte d'un fonctionnement autonome ; nous démontrons quelques résultats donnant une comparaison de ces extensions du point de vue de la puissance de description.

Dans le chapitre V, nous étudions des fonctionnements non-autonomes des RdP. Nous introduisons progressivement trois extensions de base : "synchronisation", "temporisation" et "interprétation", dont la superposition permet de définir le modèle de travail proposé pour l'étude des systèmes non-autonomes : les "RdP Interprétés". Nous montrons, au fur et à mesure, les modifications que ces extensions induisent sur les propriétés étudiées pour les RdP autonomes.

Les possibilités d'analyse et de vérification des systèmes décrits par les RdP Interprétés sont examinées dans le chapitre VI. Une synthèse des méthodes d'analyse statique développées pour les RdP autonomes est présentée ; on étudie, en parallèle, les conditions sous lesquelles l'application de ces méthodes peut être étendue au cas des RdP non-autonomes. Les possibilités de ces méthodes s'avèrent très limitées et conduisent à envisager des approches mixtes qui utilisent des résultats partiels, obtenus par l'analyse statique, pour enrichir les moyens d'une vérification par simulation.

Enfin, le chapitre VII donne une comparaison détaillée des modèles GRAFCET et RdP Interprétés, et conclut sur le rôle de chacun des deux modèles vis-à-vis de la méthodologie de conception proposée.

La troisième et dernière partie, constituée du chapitre VIII, aborde deux aspects CAO qui sont au coeur de la mise en oeuvre de cette méthodologie :

. La définition d'un outil de vérification et de simulation fonctionnelle multiniveaux de systèmes décrits par des réseaux de PETRI Interprétés.

. Le problème du passage d'une description d'un système en termes de RdP Interprétés (telle qu'elle peut être "validée" par

l'outil de simulation) à une implantation sur mini ou microcalculateur.

Cette recherche s'est appuyée sur des applications concrètes, qu'il s'agisse d'automatismes relativement simples (centrale de sécurité à usage domestique, programmateur de four) ou de systèmes de commande complexes (système de commande d'un procédé de distillation, système de gestion d'un atelier flexible de masticage de carrosseries, système de commande d'une machine-outil d'assemblage). Les études de ces applications ont fait l'objet de rapports de contrat ou de recherche dont on trouvera les références à la fin de ce mémoire. Nous en tirons des exemples simplifiés pour illustrer les concepts et arguments introduits au fur et à mesure dans ces chapitres.

- § -

PARTIE I

APPROCHE ET OUTILS POUR LA SPECIFICATION DU CAHIER DES CHARGES D'UN AUTOMATISME COMPLEXE

CHAPITRE I

STRUCTURATION ET HIÉRARCHISATION DU CAHIER DES CHARGES

I - OBJET ET CONTENU DU CAHIER DES CHARGES D'UN AUTOMATISME 1

II - STRUCTURATION ET HIERARCHISATION DE LA PRESENTATION DU CAHIER DES CHARGES 3

II - 1. DESCRIPTION DU PROCEDE 3

II - 2. SPECIFICATION DE L'AUTOMATISME 7

II - 2.1. Spécifications fonctionnelles 8

II - 2.2. Spécifications opérationnelles 9

II - 2.3. Spécifications technologiques 10

II - 2.4. Caractéristiques de sûreté de fonctionnement 11

III - QUALITES RECHERCHEES DANS UN CAHIER DES CHARGES 13

I - OBJET ET CONTENU DU CAHIER DES CHARGES D'UN AUTOMATISME [GRA.77] [MOA.79]

Le cahier des charges d'un automatisme est le document contractuel qui régit les rapports entre un "client", demandeur d'un matériel de commande, et le "fournisseur" concepteur de ce matériel. A cet effet, ce document peut faire intervenir des considérations juridiques, financières, technico-économiques, ... qui sont annexes aux spécifications techniques qui définissent l'automatisme.

Dans ce qui suit, nous nous intéresserons exclusivement au problème de la structuration et de la représentation des spécifications techniques.

Un automatisme est amené à être inséré dans un système englobant en vue d'automatiser la commande d'un procédé. Le schéma général de fonctionnement est illustré par la figure 1.

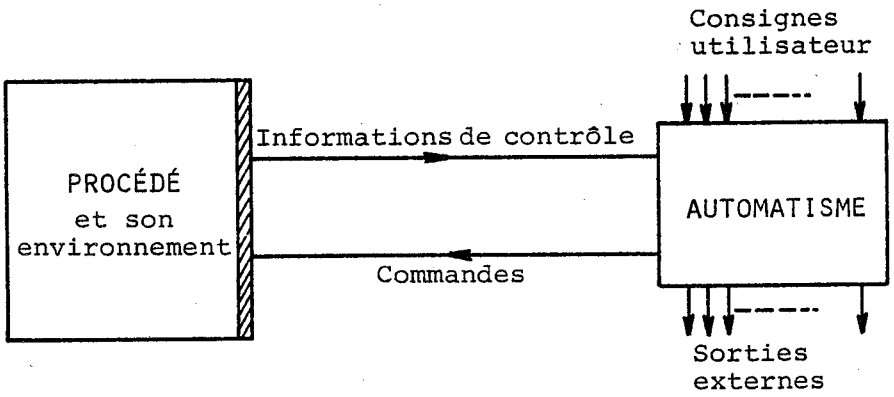


Figure 1.

Le procédé, considéré du point de vue de l'automatisation, est un "mécanisme" qui fait intervenir un ensemble d'opérateurs commandables. L'automatisme a pour rôle de gérer les activités de ces opérateurs de façon à réaliser un fonctionnement donné : Il reçoit en entrées, d'une part, des consignes utilisateur et, d'autre part, des informations de contrôle issues du procédé et/ou de son environnement. Il agit en fonction de ces entrées en émettant des commandes logiques ou numériques vers les opérateurs du procédé et, éventuellement, en élaborant des sorties externes (informations sur l'état du système, visualisations, bilans, alarmes,..).

Nous utiliserons désormais le terme procédé pour désigner globalement l'ensemble procédé et son environnement. L'interface de commande du procédé est celui sur lequel sont appliquées les sorties de commande de l'automatisme et à partir duquel sont recueillies les informations de contrôle.

Les spécifications techniques du cahier des charges ont pour but de préciser :

- d'une part, l'ensemble des fonctions que l'automatisme est amené à assurer et, par conséquent, le comportement qu'il doit avoir dans ses interactions tant avec le procédé qu'avec l'utilisateur ;
- d'autre part, la façon dont l'automatisme doit s'insérer physiquement dans le système global (caractéristiques des capteurs et actionneurs, interface d'exploitation, environnement, ...) et les diverses contraintes technologiques et opérationnelles auxquelles il doit satisfaire.

Ces informations peuvent être présentées en deux volets :

- un premier volet donnant une description du fonctionnement du procédé et de son interface de commande ;
- un deuxième volet consacré à la spécification proprement dite de l'automatisme.

Il est en effet nécessaire, dans un premier temps, que le concepteur puisse se familiariser avec le fonctionnement du procédé avec lequel l'automatisme est amené à interagir. Ce qui intéresse le concepteur est surtout une description du comportement du procédé vu à travers son interface de commande (figure 2) : Quels sont les moyens (disponibles ou envisageables) de commande et de contrôle du procédé et de quelle manière le procédé réagit-il vis-à-vis de ces commandes ?

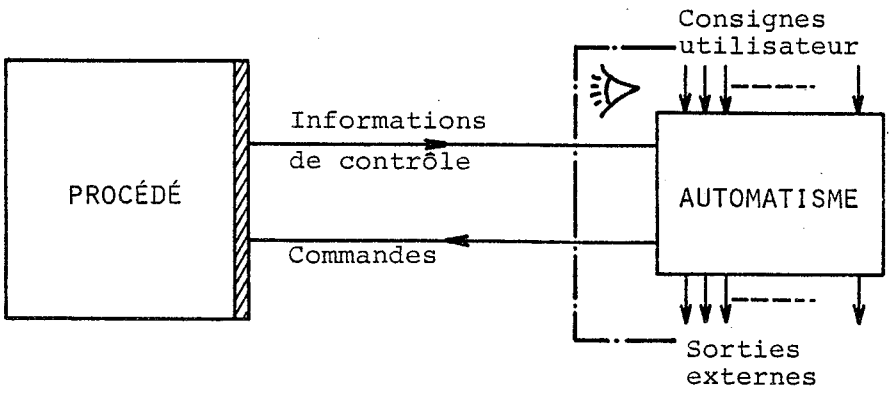


Figure 2.

Cependant, il est souvent difficile de donner une description suffisamment complète du comportement d'un procédé sans donner un minimum de détails sur la structure et le fonctionnement interne du procédé. Par ailleurs, pour un automatisme relativement complexe, la connaissance de la structure et du fonctionnement du procédé peut avoir un impact important sur la façon dont est abordée la conception de l'automatisme et les possibilités de son optimisation.

Dans un deuxième volet, on peut alors spécifier l'automatisme à concevoir en indiquant les différents types de fonctionnement que l'on veut faire effectuer au procédé et les différentes sortes de commandes et de contrôles que l'on veut automatiser.

II - STRUCTURATION ET HIERARCHISATION DE LA DESCRIPTION DU CAHIER DES CHARGES

II - 1. DESCRIPTION DU PROCEDE

La compréhension du fonctionnement d'un procédé complexe est d'autant plus aisée que la description de ce fonctionnement est donnée de façon structurée et progressive :

- Il faut considérer le procédé par "morceaux" (sous-ensembles) de taille suffisamment réduite pour que l'on puisse en maîtriser le fonctionnement.

- La description de ces sous-ensembles peut être faite par étapes : Il importe de donner au concepteur chargé de l'automatisation la possibilité de s'appuyer sur une idée entière du fonctionnement du procédé qui se précise au fur et à mesure. Il faut notamment éviter qu'il se perde dès le départ dans une foule de détails qui complique cette progression. Chaque détail doit être introduit à un stade où le concepteur est capable de comprendre et de juger de l'impact qu'il peut avoir sur la conception de l'automatisme.

L'approche que nous allons exposer s'applique directement au cas des procédés type machines-outils, procédés chimiques ou unités de production de façon générale. On peut s'en inspirer pour le cas de procédés particuliers.

On peut décomposer le procédé en unités. Chaque unité, à son tour, peut être décomposée en modules et les modules en blocs, etc. (figure 3) [RAM.75].

Une première introduction doit permettre de comprendre le fonctionnement de base du procédé (en commençant, s'il le faut, par une description élémentaire des principes qui régissent ce fonctionnement). Elle doit illustrer l'architecture du procédé au niveau des principales unités qui le constituent, et montrer le rôle que remplit chaque unité dans le fonctionnement global du procédé. La même démarche peut être utilisée pour introduire de façon un peu plus détaillée le fonctionnement de chaque unité, aidée par des schémas synoptiques.

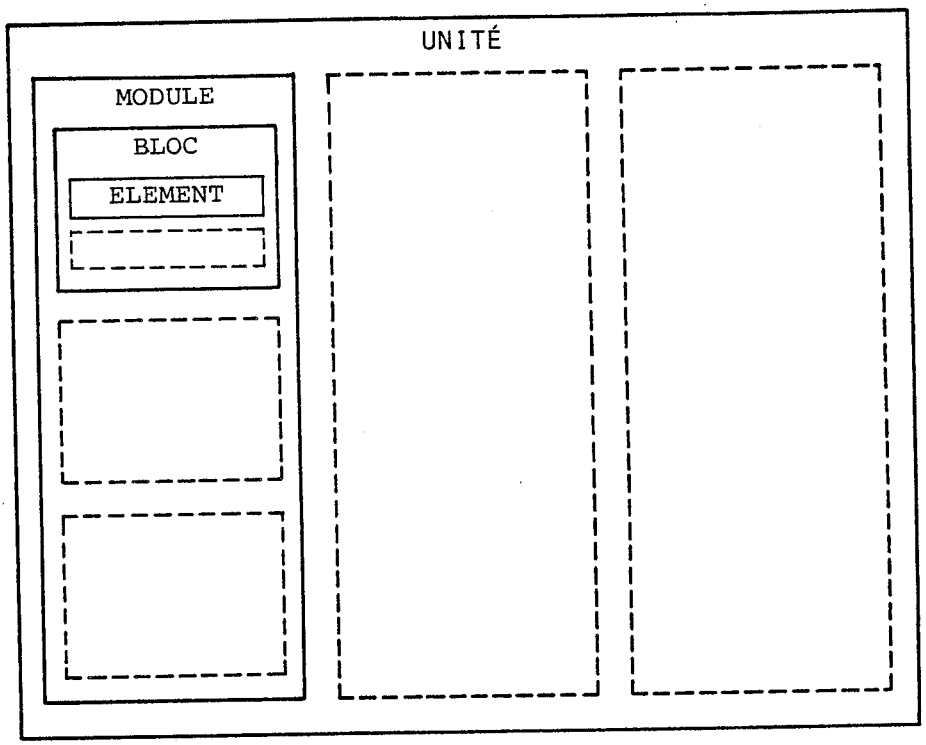


Figure 3.

L'interface de chaque unité (/module /bloc ...) est constitué de deux parties (figure 4) : la partie interface de connexion de l'unité avec le reste du procédé et la partie interface de commande destinée à être pilotée par l'automatisme (et/ou par des dispositifs de commande manuelle).

Dans cette première description, seule la partie interface de connexion au procédé doit apparaître. L'interface de commande —si cela est vraiment nécessaire— peut être décrit de façon abstraite. Peu

importe à ce stade de connaître les détails des moyens de commande et de contrôle puisqu'on ne connaît pas encore sur quels opérateurs ces commandes agissent.

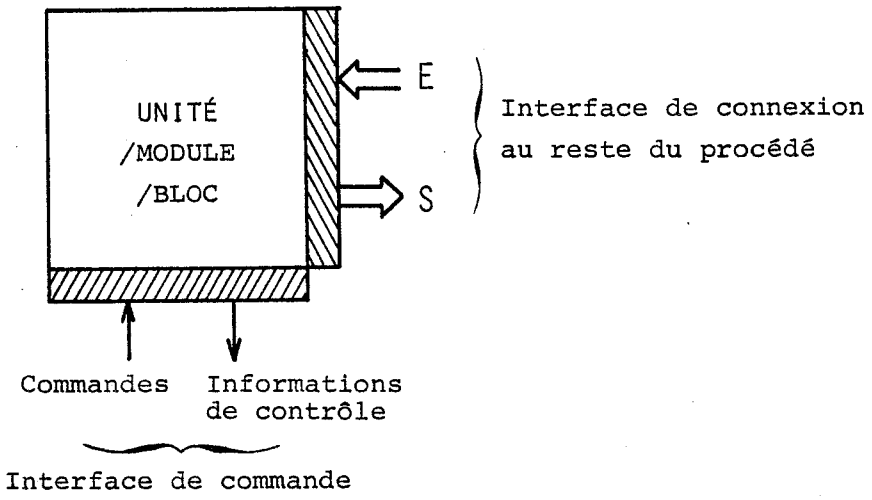


Figure 4.

La description complète et détaillée des différentes unités peut être présentée ensuite dans des chapitres séparés. La description de chaque unité peut être donnée à deux niveaux :

a) à un niveau purement fonctionnel, en commençant d'abord par une description de la structure de l'unité. Un schéma assez détaillé (éventuellement au niveau de chaque module ou de chaque bloc) doit montrer clairement quels sont les principaux opérateurs qui interviennent dans la réalisation des fonctions assurées par l'unité. Un inventaire des moyens de commande et de contrôle peut alors être établi, en précisant quels sont parmi ces moyens ceux destinés à être pilotés par l'automatisme et/ou (exclusivement ou pas) de façon manuelle.

On peut expliciter,à ce niveau,les domaines des commandes acceptables du point de vue de l'intégrité du procédé ; c'est-à-dire, les combinaisons d'activation séquentielles ou parallèles possibles des différents opérateurs, les plages de valeurs pour les commandes numériques, ... Ce faisant, il faut penser à souligner les domaines interdits qui peuvent entraîner une détérioration du procédé ou qui ne sont pas fonctionnels.

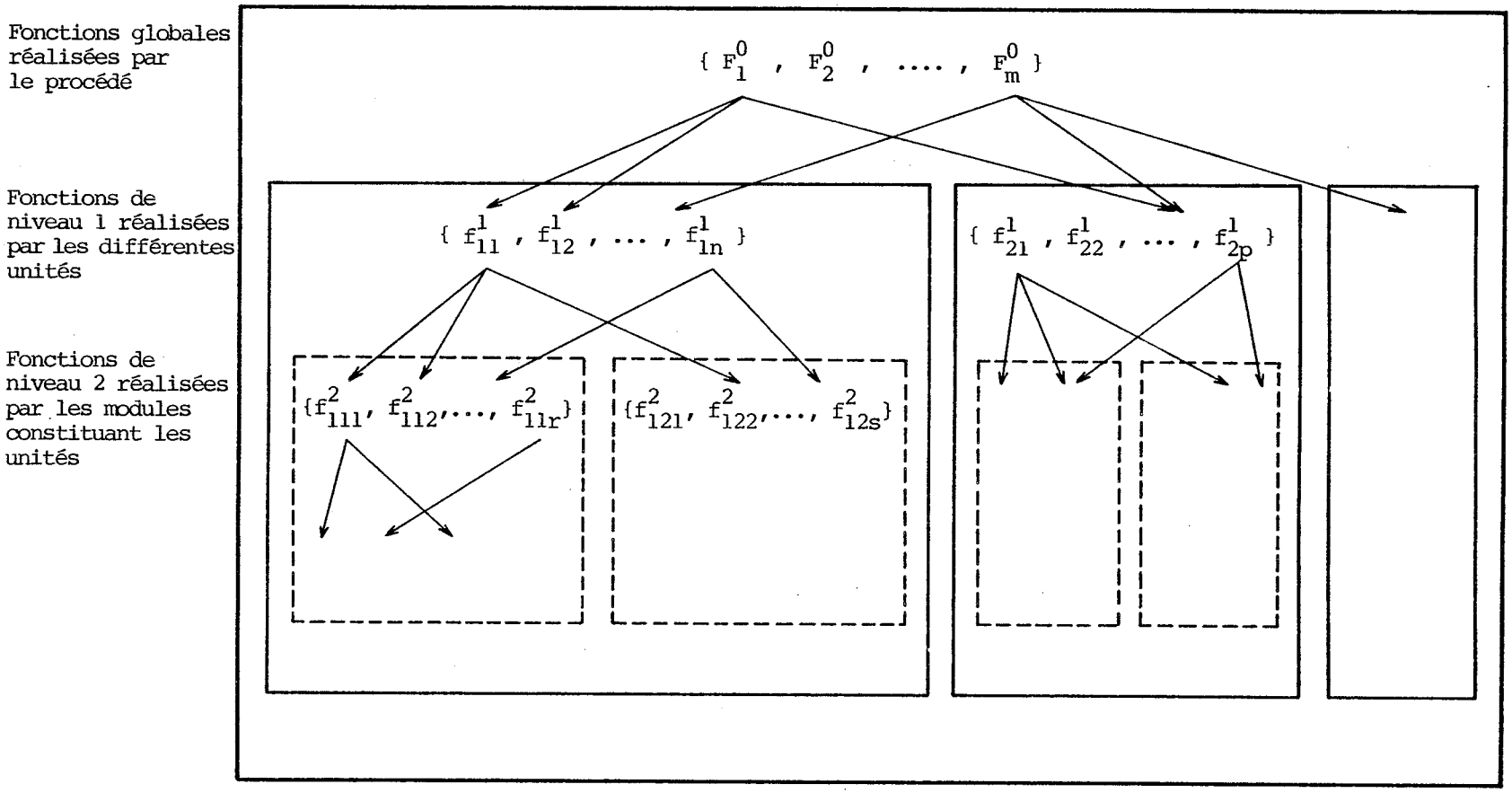

Fonctions globales réalisées par le procédé
Fonctions de niveau 1 réalisées par les différentes unités
Fonctions de niveau 2 réalisées par les modules constituant les unités
{ F⁰₁ , F⁰₂ , ... , F⁰ₘ }
{ f¹₁₁ , f¹₁₂ , ... , f¹₁ₙ }
{ f¹₂₁ , f¹₂₂ , ... , f¹₂ₚ }
{f²₁₁₁, f²₁₁₂,..., f²₁₁ᵣ}
{f²₁₂₁, f²₁₂₂,..., f²₁₂ₛ}

Figure 5.

b) à un niveau technologique, donnant des détails sur la nature physique et les caractéristiques de commande et de fonctionnement des différents opérateurs, capteurs, actionneurs et autres accessoires utilisés. Ces détails technologiques n'intéressent le concepteur que dans la mesure où ils interviennent dans la conception de l'automatisme : pour ce qui concerne la "greffe" de l'automatisme sur l'interface de commande du procédé et, éventuellement, pour la prise en compte de contraintes temporelles dans la commande. Des contraintes de sécurité issues de considérations purement technologiques peuvent être mentionnées.

La décomposition du procédé en unités, puis en modules, ... a pour but de permettre une description progressive et hiérarchisée du fonctionnement de celui-ci. Cette décomposition doit par conséquent être fonctionnelle. Il ne s'agit pas de voir une unité (un module / un bloc) comme une entité matérielle physiquement séparable et ayant certaines caractéristiques technologiques, géométriques, etc. mais surtout comme un sous-ensemble fonctionnel assurant un certain nombre de fonctions bien définies qui interviennent dans la réalisation des fonctions globales du procédé. Ceci est d'ailleurs une tendance instinctive lorsque l'on cherche à expliquer un mécanisme complexe à un non initié.

Procédant de la sorte, on peut présenter le procédé comme un système réalisant un ensemble d'objectifs (fonctions de niveau 0). Ces objectifs sont réalisés par combinaison de fonctions de niveau 1 assurées par les différentes unités ; les fonctions de chaque unité étant réalisées exclusivement à l'aide des fonctions de niveau 2 assurées par les divers modules qui la constituent ... (figure 5).

II - 2. SPECIFICATION DE L'AUTOMATISME

On peut distinguer dans la spécification d'un automatisme trois aspects fondamentaux :

- un aspect *fonctionnel* qui concerne les fonctions de commande et de contrôle que l'on veut automatiser ;
- un aspect *opérationnel* qui a trait à la qualité de service que doit offrir l'automatisme ;
- enfin, un aspect *technologique* se rapportant plus directement à la réalisation matérielle.

A chacun de ces aspects correspond un type de spécifications.

II - 2.1. Spécifications Fonctionnelles :

Il s'agit de décrire les fonctions de commande et de contrôle que l'automatisme doit assurer. La description de chaque fonction doit indiquer de façon claire et précise quelles sont les différentes informations d'entrée qui sont impliquées et la manière dont ces informations doivent être prises en compte (états de réceptivité de l'automatisme, priorités entre actions, contraintes de temps de réponse par rapport à des événements particuliers, synchronisations, ...).

Cette description doit rester purement fonctionnelle. Les détails technologiques de l'interface de commande et de contrôle du procédé auront été déjà donnés dans le premier volet. Quant aux interactions avec l'environnement utilisateur, il convient de s'en tenir à leur contenu sémantique. Les détails des moyens et procédures de communication avec l'environnement extérieur relèvent de l'aspect opérationnel et pourront donc être développés dans la rubrique réservée à cet effet.

Les fonctions de commande et de contrôle qu'un automatisme peut être amené à effectuer sont essentiellement de trois types :

- Des fonctions de surveillance et d'analyse du comportement du procédé. La fonction de surveillance consiste à vérifier que le procédé évolue dans les limites des conditions de fonctionnement normal. Elle peut s'exercer par une simple observation du procédé : dès que l'une de ces conditions cesse d'être remplie, l'automatisme intervient pour entreprendre les actions palliatives urgentes avant d'en avertir l'opérateur. Elle peut aussi s'exercer de façon préventive avant de valider une nouvelle commande, afin de vérifier que cette commande est compatible avec les normes d'utilisation et de sécurité du procédé.

Une surveillance pure peut s'accompagner d'un travail d'analyse. C'est le cas, par exemple, quand il s'agit de fournir à l'environnement utilisateur non pas des mesures brutes prélevées sur le procédé mais une information plus élaborée ; ou quand il s'agit de prédire l'évolution future du procédé tenant compte des mesures prélevées et des consignes de commande en vigueur.

- Des fonctions de commande séqentielle qui gèrent les activations des opérateurs du procédé dans le but de réaliser un ensemble de tâches définies. Ces tâches peuvent être répétitives ou occasionnelles, déclen-

chées sur demande utilisateur ou sur occurrence d'événements particuliers sur l'interface de connexion de l'automatisme avec le procédé.

- Des fonctions de régulation et de conduite qui déterminent de façon dynamique, à partir des mesures effectuées sur le procédé et des consignes en vigueur, les corrections à apporter aux paramètres de la commande dans le but de réaliser un ensemble d'objectifs.

Distinguer ainsi les fonctions de l'automatisme et les décrire dans des paragraphes séparés apporte une clarté certaine aux spécifications. Etant bien entendu que cela ne présume en aucune façon que ces fonctions devront être séparées ainsi dans la conception de l'automatisme.

II - 2.2. Spécifications Opérationnelles :

Ces spécifications rassemblent les désidérata se rapportant aux commodités d'exploitation de l'automatisme. Elles peuvent concerner :

a) les performances souhaitées de l'automatisme. Le terme performance est à prendre dans son sens le plus large ; il peut englober :

- la centralisation / décentralisation (fonctionnelle et/ou géographique) de la commande,
- la facilité de reconfiguration pour s'adapter à divers types de fonctionnements possibles du procédé (par exemple, dans le cas des procédés à structure variable),
- les temps de réponse globaux (non spécifiques à une fonction de commande particulière),
- les taux d'activité relative des différentes fonctions,
- l'évolutivité de l'automatisme, soit pour tenir compte des transformations possibles du procédé, soit pour la mise au point et le perfectionnement de la commande elle-même,
- ....

b) La qualité des interactions avec l'environnement utilisateur :

- facilités de commande à prévoir,
- présentation des sorties externes,
- fonctions de service annexes : constitution d'archives, consultation de ces archives, élaboration de bilans, ...

c) Les caractéristiques de sûreté de fonctionnement que doit avoir l'automatisme. Il s'agit là de l'un des aspects les plus importants des spécifications, cependant souvent négligé dans les cahiers des charges

actuels ; ces caractéristiques étant soit laissées à l'appréciation du concepteur, soit exprimées en termes vagues. Nous reviendrons plus longuement sur ce point dans un paragraphe séparé (§ II - 2.4.).

d) Quoique ne concernant pas directement la qualité de service en exploitation, on peut spécifier dans ce paragraphe des contraintes de test en production pour des automatismes destinés à être fabriqués en grandes séries.

e) .....

La qualité de service se paye par la valeur ajoutée au prix de revient de l'automatisme ainsi que par son coût direct à l'exploitation. Il peut être intéressant de souligner les importances relatives.

II - 2.3. Spécifications Technologiques :

Ces spécifications concernent plus directement la réalisation matérielle de l'automatisme. Elles peuvent comprendre :

a) Des précisions sur les caractéristiques technologiques de l'interface automatisme-procédé et automatisme-extérieur. Une partie de ces informations aurait pu être donnée avec la description du procédé dans le premier volet.

b) Une description de l'environnement dans lequel l'automatisme doit fonctionner : tensions d'alimentation disponibles température, humidité, poussière, chocs, vibrations ....

c) Eventuellement, une spécification des technologies qui peuvent (ou doivent) être utilisées dans la réalisation de l'automatisme. Ces choix peuvent découler de diverses raisons politiques, technico-économiques, ... ou simplement d'un souci de compatibilité avec des technologies déjà utilisées.

II - 2.4. Caractéristiques de sûreté de fonctionnement

L'une des caractéristiques à laquelle on fait le plus référence est la sécurité, c'est-à-dire le fait que les sorties de l'automatisme (qu'il s'agisse de sorties de commande vers le procédé ou de sorties de communication avec l'extérieur) ne puissent en aucun cas être à l'origine d'incidents graves.

Il est important de distinguer deux types de problèmes :

1 - Prévenir les conséquences qui peuvent résulter d'un conflit entre des sorties correctes de l'automatisme et une situation de fonctionnement anormale du procédé ou de son environnement.

2 - Prévenir les conséquences de sorties incorrectes de l'automatisme qui peuvent être dues soit à des erreurs de conception soit à une défectuosité du matériel de réalisation.

L'analyse des problèmes du premier type ne saurait être attribuée au concepteur, en tout cas pas à lui seul. Il est bien évident que l'on ne peut se protéger contre un incident que si l'on maîtrise les circonstances qui peuvent provoquer cet incident et que l'on sait les prévenir. Il est bien évident aussi que le spécificateur, connaissant bien le fonctionnement du procédé et de son interface de commande, est le mieux placé pour faire cette analyse et en déduire ce que l'on peut appeler les sécurités fonctionnelles à prendre en compte par l'automatisme.

Ces sécurités fonctionnelles se traduisent par des conditions qui assujettissent les sorties de l'automatisme, et leur description doit faire partie des spécifications fonctionnelles. Une fois spécifiées, leur réalisation pose des problèmes du deuxième type, dont la résolution incombe cette fois-ci entièrement au concepteur.

Les problèmes du deuxième type, de façon générale (prévention des sorties incorrectes qu'elles soient conditionnées par des sécurités fonctionnelles ou pas), relèvent de ce que l'on peut appeler la sécurité opérationnelle : Il s'agit de faire en sorte que les sorties incorrectes de l'automatisme soient détectées, signalées, éventuellement corrigées, et, en tout cas, leurs effets limités.

Cette sécurité opérationnelle est un paramètre mesurable dont l'estimation doit faire partie précisément de ce grand paragraphe réservé aux spécifications opérationnelles.

Que signifie, par exemple, la spécification : ≪.... . En aucun cas, la commande A ne peut être maintenue si la condition B n'est pas satisfaite ≫ ?

Ce dont on peut être sûr, c'est que le concepteur interprètera cet énoncé au moins comme une sécurité fonctionnelle : Il fera en sorte que la sortie A soit constamment assujettie à la vérification de la condition B.

Cependant, le matériel qui va réaliser cette contrainte n'est pas à l'abri d'une défaillance. Et si une panne survient !

Penser en terme de sécurité "intrinsèque" (la certitude de l'"absolu") n'a pas de sens dans le contexte des technologies actuelles. Il faut compter avec la fiabilité du matériel utilisé, qui est une valeur statistique, et donc penser plutôt en terme de probabilité. Dans le cas de l'exemple ci-dessus, on remplacera la spécification « ... . En aucun cas, ... » par « ... . La probabilité que cette sécurité fonctionnelle soit mise en cause durant la période de service pour laquelle l'automatisme est conçu doit être inférieure à p ». Cela veut dire, intuitivement, que si l'on met en fonctionnement un grand nombre de tels systèmes, alors les précautions prises doivent faire que la proportion des systèmes sur lesquels on constate cet incident ait de fortes chances d'être inférieure à p.

La normalisation en usage consiste à décrire la sûreté de fonctionnement comme une grandeur vectorielle dont les composantes sont exprimées à l'aide de probabilités en fonction du temps [SUR.76].

Les deux premières composantes, la *fiabilité* $R(t)$ et la *disponibilité* $A(t)$, classent les fonctionnements d'un système en deux parties : les fonctionnements corrects qui sont ceux correspondant aux spécifications et tous les autres qui sont jugés incorrects :

- la fiabilité à l'instant t, relativement à un instant d'origine 0, est la probabilité de n'avoir que des fonctionnements corrects durant l'intervalle [0,t] ;

- la disponibilité à l'instant t est la probabilité d'être en état de fonctionnement correct à cet instant. La différence avec la fiabilité est que l'on admet que des fonctionnements incorrects soient détectés et réparés durant l'intervalle [0,t[.

On introduit deux autres composantes qui sont la *sécurité* $S(t)$ et la *crédibilité* $C(t)$ lorsque l'on peut distinguer deux types de mauvais fonctionnements : les mauvais fonctionnements dangereux présentant des risques d'accidents graves (perte de vies humaines, destruction de matériels coûteux ....) et les mauvais fonctionnements bénins ou non dangereux (par exemple, qui entraînent au plus des coûts d'immobilisation ou des pertes de production)

- la sécurité à l'instant t est la probabilité de ne pas avoir un fonctionnement dangereux de l'instant 0 jusqu'à l'instant t ;

- la crédibilité à l'instant t est la probabilité de ne pas être en état de fonctionnement dangereux à cet instant. La crédibilité est vis-à-vis de la sécurité ce qu'est la disponibilité vis-à-vis de la fiabilité.

Une dernière composante est la *maintenabilité* $M(t)$ qui concerne l'aptitude du système à la réparation. L'origine 0 est prise en référence à l'instant où une action curative est engagée. La maintenabilité $M(t)$ exprime alors la probabilité qu'un état de fonctionnement correct puisse être rétabli dans un temps inférieur à t.

III - QUALITES RECHERCHEES DANS UN CAHIER DES CHARGES

Dans le contexte de l'évolution actuelle, à la fois de la complexité, des exigences de sûreté de fonctionnement, et des outils de conception et de synthèse, la clarté et la rigueur avec lesquelles les objectifs du cahier des charges sont définis prennent une importance capitale.

Cette clarté doit être obtenue en premier lieu par une structuration rationnelle du contenu du cahier des charges. Nous venons de donner quelques critères techniques de base pour une telle structuration : séparation de la description du procédé à commander (ce qui existe, ou du moins est complètement défini au moment où l'on rédige le cahier des charges) des spécifications de l'automatisme (ce qui est à concevoir) ; distinction des divers types de spécifications ; classification des fonctions (ou des exigences) s'apparentant à un même type de spécifications, ... D'autres critères peuvent être ajoutés, dépendant du type particulier d'application, de l'expérience du spécificateur, de sa connaissance et de son analyse des problèmes posés par l'automatisation du procédé.

Nous avons insisté de même sur l'intérêt d'une hiérarchisation de la description, tant du procédé que de l'automatisme, de façon à rendre le cahier des charges plus pénétrable.

Nous verrons dans le chapitre II que la clarté du cahier des charges peut être accrue aussi par l'utilisation d'outils de représentation appropriés qui facilitent une description rigoureuse et hiérarchisée des spécifications. La communicabilité des spécifications est d'autant

facilitée que ces spécifications utilisent des outils de description normalisés.

Une autre qualité essentielle recherchée dans un cahier des charges est la complétude : le comportement et la fiabilité des services attendus de l'automatisme doivent être définis de façon complète et précise, tout en laissant au concepteur la plus grande latitude dans les choix de réalisation. Deux écueils sont en effet à éviter :

- la sous-spécification qui conduit le concepteur à prendre des décisions qui incomberaient normalement au spécificateur ; cela risque de mener à une réalisation qui, quoique conforme aux spécifications, peut ne pas correspondre aux intentions (non formulées) du spécificateur ;
- la surspécification qui empiète sur les choix de conception et dont l'effet peut être de fermer la porte à des possibilités de réalisation qui sont techniquement ou économiquement plus avantageuses.

On en déduit les qualités sous-jacentes qui sont l'exactitude (formalisation rigoureuse et fidèle des intentions du spécificateur), la précision (description suffisamment complète et détaillée afin d'écarter tout risque d'ambiguité), la cohérence (faisabilité des exigences tant fonctionnelles qu'opérationnelles et absence de contradiction des spécifications entre elles) et l'indépendance maximum des spécifications vis-à-vis des outils et des technologies de réalisation.

On peut enfin noter l'aptitude des spécifications à la modification en cas de mise à jour intervenant en cours d'étude, ce qui peut être obtenu par une modularité des spécifications et une mise en évidence des liens entre les différentes parties qui les constituent.

Toutes ces qualités concourent à ce que les responsabilités et les niveaux de décision de chacune des parties client et concepteur soient clairement définis dès le départ, et qu'ainsi la conception puisse apporter les solutions qui répondent le mieux aux besoins posés.

- § -

CHAPITRE II

OUTILS DE REPRÉSENTATION DES SPÉCIFICATIONS FONCTIONNELLES

I - NECESSITE D'OUTILS DE REPRESENTATION 1

II - APPROCHES POSSIBLES 2

III - LE MODELE GRAFCET 7

III - 1. DEFINITIONS DE BASE 7

III - 2. EXEMPLE 11

III - 3. CONVENTIONS DE REPRESENTATION 13

III - 3.1. Identification des étapes 13
III - 3.2. Spécification des actions 14
III - 3.3. Spécification des réceptivités 15
III - 3.4. Simplification de la représentation des arcs 16
III - 3.5. Spécifications temporelles au niveau des actions 18
III - 3.6. Spécifications temporelles au niveau des réceptivités 20

III - 4. PRECISIONS SUR LES REGLES D'INTERPRETATION D'UN GRAFCET 21

III - 4.1. Evénement et condition 21
III - 4.2. Occurrence d'un événement 22
III - 4.3. Simultanéité des occurrences d'événements 25
III - 4.4. Action de calcul associée à une étape 25

III - 5. ALGORITHME D'INTERPRETATION 27

I - NECESSITE D'OUTILS DE REPRESENTATION

Parmi les trois aspects des spécifications d'un automatisme, les spécifications fonctionnelles sont celles qui occupent de loin la place la plus importante, et leur description pose généralement le plus de difficultés. Donner cette description entièrement dans le langage courant présente, outre la lourdeur, un risque permanent d'incompréhension ou de malentendu : plusieurs mots, d'usage courant, sont peu précis, mal définis, ou —ce qui est plus critique— possèdent plusieurs sens. Aussi, le langage courant s'adapte mal à la description des dépendances entre actions, telles que le séquencement, le parallélisme et la synchronisation. Il apparaît dès lors nécessaire de recourir à des outils de représentation appropriés qui puissent rendre de telles descriptions plus compactes, mieux assimilables et, surtout, dépourvues d'ambiguités.

Pour permettre d'approcher le mieux possible les qualités recherchées d'un bon cahier des charges (chap. I, § III), de tels outils devraient posséder les caractéristiques suivantes :

- être faciles d'usage et de compréhension : un formalisme compliqué est source d'erreurs tant dans son utilisation que dans son interprétation ;
- être suffisamment riches en concepts pour couvrir une gamme de cahiers des charges assez étendue ;
- les primitives de description issues de ces concepts doivent être suffisamment souples pour permettre au spécificateur d'exprimer le plus directement et le plus complètement possible ses désidérata sans que les limitations de ces primitives l'obligent à introduire des surspécifications. Ces primitives doivent être d'une interprétation rigoureuse ;
- de tels outils doivent s'adapter à une spécification modulaire et hiérarchisée, ce qui implique en particulier la possibilité de les utiliser pour des descriptions à différents niveaux de détail ;
- il est souhaitable que ces outils puissent présenter un aspect visuel : d'une façon générale, on exprime plus directement un schéma fonctionnel à l'aide d'un graphisme qu'à l'aide de phrases ;
- enfin, de tels outils doivent être assortis de méthodes de vérification et d'analyse qui permettent de mettre en évidence certaines incomplétudes ou contradictions des spécifications : insuffisances, incohérences, ambiguités, blocages, impossibilités, ...

Cette dernière caractéristique est importante et illustre davantage la nécessité de recourir à des outils de représentation adéquats.

II - APPROCHES POSSIBLES [CAS.80]

D'une façon générale, une spécification définit une sorte de "machine abstraite" que le concepteur interprète pour en comprendre le comportement et le recréer dans une réalisation concrète. En fonction du degré de liberté laissé au concepteur, cette machine abstraite peut admettre, non pas un comportement unique, mais une classe de comportements acceptables ; le choix parmi ces comportements sera déterminé progressivement par les décisions prises au fur et à mesure de la conception et de la réalisation. Pour la clarté de la spécification, la description de la machine abstraite peut contenir des redondances. Elle peut aussi faire intervenir des variables intermédiaires (autres que les variables d'entrée ou de sortie) et des opérateurs qui n'ont pas à correspondre nécessairement à une réalité physique.

Les outils de spécification en usage ou en cours d'étude peuvent être classés selon les trois types de description d'un système proposés par BARBACCI dans [BAR.75], à savoir : les types structurels, fonctionnels ou comportementaux.

1 - Approche structurelle :

Il s'agit d'une approche par analogie : on spécifie le comportement désiré d'un système en décrivant dans une certaine technologie la structure d'une machine présentant ce comportement. C'est le cas de l'approche schémas à relais ou ampli-opérationnels utilisée autrefois pour rester proche de la technologie d'implantation, et qui continue d'être utilisée alors que la technologie a évolué (automates programmables, microprocesseurs) afin de tirer le meilleur parti du savoir faire acquis. On peut aussi classer dans ce type d'approche la spécification à l'aide d'un langage de programmation, dans la mesure où le programme, lié à des choix matériels sous-jacents, est censé représenter une réalisation possible du comportement désiré.

Le fait que les primitives de description soient inspirées d'une technologie particulière limite inévitablement les champs d'application de tels outils à des secteurs cloisonnés.

De plus, dans ce type d'approche, le rôle du spécificateur et celui du concepteur ont tendance à se confondre :

- s'agissant d'une approche qui vise, par essence, à ce que la spécification soit quasiment réalisable telle quelle, le spécificateur est amené à proposer une solution possible (en référence à la technologie en question) plutôt que de se restreindre à spécifier une classe de comportements admissibles ;

- à partir d'un certain degré de complexité, l'extraction du comportement décrit par une telle spécification devient difficile. Les possibilités d'analyse se réduisent alors à une vérification structurelle (correction de syntaxe ou de construction) et le concepteur aura tendance à simuler cette spécification plutôt que de concevoir une réalisation optimisée. Par exemple, il écrira un programme sur microcalculateur qui simule directement des schémas à relais.

2 - Approche fonctionnelle ou "algorithmique" :

C'est l'approche qui consiste à spécifier le comportement attendu du système en explicitant, à un niveau donné, les algorithmes qui doivent régir son fonctionnement interne. Elle conduit à distinguer deux parties : une *partie opérative* définie comme un ensemble de "tâches", et une *partie commande* qui décrit la synchronisation des activations de ces tâches en fonction des interactions avec l'environnement externe et de façon à réaliser le comportement souhaité.

L'accent est essentiellement mis sur la représentation du fonctionnement de la partie commande. Ce fonctionnement est assimilé à celui d'un automate dont les états constituent les conditions d'activation des tâches de la partie opérative, et dont les évolutions entre états sont une conséquence des résultats de l'exécution de ces tâches et des interactions engendrées avec l'environnement externe.

Cette approche est surtout utilisée pour la spécification des automatismes logiques. A un premier niveau, le procédé est considéré comme étant la partie opérative vis-à-vis de l'automatisme qui constitue alors la partie commande, les tâches correspondant aux activations des opérateurs du procédé. Mais l'automatisme lui-même peut être décomposé en une partie commande et une partie opérative : la première constituée de l'automate de commande proprement dit et la deuxième regroupant un

ensemble de variables et d'opérateurs (registres mémoire, temporisateurs, opérateurs spécialisés ....) qui participent à l'élaboration de la commande (figure 6).

L'automatisme peut aussi être représenté comme l'interconnexion de modules, chaque module étant constitué d'une partie commande et d'une partie opérative (dans le cas, par exemple, d'une commande décentralisée).

Les graphes d'états ont été le premier modèle à être utilisé. Des modèles plus souples, dérivés des réseaux de PETRI, ont été ensuite introduits (RCPP [SIF.74], ORGANIPHASE [PRU.74], RdPI [BLA.76][MOA.3.76], Schémas à Réseaux de PETRI [VAL.76], ...) qui ont notamment l'avantage de condenser considérablement la représentation des états et de faciliter l'expression du parallélisme. Un effort de normalisation de ces modèles a été entrepris par une commission du groupe de travail "Systèmes Logiques" de l'AFCET et a abouti à la définition du GRAFCET [GRA.77].

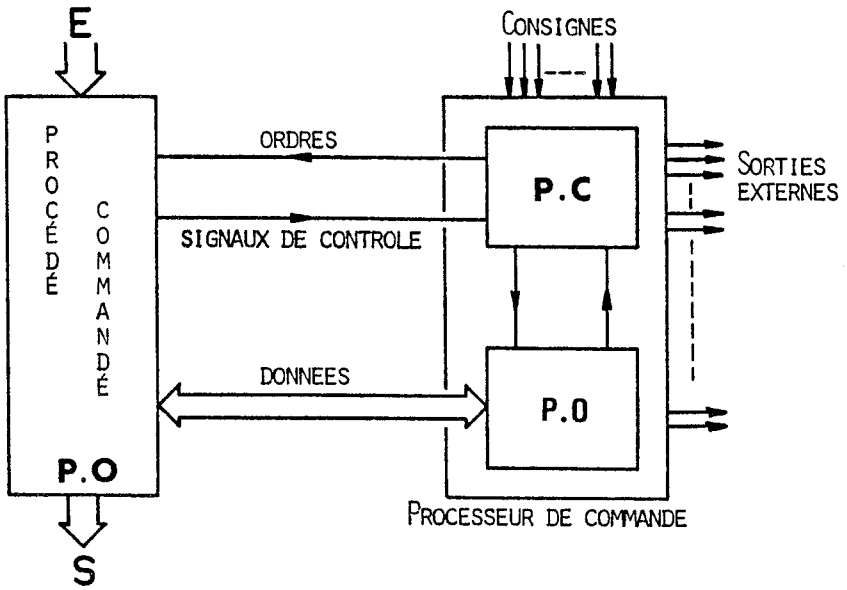


Figure 6.

Basée sur des concepts généraux plutôt que sur des primitives liées à une technologie particulière, cette approche évite les écueils constatés pour l'approche structurelle. De plus, la séparation des parties

commande et opérative est de nature, d'une part, à simplifier la spécification et, d'autre part, à faciliter l'analyse de la complétude en focalisant l'attention sur les évolutions des états qui traduisent le comportement du système ; ce qui constitue l'avantage principal de cette approche, mais qui peut être considéré aussi comme une limitation dans le cas des applications pour lesquelles toute spécification d'états internes est "artificielle" et présente le risque d'introduire des surspécifications. C'est le cas notamment des applications orientées vers le traitement des signaux. Un exemple type est donné dans [CAS.80].

3 - Approche comportementale :

Cette approche consiste à voir le système à décrire comme une "boîte noire" qui réalise une transformation entrées-sorties. La spécification se limite à décrire les relations (fonctionnelles et, éventuellement, dynamiques) qui lient les entrées et sorties, sans préjuger d'aucune façon de la manière dont cette transformation doit être conçue ; ces relations sont représentées sous la forme de formules mathématiques, de chronogrammes, de tables de vérité, ... commentés éventuellement en langue naturelle.

Des propositions ont été faites dans [HEN.79] pour donner à de telles descriptions un cadre plus formel, moyennant l'utilisation de formulaires de description de variables et de fonctions. D'un autre côté, l'accent a été mis dans [ABR.78] et [CAP.78] sur l'intérêt et les possibilités que l'on peut trouver dans l'utilisation du langage mathématique —précision, compréhension universelle, généralité, aptitude à la transformation et à la démonstration— pour la spécification. Il convient de noter que ces propositions, ayant été développées précisément pour la spécification des programmes, ne concernent que l'aspect fonctionnel des relations entrées-sorties. Récemment seulement, des éléments de solutions pour la prise en compte des contraintes temps réel ont été proposés dans [CAS.80].

Conclusions

L'approche structurelle, si elle continue d'être utilisée jusqu'à présent, ne peut se justifier que par des raisons d'habitude et de commodité particulière, dans des secteurs d'application bien délimités où le spécificateur est très associé à la conception et où la nécessité de

nouveaux outils plus généraux ne s'est pas encore fait sentir.

D'un point de vue théorique, l'approche comportementale, basée sur l'utilisation d'un outil formel et rigoureux tel que le langage mathématique, apparaît celle qui permet le mieux d'éviter les risques de surspécification. Sur le plan pratique, on peut souligner deux inconvénients majeurs :

. les spécifications que l'on obtient —du moins avec les outils existants— sont compliquées et difficilement lisibles, ce qui va à l'encontre des facilités de communication recherchées ;

. l'approche comportementale, de par sa définition, ne peut pas être constructive. De là, elle ne facilite en rien la première étape de la conception.

Néanmoins, cette approche n'en est encore qu'à ses débuts et il faudrait attendre le développement d'outils appropriés aux systèmes temps réel ainsi que des exemples de mise en oeuvre pour pouvoir en donner un jugement significatif.

En revanche, l'approche fonctionnelle a commencé à faire ses preuves et nous paraît actuellement celle qui s'adapte le mieux à l'usage pratique.

Dans le cas spécifique des automatismes discrets, la notion d'"état interne" qui résume l'information utile sur le passé du fonctionnement est souvent nécessaire. Aussi, même si en théorie il y a une infinité de spécifications d'états équivalentes pour un automatisme, le plus souvent il y en a une qui a un sens concret et qui s'exprime de façon immédiate. Des outils tels que les réseaux de PETRI ou le GRAFCET permettent de représenter clairement (de façon graphique, et donc visuelle) ces états et les évolutions possibles entre ces états en tenant compte des possibilités ou contraintes de parallélisme et de synchronisation des actions effectuées par l'automatisme. Ces actions sont associées aux états, et leur description est laissée ouverte à tout langage —le plus approprié à l'application traitée— qui permet de donner cette description de façon non ambiguë. Les concepts sous-jacents à ces outils facilitent la vérification et l'analyse des spécifications. Enfin, ces modèles ont déjà été utilisés, non seulement en tant qu'outil de spécification, mais aussi en tant qu'outil de conception et de synthèse des automatismes logiques. De nombreuses maquettes universitaires (une bibliographie assez complète

est donnée dans [BLA.2.79]) ainsi que des produits industriels (TSX 80 - Télémécanique Electrique, MASC 16 - Sems, PROMODUL - Schlekher Elektronik) ont été développés, permettant la synthèse directe, câblée ou programmée, d'automatismes logiques décrits en termes de GRAFCET et de réseaux de PETRI.

Comme nous l'avons annoncé dans l'introduction de ce mémoire, notre objectif est de généraliser l'utilisation du GRAFCET et des réseaux de PETRI en tant qu'outil de travail de base pour la conception sûre d'automatismes discrets complexes.

Nous allons commencer par rappeler les définitions du GRAFCET. Ayant participé à la définition et à la diffusion de cet outil, nous nous permettrons d'insister sur certaines précisions importantes que les publications faites à ce jour sur le GRAFCET n'ont pas, à notre avis, suffisamment explicitées et éclaircies. Nous proposerons ensuite quelques extensions qui permettent d'étendre l'utilisation de cet outil à la spécification des automatismes numériques et des systèmes de commande temps réel de façon générale. Nous enchaînerons par la présentation et l'étude des réseaux de PETRI.

III - LE MODELE GRAFCET [GRA.77][GRA.79][MOA.79][MOA.81]

III - 1. DEFINITIONS DE BASE

Un grafcet est un graphe sur lequel est superposée une interprétation.

a) Le graphe comprend un nombre fini de noeuds de deux types :
   - les étapes représentées par des cercles,
   - les transitions représentées par des traits (figure 7 ; voir aussi nota 1 à la fin de ce paragraphe).

Ces noeuds sont connectés entre eux par des arcs orientés, en respectant les deux règles suivantes :

- un arc ne connecte que des noeuds de types différents (étape à transition ou transition à étape),
- deux noeuds ne peuvent être connectés que par au plus un arc dans chaque sens.

Quand on décrit un système à l'aide d'un grafcet, les étapes sont utilisées pour représenter les différentes situations (états) du système :

. une étape est une composante qui caractérise à tout instant, de façon partielle ou totale, la situation du système décrit ;

. une étape peut être active ou inactive. A tout instant, la situation du système est représentée par la configuration des étapes actives.

Sur le graphe, on peut indiquer qu'une étape est active en plaçant un point (ou une marque) à l'intérieur du cercle associé. On distingue les étapes qui définissent la situation initiale du système en dessinant un deuxième cercle à l'intérieur du premier.

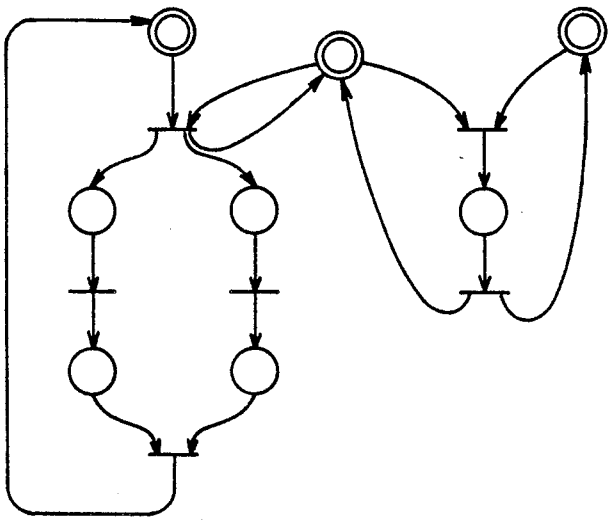


Figure 7. Exemple de grafcet sans son interprétation (graphe uniquement)

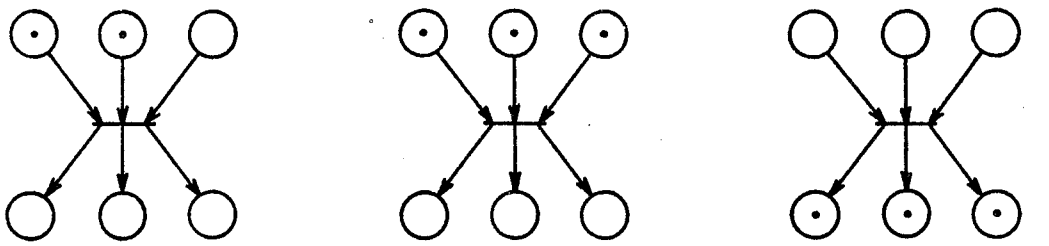
Les transitions représentent les possibilités d'évolution entre situations ; ces évolutions se traduisent par le franchissement de transitions :

. une transition est dite validée si toutes les étapes immédiatement en amont de cette transition sont actives ;

. une transition ne peut être franchie que si elle est validée ; le franchissement consiste à désactiver toutes les étapes immédiatement en amont et à activer toutes les étapes immédiatement en aval (figure 8 ;

voir aussi nota 2).

Ainsi, si une situation s valide une transition t, le franchissement de t conduit à une nouvelle situation s' dans laquelle les étapes en amont de t sont désactivées et les étapes en aval activées. Cependant, le franchissement d'une transition validée peut être soumis à des contraintes déterminées par l'interprétation superposée au graphe.



a) la transition n'est pas validée

b) la transition est validée

c) la transition est franchie

Figure 8. Validation puis franchissement d'une transition

b) L'interprétation associe à chaque étape des actions (éventuellement aucune) et à chaque transition une réceptivité :

- les actions correspondent à ce que le système doit entreprendre chaque fois que l'étape est activée.

- La réceptivité détermine les instants où la transition, si elle est validée, doit être franchie. Elle est composée d'un événement et d'une condition logique (voir nota 3) :

. L'événement traduit le passage de la valeur d'une variable (ou d'une fonction dépendant de plusieurs variables) d'un domaine de valeurs à un autre. Il peut être provoqué par l'environnement extérieur ou résulter d'une évolution quelconque interne au système.

. La condition peut porter sur les valeurs des variables internes au système, sur les valeurs des variables à niveau externes et, éventuellement, sur l'état d'activité ou d'inactivité des étapes.

Le fonctionnement d'un système ainsi décrit par un grafcet est le suivant :

- Les étapes représentées par des doubles cercles définissent la situation initiale du système. Ce sont les étapes qui sont systématiquement activées à l'initialisation du fonctionnement du système.

- Une transition est franchissable à un instant donné si elle est validée et si la condition associée est vérifiée. Une transition franchissable doit être franchie (et ne peut être franchie que) lors de l'occurrence de l'événement associé. Si ce même événement se trouve associé à plusieurs transitions franchissables, toutes ces transitions doivent être franchies simultanément.

C'est précisément ce franchissement simultané sur occurrence d'événement qui constitue la règle essentielle d'évolution entre situations du système.

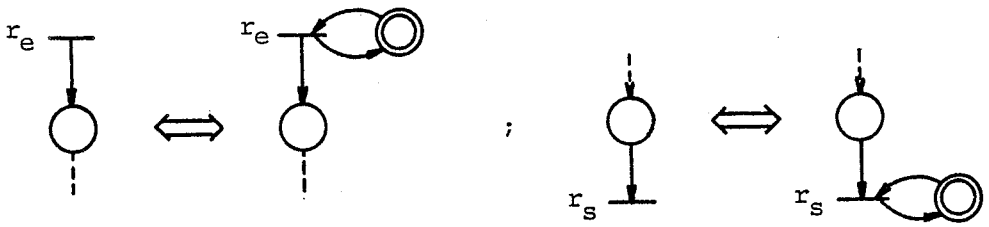
- Si ce franchissement désactive et active simultanément une étape, cette étape reste active.

- L'activation d'une étape déclenche l'exécution des actions associées. Une action peut être à niveau ou impulsionnelle. Elle est dite à niveau si elle est effectuée en permanence tant que l'étape reste active (exemple : sortie à niveau validée tant que l'étape est active). Elle est dite impulsionnelle si elle n'est effectuée qu'une seule fois, pendant une durée limitée, à chaque nouvelle activation de l'étape (Nous entendons par nouvelle activation un nouveau passage de l'état inactif à l'état actif. La réactivation d'une étape déjà active n'entraîne pas une nouvelle exécution des actions impulsionnelles associées).

Nota 1 : Le grafcet a deux représentations possibles. La première, celle que nous reprenons ici, est la forme originale proposée dans le rapport de la commission ayant défini le grafcet [GRA.77]. La deuxième, établie par l'Agence nationale pour le DEveloppement de la Production Automatisée (ADEPA), a pour objet de faciliter le tracé automatique tout en tenant compte des usages généraux des symboles normalisés [GRA.79]. Les deux formes de représentation sont résumées dans le tableau de la figure 15 en fin de ce chapitre.

Nota 2 : Une transition sans étapes en amont est toujours validée ; son

franchissement se limite à l'activation des étapes en aval. Le franchissement d'une transition sans étapes en aval se limite à la désactivation des étapes en amont. Les représentations suivantes sont équivalentes deux à deux :



Nota 3 : La distinction entre événement et condition dans chaque réceptivité n'est pas explicitée dans [GRA.77]. Elle nous paraît cependant nécessaire pour pouvoir énoncer de façon rigoureuse les règles de fonctionnement d'un grafcet (voir § III-5).

III - 2. EXEMPLE

« Un dispositif de sécurité est greffé sur l'interface de commande d'un procédé afin d'inhiber toute commande sur cet interface en cas de détection d'anomalie. La présence d'anomalie est signalée par le procédé à l'aide d'un signal à niveau AL.

Ce dispositif a deux modes de fonctionnement : CONTROLE et SERVICE. En mode CONTROLE, la détection du signal AL a pour effet, outre l'inhibition des commandes, d'allumer un voyant V ; ce voyant restera allumé jusqu'à la remise à zéro de AL. En mode SERVICE, la détection du signal AL active une sirène interne et fait clignoter en même temps le voyant V à la fréquence d'un signal H ; ceci jusqu'à la remise à zéro de AL.

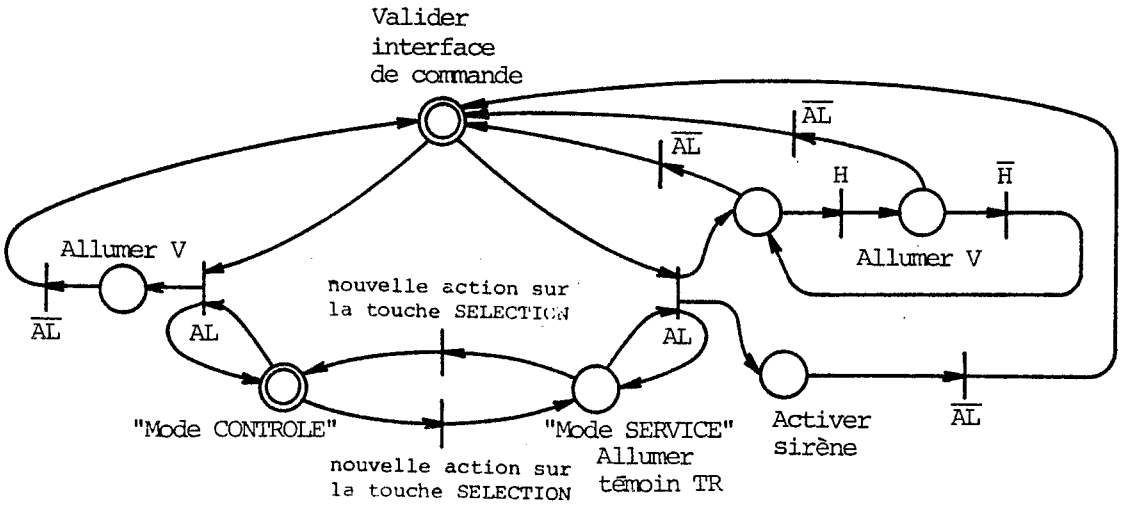
A la mise sous tension du dispositif, le mode est initialisé à CONTROLE. Une touche SELECTION (interrupteur à une position) permet de commander les changements de mode : chaque nouvelle action sur cette touche fait changer de mode.

Un changement de mode peut être effectué à tout moment. Toutefois, si ce changement intervient en présence du signal AL, le comportement du dispositif n'est pas modifié. Le comportement du dispositif reste déterminé

par le mode actif à l'instant de détection du signal AL.

Le mode de fonctionnement est visualisé à l'aide d'un témoin rouge TR, allumé en mode SERVICE... >>

Ce fonctionnement peut être décrit de façon concise et rigoureuse à l'aide du grafcet de la figure 9.



* 7 étapes, 10 transitions, 8 situations différentes, actions associées aux étapes du type à niveau.

Figure 9.

Remarque :

Pour rappeler le schéma donné en figure 1 (chap. I), on peut considérer le dispositif de sécurité comme étant un automatisme dont l'interface avec l'environnement utilisateur est le suivant :

- consignes :
  - commande de changement de mode
- sorties externes
  - allumage du voyant V
  - allumage du témoin TR
  - activation de l'avertisseur sonore

L'interface avec le procédé est réduit :

- dans le sens dispositif-procédé (commandes) :
  . à l'autorisation / interdiction de prise en compte des commandes
- dans le sens procédé-dispositif (informations de contrôle) :
  . à l'information sur l'état normal ($\overline{AL}$) ou anormal (AL) du fonctionnement du procédé.

Le signal H est généré par un oscillateur réglable interne au dispositif.

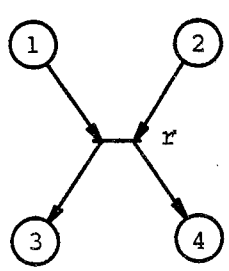
III - 3. CONVENTIONS DE REPRESENTATION

III - 3.1. Identification des étapes

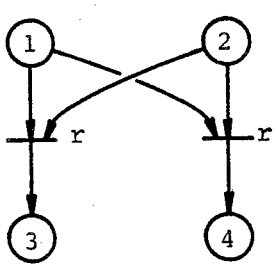
Cette identification peut être faite simplement en affectant à chaque étape un numéro distinct. Quand plusieurs grafcets sont présentés dans un même document, il peut être plus commode d'utiliser plutôt des lettres (ou des étiquettes) indicées : par exemple, une lettre distincte pour chaque grafcet.

L'identification des étapes est utile ne serait-ce que pour faciliter un éventuel commentaire du grafcet. Elle est nécessaire si l'on veut faire intervenir dans l'expression d'une réceptivité quelconque l'état d'activité ou d'inactivité d'une étape. Si α est l'identifieur (numéro ou étiquette) choisi pour une étape, il est convenu d'utiliser la notation φ_α pour désigner la variable booléenne qui vaut "1" si et seulement si l'étape α est active.

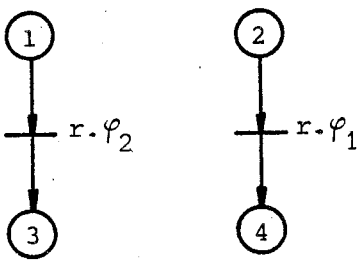
Il découle des règles de fonctionnement énoncées en III - 1. que les représentations suivantes sont équivalentes :



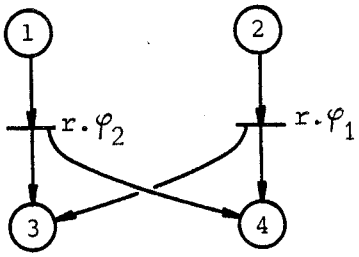
(G1)



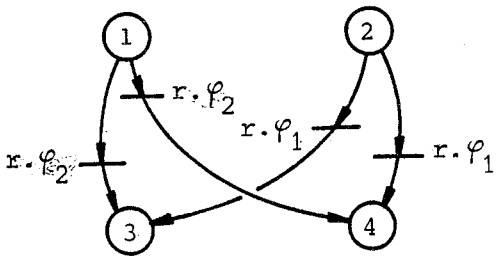
(G2)



(G3)



(G4)



(G5)

* On peut trouver au moins 9 autres représentations équivalentes.
* Certaines de ces représentations (comme G4 et G5) comportent des redondances.

III - 3.2. Spécification des actions

Les actions associées aux étapes peuvent être de natures diverses dépendant du système décrit :

- ouvrir vanne V1, fermer vanne V2
- activer pompe volumétrique PV2
- brasser produit dans ballon tampon B12
- incrémenter compteur C4, décrémenter compteur C10
- filtrer les entrées X et Y
- valider l'interface de commande
- allumer voyant V, activer sirène interne BUZ
- .....

Elles sont élémentaires ou complexes, selon le niveau de détail où l'on veut se situer.

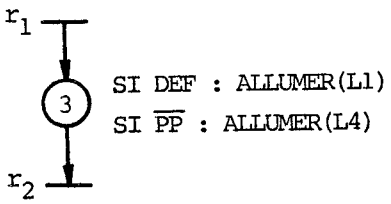
Leur description peut être faite dans un langage libre (à condition qu'il puisse être compris sans ambiguïté) et donnée, sur le grafcet, à côté des étapes auxquelles elles sont associées. Toutefois, lorsque cette description nécessite un texte assez long, il est préférable de ne retenir sur le grafcet qu'une formulation condensée et de donner l'explication détaillée de cette formulation dans un commentaire séparé. Les actions citées en exemple ci-dessus peuvent être formulées comme suit :

- OUV(V1) ; FERM(V2)
- MARCHE(PV2)
- BRASSAGE(B12)

- C4 ← C4+1 ; C10 ← C10-1
- FILTRAGE(X, Y)
- VIC
- V ; BUZ

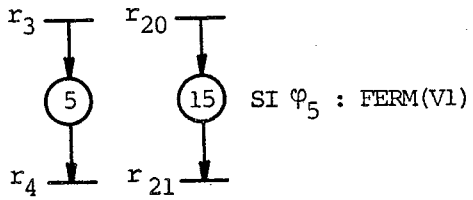
On peut spécifier des actions conditionnelles ; c'est-à-dire des actions dont l'exécution, lorsque les étapes auxquelles elles sont associées sont actives, est soumise à la vérification de conditions portant sur les états d'activité ou d'inactivité d'autres étapes et/ou sur les valeurs des variables du système.

Exemple 1 :



Tant que l'étape 3 est active, on allumera L1 si la variable DEF vaut "1" et L4 si la variable PP vaut "0"

Exemple 2 :



La coïncidence des états d'activité des étapes 5 et 15 valide la commande de fermeture de la vanne V1

III - 3.3. Spécification des réceptivités

Comme pour les actions, les réceptivités associées aux transitions peuvent correspondre à des événements et conditions divers :

- vanne V1 fermée
- pompe PV2 en fonctionnement
- brassage en cours dans le ballon tampon B12
- valeur du compteur C4 égale à 20

- nouvelle impulsion sur l'entrée H pendant que le contact C est ouvert
- que la porte P3 soit ouverte ou fermée, on attend le prochain événement de fermeture
- signal d'anomalie AL présent / absent
- nouvelle action sur la touche SELECTION
- ....

Il convient d'en faire, sur le grafcet, la description la plus compacte possible, en prévoyant, si besoin est, un commentaire séparé.

Les notations suivantes sont recommandées :

. Une variable à niveau X (variable logique) est notée X ou $\overline{X}$ selon que l'on s'intéresse à sa valeur "1" ("vrai") ou "0" ("faux").

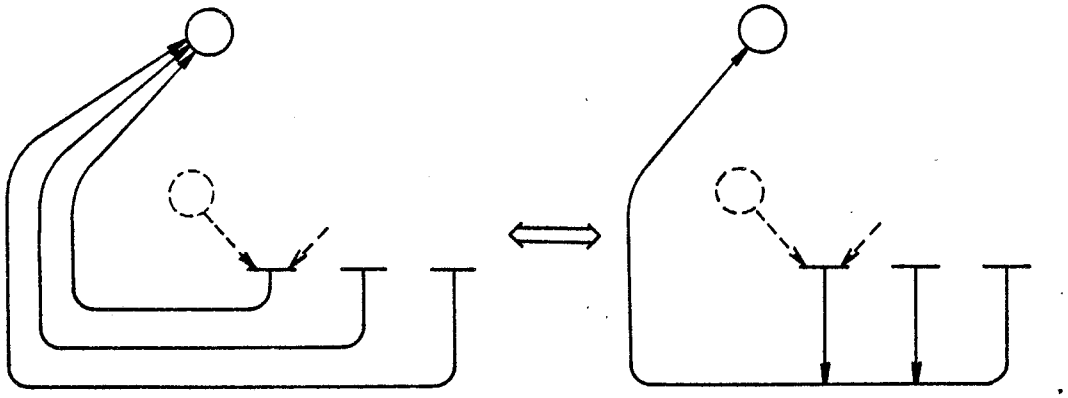
. L'événement correspondant au passage de la variable X de la valeur "0" à la valeur "1" (front montant) est noté par ↑X. De même, l'événement correspondant au passage de la variable X de la valeur "1" à la valeur "0" (front descendant) est noté ↓X. Si X est une expression logique complexe, ces événements sont notés respectivement ↑(X) et ↓(X).

Ainsi, une façon de formuler les réceptivités données en exemple ci-dessus peut être la suivante :

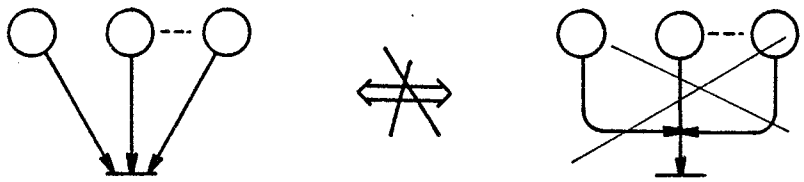
- $\overline{TV1}$
- TPV2
- TBRASSAGE(B12)
- (C4 = 20)
- ↑H.$\overline{C}$
- ↓TP3
- AL / $\overline{AL}$
- ↑SELECTION (ou plus simplement ↑S)

III - 3.4. Simplification de la représentation des arcs

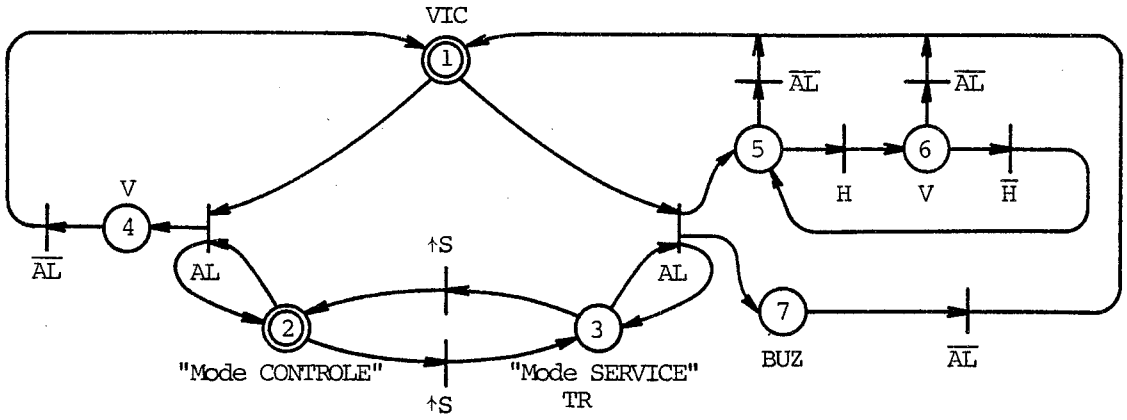
Pour ne pas multiplier les arcs reliant des transitions différentes à une même étape, on peut les fusionner sur une partie de leur tracé comme le montre l'exemple suivant :



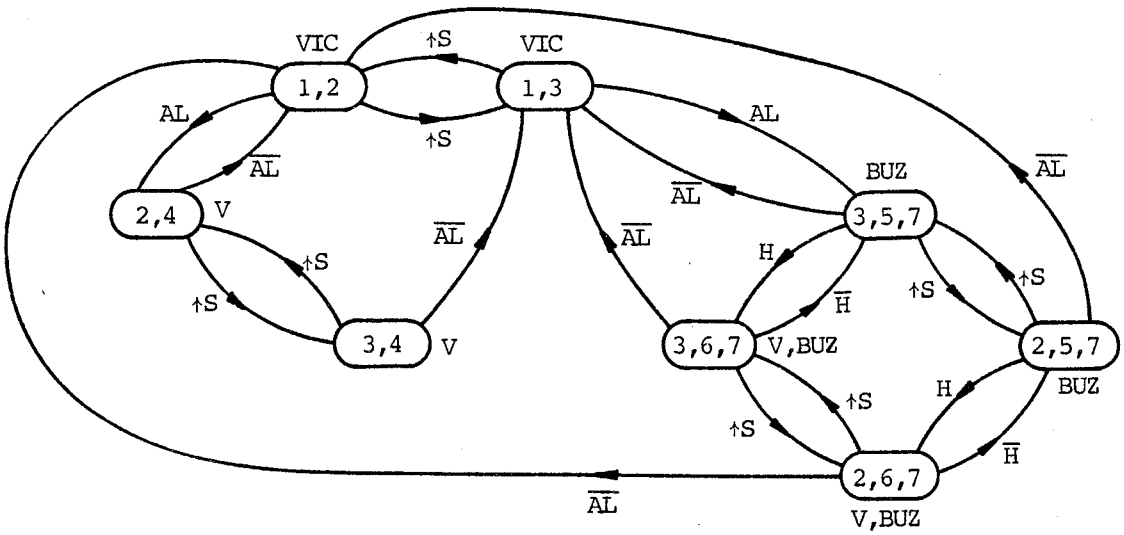
Cette simplification n'est pas correcte pour les arcs reliant des étapes différentes à une même transition. En effet, le point de fusion de deux arcs (⊥) prend la signification d'un OU logique. Son utilisation devant une transition est incohérente avec la signification assignée à la transition qui est celle d'un ET logique entre l'ensemble des étapes en amont.



Compte tenu de ces simplifications, la représentation du fonctionnement du dispositif de sécurité donnée en III - 2 (figure 9) peut être redonnée sous la forme suivante :



Le graphe d'état équivalent est le suivant :



III - 3.5. Spécifications temporelles au niveau des actions

Comme il a été précisé plus haut, les actions associées aux étapes peuvent être de deux types :

- une action à niveau est effectuée tant que l'étape à laquelle elle est associée est active.

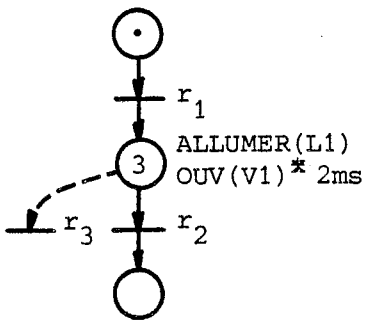
- une action impulsionnelle est effectuée une seule fois, pendant une durée limitée, à chaque nouvelle activation de l'étape à laquelle elle est associée.

Pour les distinguer, on note une action impulsinnelle à l'aide d'un

astérisque suivi, éventuellement, d'une précision de la durée qu'elle doit prendre (figure 10).

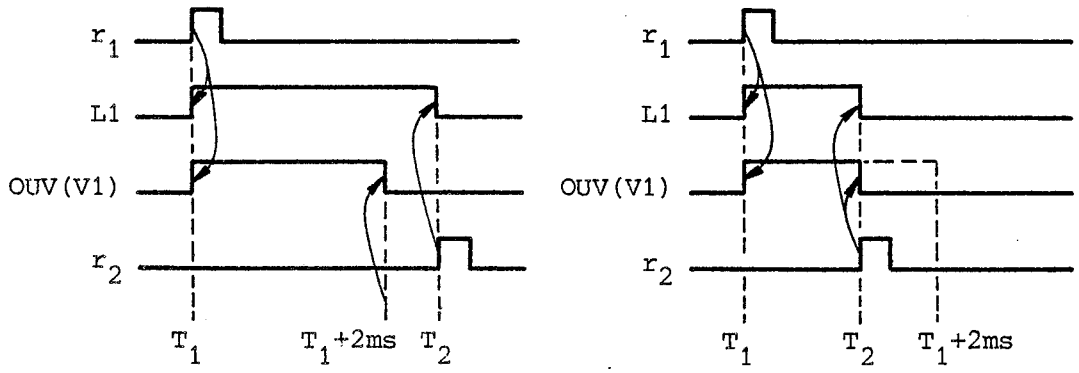
On peut associer à une même étape des actions des deux types. L'activation, ne serait-ce que transitoire, de l'étape engage l'exécution de l'ensemble de ces actions. Cependant, la durée précisée pour une action impulsionnelle est interprétée comme une durée maximum pendant laquelle cette action est effectivement maintenue si l'étape reste active. Si l'étape est amenée à être désactivée pendant cette durée, l'exécution de l'action est interrompue.

Dans l'exemple donné en figure 10, chaque fois que l'étape 3 est activée, la lampe L1 est allumée et la commande OUV(V1) est initialisée pour une durée maximum de 2ms. Si l'on veut que l'action OUV(V1) soit maintenue pendant toute la durée de 2ms, il faut le préciser dans les réceptivités associées aux transitions de sortie comme le montre le paragraphe suivant (figure 12).



Chaque fois que l'étape 3 est activée, la lampe L1 est allumée et la commande OUV(V1) est initialisée pour une durée maximum de 2ms.

On peut avoir les deux diagrammes de temps suivants :



a) $T_2-T_1 > 2ms$ *b)* $T_2-T_1 < 2ms$

Figure 10. Temporisation au niveau des actions

III - 3.6. Spécifications temporelles au niveau des transitions

Une réceptivité peut faire intervenir un test sur le temps écoulé depuis la dernière activation d'une étape. La notation T_α^{tmin} exprime la condition suivante : < le temps écoulé depuis l'instant où l'étape α a été activée pour la dernière fois est supérieur ou égal à tmin >. Ce qui revient à assimiler T_α^{tmin} à une variable logique qui est réinitialisée à "0" à chaque nouvelle activation de l'étape α puis positionnée à "1" après un intervalle de temps tmin. Pour bien marquer que l'étape α est à l'origine d'une temporisation, la valeur tmin peut être indiquée entre parenthèses à côté du cercle représentant l'étape (figure 11 et 12).

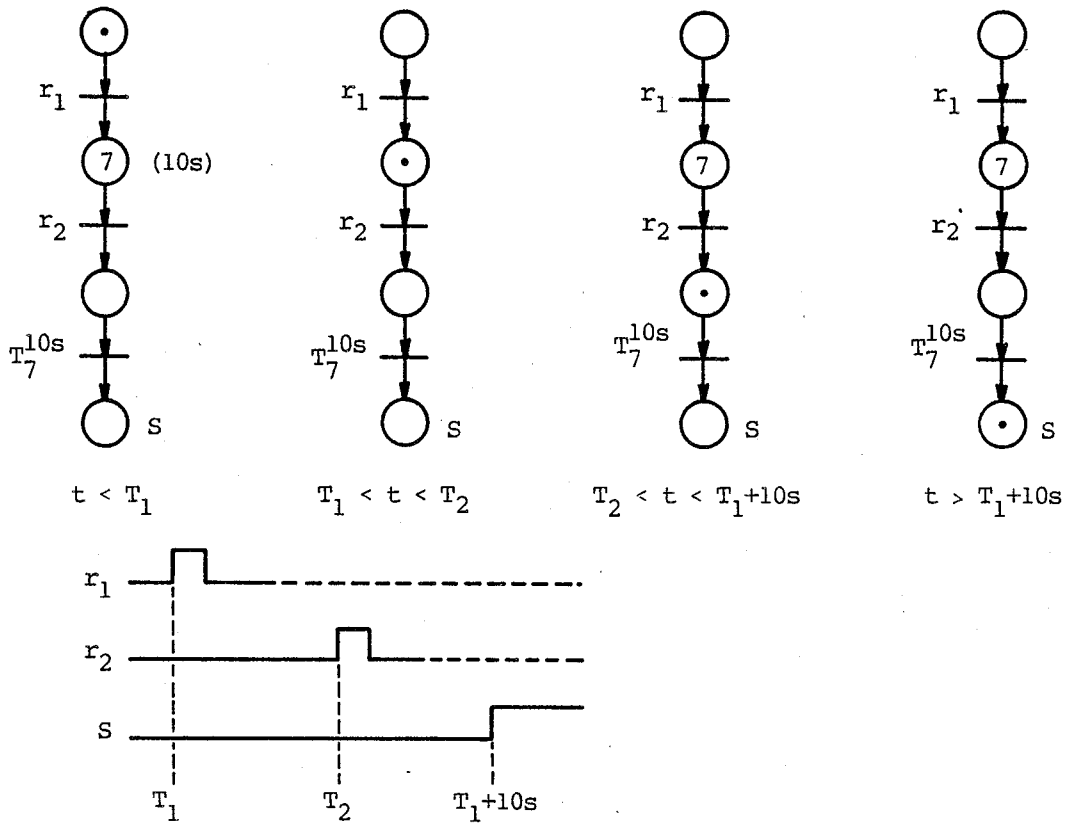


Figure 11. Temporisation au niveau des transitions (exemple 1)

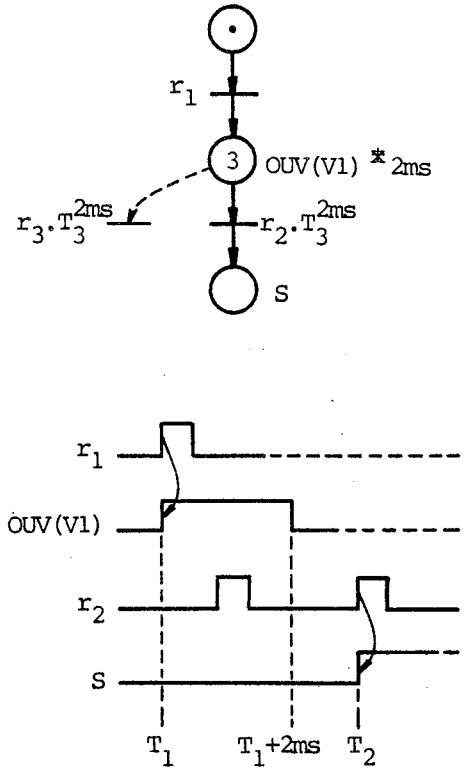


Figure 12. Temporisation au niveau des transitions (exemple 2)

III - 4. PRECISIONS SUR LES REGLES D'INTERPRETATION D'UN GRAFCET [MOA.79] [MOA.81]

Ces précisions ont pour but de cerner de plus près les règles de fonctionnement d'un grafcet afin de lever toute ambiguïté. Elles ne figurent pas dans le rapport de la commission présentant le grafcet [GRA.77] mais nous paraîssent tout à fait en accord avec l'esprit dans lequel ce modèle a été défini.

III - 4.1. Evénement et condition

Dans le paragraphe III - 1, nous avons d'emblée introduit une réceptivité comme étant toujours constituée d'un événement et d'une condition. La définition originelle du grafcet n'est pas aussi explicite sur ce point. Cela nous paraît cependant nécessaire afin, d'une part, d'unifor-

miser la syntaxe des réceptivités pour toutes les transitions et, d'autre part, de pouvoir énoncer de façon rigoureuse les règles de fonctionnement du grafcet (§ III - 5).

On peut considérer un événement se présentant seul dans l'expression d'une réceptivité comme étant associé à une condition toujours vérifiée que l'on peut noter par =1. De même, on peut considérer une condition se présentant seule dans l'expression d'une réceptivité comme étant associée à un "événement toujours présent" que l'on peut noter par e. La définition précise de cet événement sera donnée par la suite.

Une réceptivité non spécifiée explicitement est considérée comme étant composée de l'événement e et de la condition =1. Pour éviter toute ambiguïté à ce sujet (l'absence d'une réceptivité peut être attribuée à un oubli), il est préférable de noter explicitement la condition =1.

III - 4.2. Occurrence d'un événement

Un événement correspond au passage de la valeur d'une variable (ou d'une fonction dépendant de plusieurs variables) d'un domaine de valeurs à un autre. Sur le plan strictement théorique, l'occurrence d'un événement a une "durée" éphémère qui peut être considérée aussi petite que l'on veut ; il en est de même de la durée de franchissement d'une transition. Du point de vue de l'interprétation pratique du grafcet, l'occurrence d'un événement se présente comme une impulsion "1" dont la largeur est juste suffisante pour permettre le franchissement d'une transition (ou le franchissement simultané de plusieurs transitions réceptives à ce même événement).

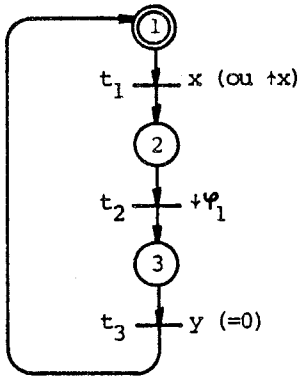
Mais quand doit-on considérer que l'occurrence d'un événement a lieu ?

La réponse paraît triviale quand il s'agit d'un événement engendré par l'environnement extérieur au système décrit. Elle est moins évidente quand il s'agit d'un événement engendré de façon interne. L'exemple du grafcet de la figure 13 l'illustre bien.

On peut comprendre le comportement décrit par ce grafcet de deux façons :

a) la situation représentée par l'étape 2 active est une situation puits : une fois activée, cette étape ne pourra plus être désactivée ;

b) la situation représentée par l'étape 2 active est une situation instable : cette étape est aussitôt désactivée chaque fois qu'elle est



Deux interprétations paraissent possibles :

a) *La situation représentée par l'étape 2 active est une situation puits : une fois activée, cette étape ne pourra plus être désactivée.*

b) *La situation représentée par l'étape 2 active est une situation instable : cette étape est aussitôt désactivée chaque fois qu'elle est activée.*

L'une ou l'autre des deux interprétations dépend de l'instant où l'on considère que l'événement $\downarrow\varphi_1$ a lieu :

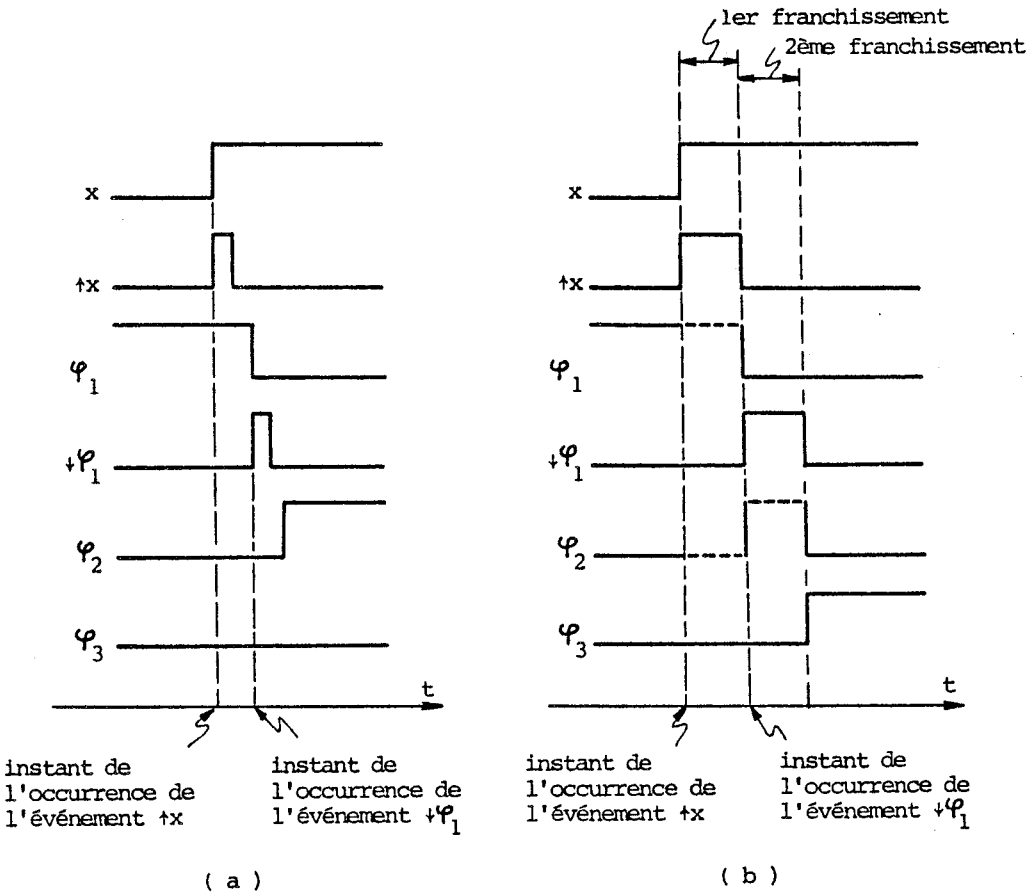


Figure 13. Evénements consécutifs au franchissement d'une transition

activée.

Cela dépend en fait de l'instant où l'on considère que l'événement $\downarrow\varphi_1$ a lieu lors du franchissement de la transition issue de l'étape 1. La figure 13-a suppose que la désactivation de l'étape 1 précède l'activation de l'étape 2. Par conséquent, quand l'étape 2 devient active, l'occurrence de l'événement $\downarrow\varphi_1$ a déjà eu lieu et le grafcet se bloque dans cette situation. On aurait eu une évolution différente si l'on avait considéré l'hypothèse inverse où l'activation de l'étape 2 précède la désactivation de l'étape 1 ; le franchissement de t_1 aurait été suivi immédiatement par le franchissement de t_2.

L'interprétation que nous proposons (illustrée par la figure 13-b) s'attache essentiellement à ce que les activations et désactivations ne soient prises en compte que simultanément à la fin du franchissement :

. l'occurrence d'un événement déclenche le franchissement de toutes les transitions franchissables et réceptives à cet événement ;

. l'événement de désactivation d'une étape ($\downarrow\varphi$) est signalé à la fin du franchissement si cette étape était active au début du franchissement et est constatée inactive à la fin du franchissement ;

. l'événement de nouvelle activation d'une étape ($\uparrow\varphi$) est signalé à la fin du franchissement si cette étape était inactive au début du franchissement et est constatée active à la fin du franchissement ;

. durant le franchissement, la situation du grafcet n'est pas définie.

Cette interprétation est cohérente avec la règle qu'une étape active qui est désactivée et réactivée lors d'un même franchissement reste active. Ce qui signifie que la désactivation d'une étape ne peut être considérée comme étant effective que si elle est maintenue après l'exécution complète du franchissement et, par conséquent, la détection de l'événement de désactivation de l'étape ne peut avoir lieu (et n'a de sens) qu'à la fin de l'exécution du franchissement.

Il s'agit là encore d'une interprétation théorique qui est à suivre pour comprendre sans ambiguïté quelles sont les évolutions décrites par le grafcet. L'implantation physique n'a pas à réaliser obligatoirement cette prise en compte simultanée des activations et désactivations. Elle doit seulement assurer que le comportement du grafcet tel qu'il est

implanté soit conforme aux évolutions décrites selon l'interprétation théorique. Cette remarque est valable pour toutes les précisions d'interprétation que nous serons amenés à donner par la suite.

III - 4.3. Simultanéité des occurrences d'événements

Il est important de distinguer les événements externes engendrés par l'environnement du système décrit et les événements internes engendrés par les propres actions et évolutions du système.

Il découle du paragraphe précédent qu'un système peut toujours décider de façon unique quand les occurrences d'événements internes doivent être considérées comme étant simultanées : quand ces occurrences sont engendrées lors d'un même franchissement et leur détection effectuée à la fin de ce franchissement. L'événement principal est en fait la fin du franchissement qui valide le changement de valeur des variables internes.

Le système n'a pas la possibilité de décider de façon aussi sûre quand les occurrences de deux événements élémentaires externes doivent être considérées comme étant simultanées. Il convient, afin de maintenir une interprétation unique du grafcet quelle que soit la technologie de réalisation utilisée, de considérer (et de traiter) les occurrences des événements élémentaires externes comme étant toujours disjointes.

Les événements de comptage du temps (nécessaires pour gérer les durées limites des actions impulsionnelles et les variables T_α^t) sont considérés comme étant externes.

III - 4.4. Action de calcul associée à une étape

On peut définir une action de calcul (comptage, évaluation de paramètres, ...) comme étant une action impulsionnelle particulière pour laquelle il n'est pas précisé une durée limite. Une action de calcul est effectuée entièrement, une fois à chaque nouvelle activation de l'étape à laquelle elle est associée, et sa durée est considérée, théoriquement, aussi petite que l'on veut. Pratiquement, cette durée doit être telle qu'elle ne puisse d'aucune manière empêcher (ni altérer) les évolutions normales entre situations du grafcet.

Exemple :

<< Le fonctionnement très simplifié du régulateur de vitesse d'un pilote automatique de métro est le suivant : le long de la voie sont placées des balises entre lesquelles sont intercalés un certain nombre de plots ; la détection d'une balise ou d'un plot lors du passage des rames engendre respectivement les signaux B et P. D'un autre côté, le régulateur reçoit des impulsions périodiques HC émises par un poste central de commande. Si n est le nombre de plots détectés entre deux balises B_p et B_{p+1}, et N le nombre d'impulsions HC reçues pendant la durée de ce parcours, la correction de vitesse à apporter au passage devant la balise B_{p+1} est fonction de la différence N - n >>.

Ce fonctionnement peut être représenté par le grafcet de la figure 14 : Deux étapes initiales sont associées respectivement à l'état "arrêt" du régulateur (étape 0) et au point de réinitialisation d'un nouveau cycle de régulation (étape 1). La régulation ne pourra commencer qu'après le passage à l'état "marche" (désactivation de l'étape 0 et activation de l'étape 2). Un cycle de régulation commence dès la détection d'un nouvel intervalle interbalises (front ↓B) : il active les étapes 3 et 5 qui vont servir à compter, respectivement, les occurrences des événements ↑HC et les occurrences des événements ↑P, et ce jusqu'à la détection de la prochaine balise. Les étapes 3 et 5 sont alors désactivées et l'étape 7 est activée pour calculer la correction de vitesse à apporter. On revient ensuite à l'étape 1 pour attendre le début d'un nouveau cycle (nouvelle occurrence de ↓B) si le régulateur est maintenu à l'état "marche" (étape 2 maintenue active).

Dans cet exemple, l'étape 4 n'est pas stable ; elle est désactivée aussitôt qu'elle est activée étant donné qu'à la transition de sortie de cette étape est associée la condition toujours vérifiée. Toutefois, en vertu de la règle que l'activation ne serait-ce que transitoire d'une étape engage l'exécution des actions associées, on doit considérer que l'action d'incrémentation de N a pu se faire pendant l'activité fugitive de l'étape 4 (cette activité fugitive étant de durée aussi petite que l'on veut le supposer théoriquement). Il faut noter aussi que l'étape 3 redevient aussitôt active, ce qui garantit le comptage de toutes les occurrences de ↑HC tant qu'une nouvelle balise n'est pas détectée. L'im-

plantation physique devra faire en sorte que ce fonctionnement soit réalisé. C'est-à-dire que l'opération d'incrémentation de N, qui aura nécessairement une durée non nulle, devra prendre un temps suffisamment court pour que l'étape 3 puisse redevenir active avant la prochaine occurrence de ↑HC, et donc qu'aucune de ces occurrences ne soit perdue. Les étapes 6 et 7 sont également instables et conduisent au même type de raisonnement.

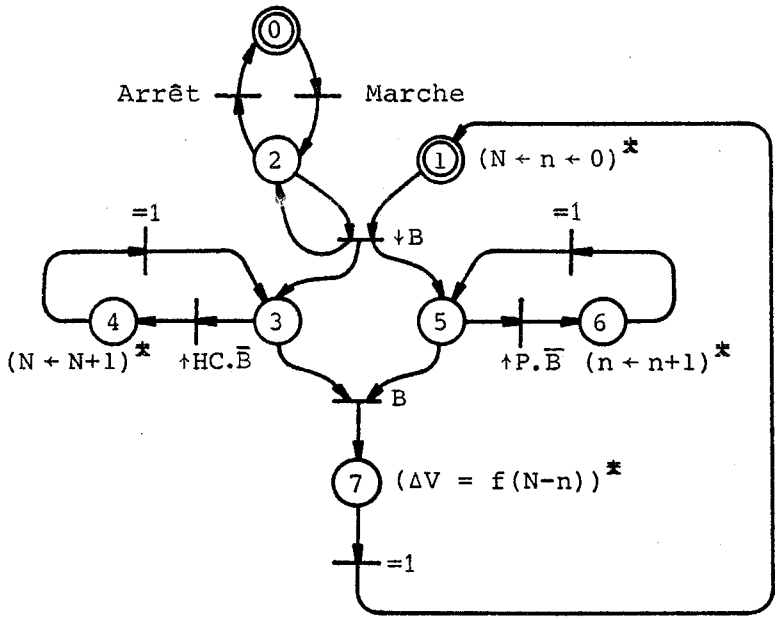


Figure 14. Grafcet du régulateur de vitesse

III - 5. ALGORITHME D'INTERPRETATION

Ces précisions étant apportées, on peut maintenant définir de façon rigoureuse un algorithme d'interprétation unique d'un grafcet :

A : La situation initiale définit l'ensemble des étapes qu'il faut systématiquement activer au départ. Les actions de calcul associées à ces étapes (essentiellement des initialisations de variables) sont effectuées ; les autres actions à niveau ou impulsionnelles, non conditionnelles ou à condition vraie, sont initialisées. Les conditions associées aux transitions sont évaluées.

B : On appelle situation stable une situation à partir de laquelle aucune évolution vers une autre situation n'est possible sans une nouvelle occurrence d'un événement externe. Si la situation initiale n'est pas stable, on doit effectuer une boucle de stabilisation : on considère cette situation initiale comme étant le résultat d'un franchissement qui vient juste de se terminer et on effectue la séquence suivante :

b1) On considère l'ensemble des transitions franchissables et réceptives soit à un événement interne dont une occurrence a été engendrée par le dernier franchissement effectué, soit à l'événement e.

b2) On effectue le franchissement simultané de toutes ces transitions.

b3) A la fin de ce franchissement, les actions associées aux étapes qui viennent d'être désactivées sont annulées. Les actions de calcul associées aux étapes qui viennent d'être activées sont effectuées ; les autres actions à niveau ou impulsionnelles, non conditionnelles ou à condition vraie, sont initialisées. Enfin, les conditions associées aux transitions sont réévaluées.

b4) Si la situation atteinte est stable alors on poursuit en C, sinon on boucle en b1). Un grafcet correctement construit doit atteindre une situation stable au bout d'un nombre fini de pas.

C : L'occurrence d'un événement externe $\uparrow x$ déclenche la séquence suivante :

c1) On considère l'ensemble des transitions franchissables (y compris celles qui sont devenues franchissables par suite de l'occurrence de l'événement $\uparrow x$) et qui sont réceptives soit à l'événement $\uparrow x$ soit à l'événement e (exemple : transitions validées et ayant comme réceptivités $\uparrow x.\overline{z}$ ou $(x+y)$; x, y et z valant "0" auparavant).

c2) On effectue le franchissement simultané de ces transitions.

c3) A la fin de ce franchissement, les actions associées aux étapes qui viennent d'être désactivées sont annulées. Les actions de calcul associées aux étapes qui viennent d'être activées sont effectuées ; les autres actions à niveau ou impulsionnelles, non conditionnelles ou à condition vraie, sont initialisées. Enfin, les conditions associées aux transitions sont réévaluées.

c4) Si la situation atteinte est stable alors on revient en C sinon on boucle en b1) (boucle de stabilisation).

Comme on peut le remarquer, cet algorithme repose sur deux éléments essentiels :

1) La notion de "franchissement simultané sur occurrence d'événement": Le franchissement d'une transition prise isolément et sans référence à une occurrence d'événement n'a pas véritablement de sens dans l'interprétation du grafcet. Tout franchissement est synchronisé sur l'occurrence d'un événement. De plus, toutes les transitions franchissables et réceptives à un même événement sont franchies simultanément.

On comprend mieux alors la signification et l'intérêt de l'événement e. Les occurrences de cet événement coïncident avec toute occurrence d'un événement externe et toute fin d'un franchissement simultané. Par le biais de cet artifice, on peut généraliser la notion de franchissement simultané sur occurrence d'événement au franchissement simultané des transitions franchissables associées à des réceptivités composées exclusivement de conditions logiques.

2) La notion d'"évolution synchrone" : Les évolutions entre situations du grafcet se font pas à pas, chaque pas consistant en un franchissement simultané sur occurrence d'événement. Toute occurrence d'un événement externe provoque un franchissement simultané des transitions franchissables et réceptives à cet événement, suivi d'une série de franchissements sur occurrence de e jusqu'à stabilisation. Une nouvelle occurrence d'événement externe ne peut être prise en compte tant que le grafcet n'a pas atteint une situation stable.

On pourra vérifier que cet algorithme réalise bien les évolutions des grafcets donnés dans les exemples précédents.

Remarques :

i) La réactivation d'une étape déjà active est sans effet : les actions à niveau sont poursuivies jusqu'à la désactivation de l'étape. Les actions impulsionnelles ne sont pas reprises.

ii) L'exécution d'une action à niveau conditionnelle est validée à tout instant où l'étape associée est active et la condition à laquelle elle est assujettie est vérifiée.

Les règles concernant l'exécution d'une action impulsionnelle condi-

tionnelle n'ont pas été définies à ce jour. Nous proposons l'interprétation suivante : une action impulsionnelle conditionnelle est initialisée à chaque nouvelle occurrence de l'événement ↑(étape associée active et condition d'exécution vérifiée). L'invalidation de la condition ou la désactivation de l'étape interrompt l'action. (Une action impulsionnelle non conditionnelle se comporte ainsi comme une action impulsionnelle conditionnelle associée à une condition toujours vérifiée).

iii) Insistons une dernière fois sur le fait que l'interprétation théorique des durées d'un événement, d'un franchissement ou d'un calcul a pour seul but d'assurer une interprétation unique du grafcet, indépendante de toute technologie d'implantation. Il est clair qu'aucune technologie ne permet de réaliser strictement une telle interprétation. L'important est que, compte-tenu des caractéristiques de la technologie d'implantation et de celles du comportement de l'environnement, le comportement de l'automatisme réalisé soit le même, à une marge près de temps de réponse tolérée par le cahier des charges.

- § -

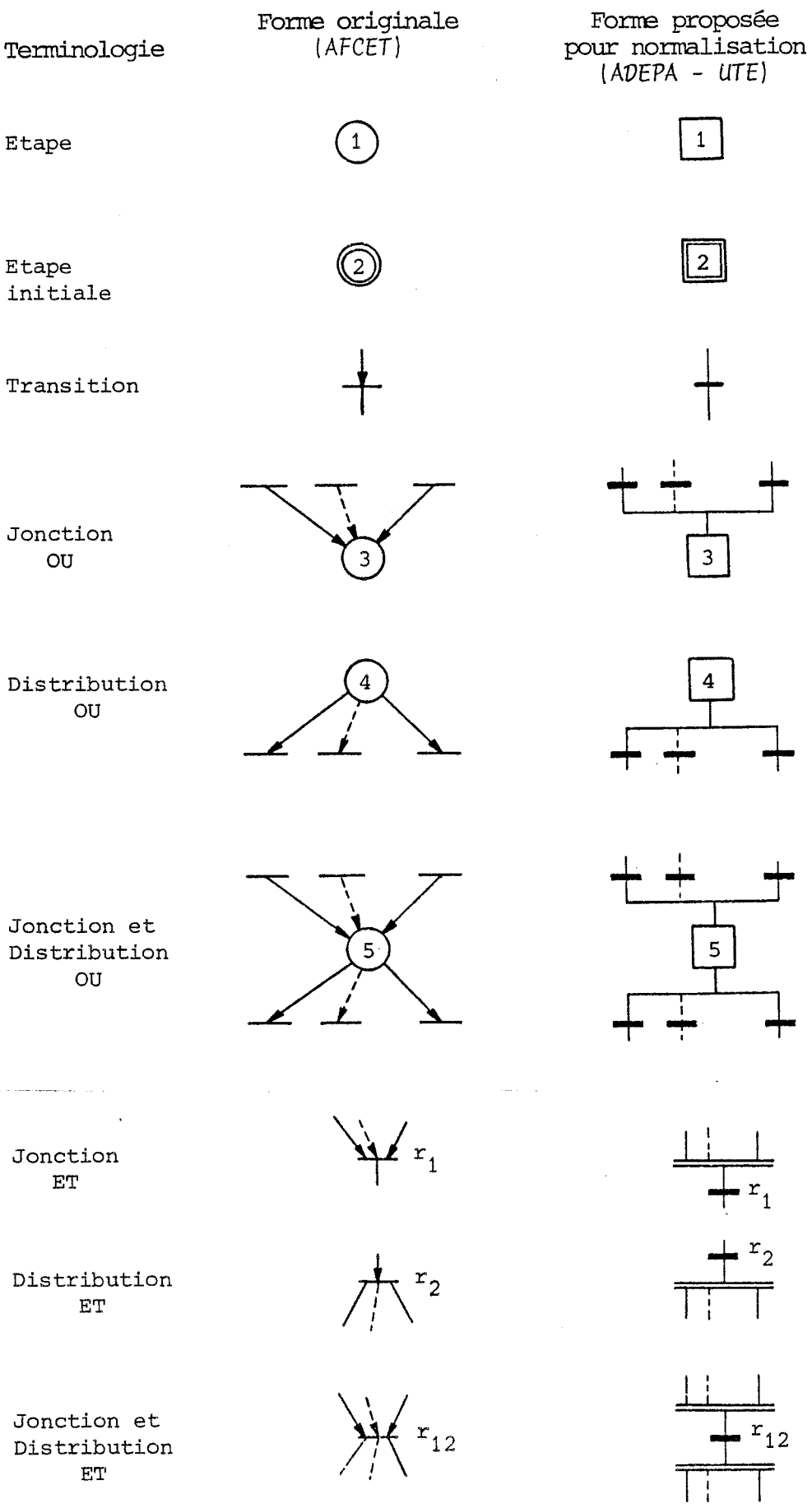
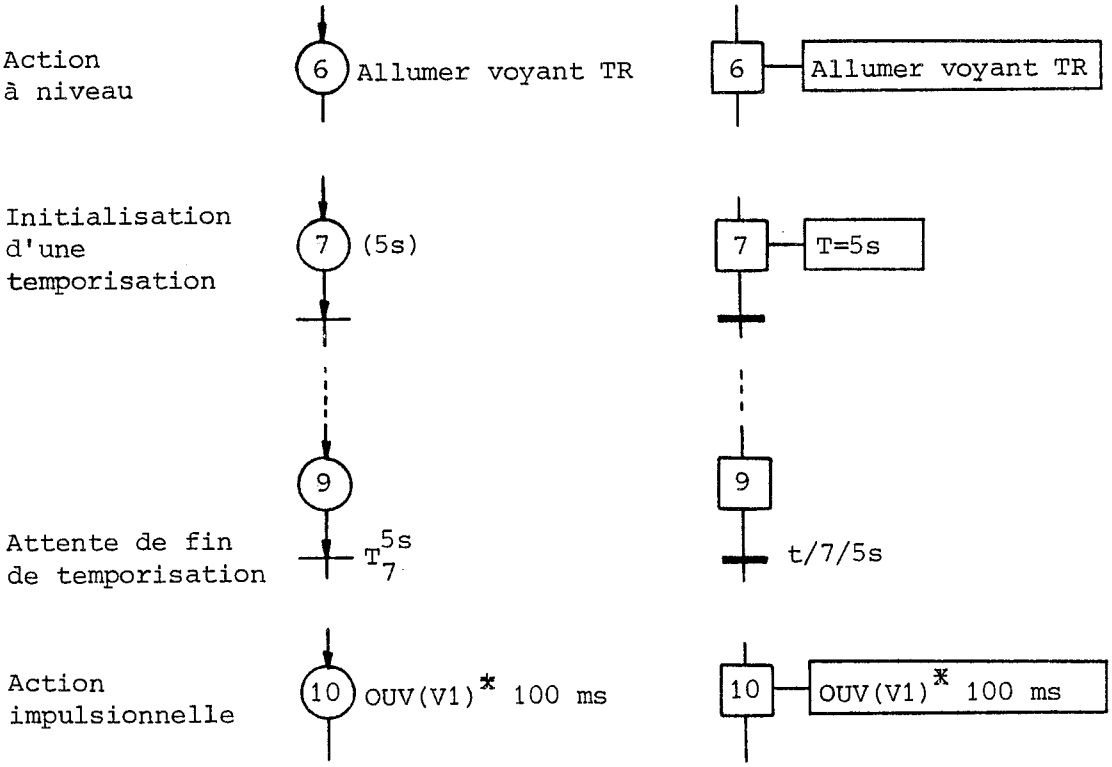
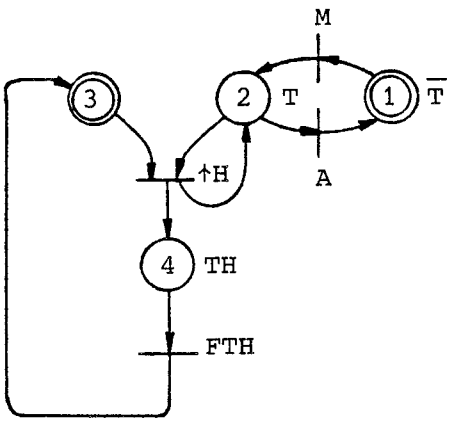
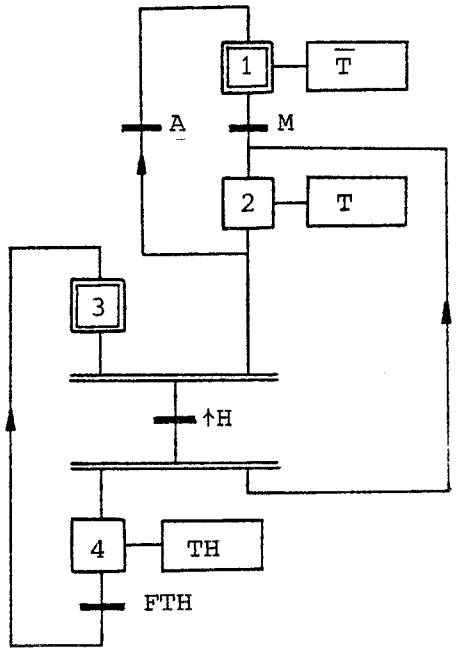


Figure 15. Formes de représentation du grafcet (début)



Exemple





Obligation d'indiquer l'orientation des arcs

Indication de l'orientation des arcs obligatoire uniquement dans le sens du bas vers le haut

Figure 15. Formes de représentation du grafcet (suite et fin)

CHAPITRE III

EXTENSION DU GRAFCET POUR LA SPÉCIFICATION DE SYSTÈMES TEMPS RÉEL COMPLEXES

I - INTRODUCTION 1

II - DES EXTENSIONS UTILES, RELATIVES AUX SPECIFICATIONS TEMPORELLES 2

II - 1. PARALLELISME OBLIGATOIRE ET PARALLELISME AUTORISE 2

II - 2. ACTION DIFFEREE ET FRANCHISSEMENT A ECHEANCE LIMITE 5

II - 3. INCIDENCES SUR LES REGLES D'INTERPRETATION DU GRAFCET 8

III - REPRESENTATION STRUCTUREE ET HIERARCHISEE 9

III - 1. NOTIONS DE MACROETAPE ET DE PSEUDO-MACROETAPE 9

III - 2. NOTION DE FONCTION 15

IV - CONCLUSIONS GENERALES SUR L'UTILISATION DU GRAFCET POUR LA SPECIFICATION D'UN AUTOMATISME COMPLEXE 19

I - INTRODUCTION

Les précisions que nous avons apportées concernant les actions de calcul constituent en soi une première extension du grafcet pour la prise en compte de l'aspect numérique. Le grafcet originel a été défini pour la représentation d'automatismes logiques relativement simples, pour lesquels les entrées/sorties sont du type tout ou rien. Grâce à cette extension, il devient possible de prendre en compte des entrées/sorties numériques et d'associer aux étapes (resp. aux transitions) des traitements (resp. des réceptivités) plus complexes.

Le grafcet originel prévoyait qu'une action impulsionnelle avait une durée déterminée, mais que cette action était interrompue si l'étape correspondante venait à être désactivée. La durée associée à l'action était censée être connue à l'avance (largeur d'impulsion pour la commande directe d'un opérateur dont on connaît les temps de réponse, temporisation, ...). Il est clair qu'il ne peut pas en être de même pour les actions de calcul.

Nous avons proposé qu'une action de calcul soit considérée comme étant une action impulsionnelle de durée aussi courte que nécessaire, et non interruptible. Cette notion pourrait être généralisée à des actions d'autre nature. On aurait alors deux types d'actions : 1) actions impulsionnelles non interruptibles (dont la fin conditionne tout nouveau franchissement simultané sur occurrence d'événement et dont la durée est suffisamment courte pour ne pas altérer les évolutions correctes entre situations ; 2) actions impulsionnelles à durée limitée (interruptibles). Cette généralisation nécessiterait l'introduction d'un nouveau symbole, par exemple la mise entre crochets d'une action non interruptible. Nous ne l'utiliserons pas ici, mais on peut imaginer des cas où elle pourrait être utile.

Dans ce chapitre, nous allons proposer d'autres extensions que l'on peut répartir en deux groupes :

1) Des extensions relatives aux spécifications temporelles : La première de ces extensions consiste à distinguer quand le parallélisme entre deux actions est obligatoire et quand il est simplement autorisé. Une deuxième extension concerne la définition des intervalles de temps dans lesquels une action impulsionnelle doit être effectuée ; on introduit les notions d'action différée et d'échéance limite.

2) Des extensions relatives à la représentation hiérarchisée des

spécifications. On introduit les notions de macroétape, de pseudo-macroétape et de fonction.

Comme nous le verrons, ces extensions ne modifient en rien les définitions initiales du grafcet mais les complètent, et ce principalement avec deux objectifs :

- apporter à la fois une plus grande souplesse et une plus grande rigueur à la spécification des contraintes d'ordonnancement entre actions et des contraintes de temps de réponse ; l'objectif étant d'éviter le plus possible les surspécifications et, donc, de laisser au concepteur le maximum de latitude dans les choix de la réalisation.
- étendre l'utilisation du grafcet à la spécification hiérarchisée des automatismes complexes.

II - DES EXTENSIONS UTILES, RELATIVES AUX SPECIFICATIONS TEMPORELLES [MOA.79] [MOA.81].

II - 1. PARALLELISME OBLIGATOIRE ET PARALLELISME AUTORISE

Le grafcet, tel qu'il est défini jusqu'ici, ne permet pas de distinguer entre la représentation d'un parallélisme obligatoire et celle d'un parallélisme simplement autorisé. Cette distinction est cependant importante à faire.

Exemple 1 :

« Une centrale de sécurité doit, sur détection du signal ALARME, déclencher deux actions parallèles :

. l'action 1 consiste à valider une sortie d'éclairage pendant 3 mn ;

. l'action 2 consiste à alimenter une sirène interne quatre fois pendant 45 sec. avec des intervalles de repos de 15 sec.

Le retour à l'état normal (avec suspension des actions en cours) s'effectue dès la disparition du signal ALARME».

Ce fonctionnement peut être représenté par le grafcet de la figure 16. Il est évident, sur cet exemple, que le parallélisme des deux actions d'éclairage et d'alimentation de sirène est obligatoire.

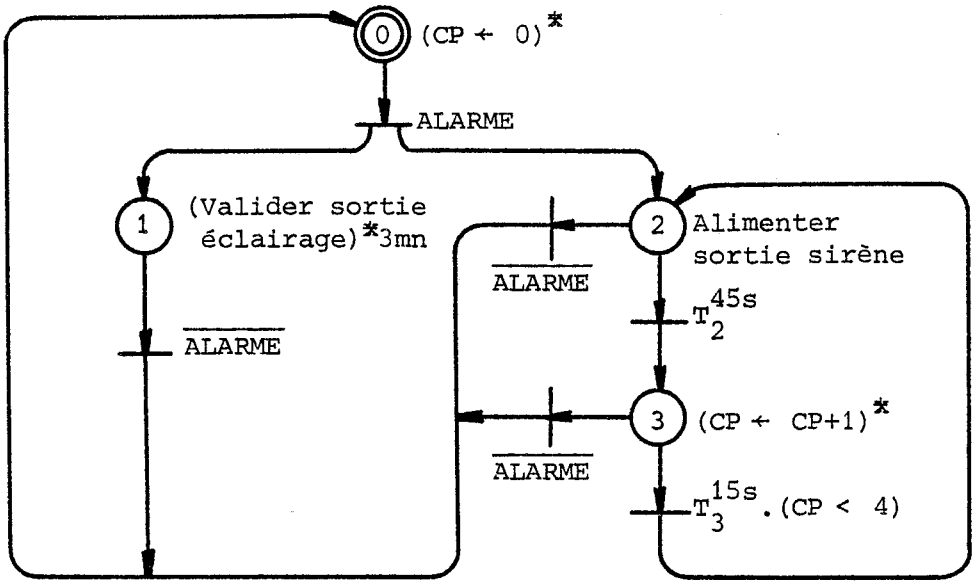


Figure 16. Centrale de sécurité

Exemple 2 :

« L'initialisation de la fonction de brassage de produit dans le ballon B10 nécessite en premier lieu que l'on s'assure que les électrovannes V5 et V6 soient fermées, sinon on doit commander leur fermeture. Ensuite, on commande l'ouverture de l'électrovanne V7. Puis, quand on a vérifié que l'électrovanne V7 s'est ouverte, on met en marche la pompe P1 ... ».

Les trois représentations (a), (b) et (c) données en figure 17 correspondent certes à des solutions de réalisation acceptables. Cependant, en tant que spécifications, elles comportent de "fausses" contraintes : les représentations (a) et (b) obligent à considérer les actions associées aux étapes 1 et 2 dans un ordre défini. De même, on peut comprendre de la représentation (c) que ces actions doivent être exécutées dans un parallélisme obligatoire ; alors qu'il s'agit clairement, dans cet exemple, d'un parallélisme simplement autorisé.

Une représentation fidèle de ces spécifications doit laisser le

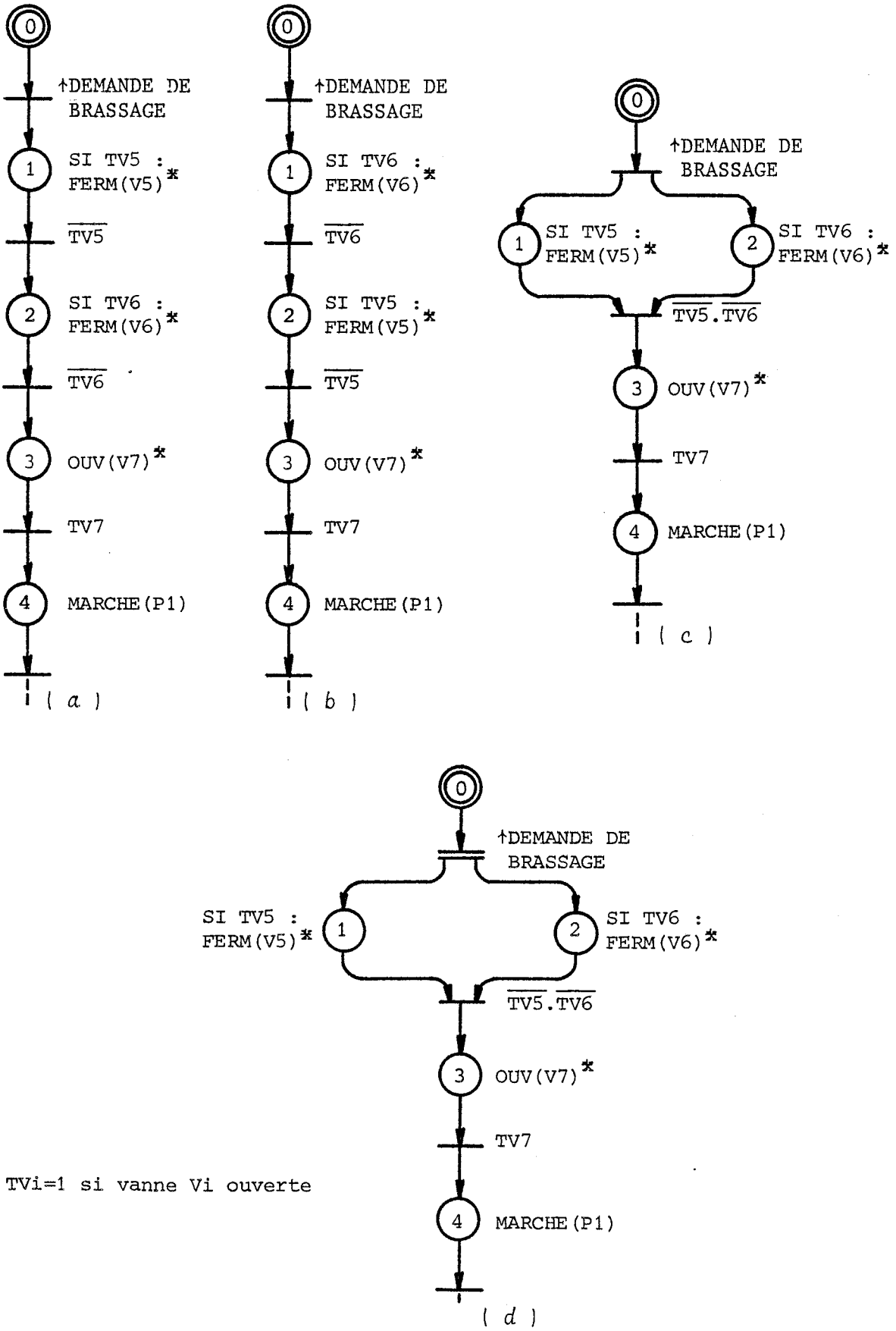


Figure 17. Parallélisme autorisé

choix ouvert au concepteur qui en décidera en fonction de l'analyse qu'il fera de l'ensemble des spécifications de l'automatisme, du matériel dont il dispose, de la facilité de mise en oeuvre, etc.

Pour distinguer explicitement les deux cas de parallélisme, nous suggérons de représenter une transition par un double trait pour préciser que le parallélisme des actions associées aux étapes en aval est non obligatoire.

On obtient pour l'exemple de la fonction de brassage la représentation donnée par la figure 17-d. Telle quelle, cette représentation n'est pas directement "réalisable". Comme nous l'avons dit, le concepteur doit d'abord choisir une des trois solutions possibles.

La précision du parallélisme non obligatoire ne concerne évidemment que les actions impulsionnelles. Dans le cas de cet exemple, si une action à niveau est demandée en parallèle aux actions de fermeture des électrovannes, cette action devra être associée aux deux étapes consécutives 1 et 2 des grafcets (a) et (b) de la figure 17.

II - 2. ACTION DIFFEREE ET FRANCHISSEMENT A ECHEANCE LIMITE

D'une façon générale, la représentation à l'aide du grafcet fixe pour chaque action associée à une étape des échéances strictes de début et de fin ; ces échéances étant déterminées, directement ou indirectement, par le comportement de l'environnement externe.

Ainsi, dans le cas de l'exemple 1, l'action d'éclairage associée à l'étape 1 doit commencer dès la détection du signal ALARME et se terminer 3 mn après ou sur disparition du signal ALARME. De même, l'action d'alimentation de la sirène interne doit être déclenchée dès la détection du signal ALARME pour une première période de 45 sec. sauf disparition du signal ALARME....

Sur le plan de la réalisation effective, ces échéances sont assurées moyennant une tolérance de temps de réponse que le cahier des charges peut préciser quantitativement. Le plus souvent, pour les automatismes logiques simples, cette tolérance n'est pas précisée ; elle est déterminée indirectement par le choix du matériel d'implantation : temps de basculement d'un bistable, durée maximum du cycle de traitement d'un automate programmable, ...

Dans le cas d'un automatisme complexe destiné à être implanté sur un matériel informatique important, cette tolérance est nécessairement plus large. Cependant, il n'est pas nécessaire qu'elle soit uniforme : certains événements peuvent nécessiter un temps de réponse très rapide alors que d'autres actions peuvent être différées pourvu qu'elles soient exécutées avant une échéance limite. Le fait de préciser les importances relatives de ces tolérances permet de donner aux spécifications une plus grande rigueur et au concepteur une plus grande latitude dans les choix de la réalisation.

Il peut aussi arriver que le rédacteur du cahier des charges ne soit pas en mesure de spécifier avec précision les échéances de début et de fin de chaque action sans être amené à anticiper sur la conception de l'automatisme et, par conséquent, à introduire de fausses contraintes. L'exemple suivant illustre une telle situation.

Exemple 3 :

« L'une des fonctions d'un automatisme consiste à effectuer une mesure sur le procédé, filtrer cette mesure et l'afficher sur un écran. La seule contrainte que l'on impose est que la valeur affichée soit constamment actualisée avec des mesures vieilles d'au plus une seconde ».

Il est évident :

1°) que l'automatisme doit être capable d'effectuer les trois opérations de mesure, filtrage et affichage en moins d'une seconde ;

2°) que plusieurs solutions de conception sont possibles, dont les extrêmes consistent soit à synchroniser le départ de la séquence des trois opérations sur les impulsions d'une horloge 1Hz, soit à boucler en permanence sur l'exécution des trois opérations.

3°) que la seule représentation qui permet de rester neutre vis-à-vis de la conception est celle qui se limiterait à préciser l'échéance à partir de laquelle l'exécution de la séquence peut commencer et l'échéance avant laquelle cette exécution doit se terminer.

Nous proposons, à ces fins, d'introduire deux nouveaux symboles : le symbole Δ à placer à côté de l'astérisque (*) quand on veut indiquer qu'une action impulsionnelle peut être différée, et la notation L_{α}^{tmax} à placer à côté d'une transition pour indiquer une échéance limite de

franchissement (de façon duale à la notation T_{α}^{tmin} déjà prévue dans la définition initiale du grafcet pour spécifier une temporisation minimum avant le franchissement).

On obtient, pour l'exemple 3, le grafcet de la figure 18.

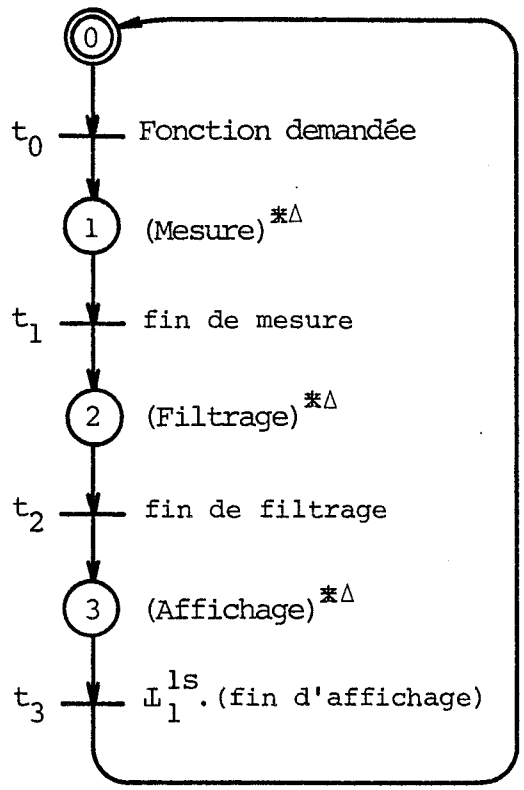


Figure 18. Actions différées et franchissement à échéances limites

Sur ce grafcet, l'état au repos est représenté par l'étape 0 active. Dès que la fonction est demandée, on passe à l'étape 1. On a alors l'action impulsionnelle (mesure)$^{*\Delta}$ à effectuer, mais le symbole Δ indique que cette action peut être différée. Quand la mesure aura été faite, on passera à l'étape 2 à laquelle est associée l'action (filtrage)$^{*\Delta}$ qui peut également être différée... . L'échéance limite est indiquée sur la transition t_3 : la notation $\mathbb{L}_1^{ls}$ indique que cette transition doit être franchie au plus tard 1 sec. après l'activation de l'étape 1 ; on spécifie ainsi qu'une séquence complète comportant une mesure, un filtrage de cette

mesure et un affichage doit être effectuée en un temps inférieur à 1 sec. Le franchissement de t_3 est suivi immédiatement de celui de t_0 tant que la demande de la fonction est maintenue.

La notion d'échéance limite, dans le cas général, sera précisée au paragraphe suivant.

II - 3. INCIDENCES SUR LES REGLES D'INTERPRETATION DU GRAFCET

Il est clair que les extensions qui sont proposées ne sont pas en contradiction avec les règles du grafcet précédemment définies. Il faut cependant préciser les interprétations consécutives à ces extensions.

1) Une transition représentée par un double trait indique que le parallélisme des actions impulsionnelles associées aux étapes qui suivent est autorisé, mais non obligatoire.

2) Une action impulsionnelle marquée de Δ peut être différée (les actions à niveau ne peuvent pas être différées).

3) La notation L_{α}^{tmax} placée à côté d'une transition traduit la contrainte de fonctionnement suivante : < Si δ est la situation du grafcet à l'instant d'une nouvelle activation de l'étape α et si l'évolution des situations à partir de δ doit conduire (tôt ou tard) au franchissement de cette transition sans passer par une nouvelle activation de l'étape α, alors ce franchissement doit avoir lieu au bout d'un temps maximum inférieur à tmax > .

Dans l'exemple de la figure 19, toute nouvelle activation de l'étape 1 doit conduire soit au franchissement de t_5 au bout d'un temps inférieur à 2 sec., soit au franchissement de t_6 au bout d'un temps inférieur à 5 sec.

L'indication d'une échéance limite ne correspond pas à une réceptivité et ne modifie en rien les règles de franchissement de la transition. Sa présence dans des réceptivités s'impose si l'on utilise les concepts de parallélisme autorisé et d'actions différées qui rendent l'interprétation du grafcet non déterministe. L'indication des échéances limites sert alors à restreindre ce non-déterminisme et, en particulier, à éviter que les actions différées ne le soient indéfiniment.

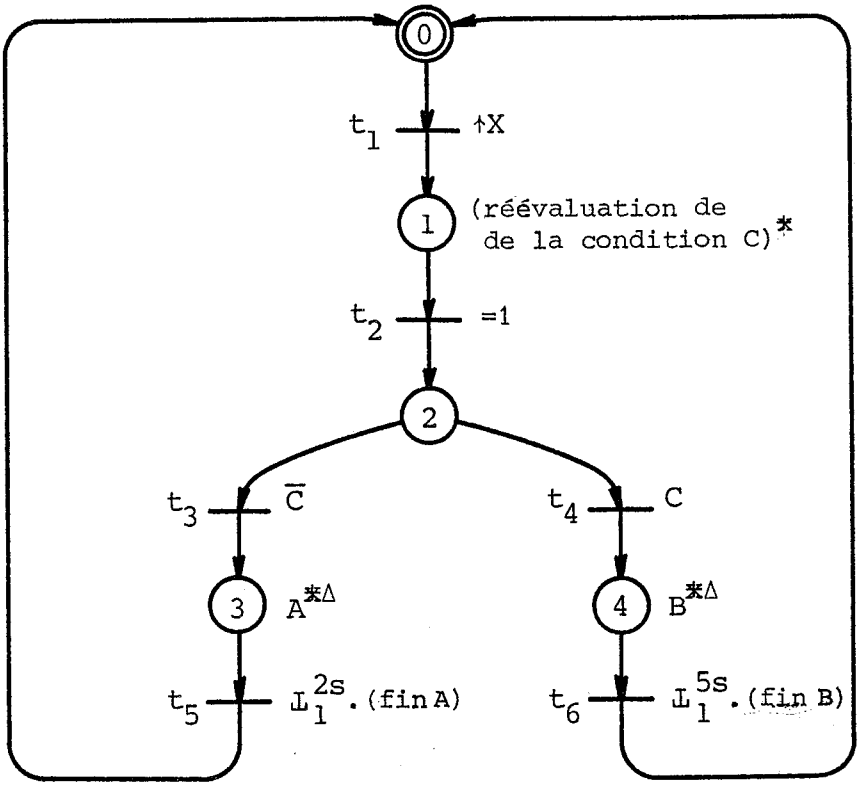


Figure 19. Echéances mutuellement exclusives

III - REPRESENTATION STRUCTUREE ET HIERARCHISEE

Nous allons introduire dans ce paragraphe trois notions importantes et nettement distinctes. Les deux premières, celles de macroétape et de pseudo-macroétape, sont relatives à une description progressive. La troisième, celle de fonction, est relative à une hiérarchie entre automates.

III - 1. NOTIONS DE MACROETAPE ET DE PSEUDO-MACROETAPE

On peut aborder la représentation progressive d'un fonctionnement complexe de plusieurs façons :

- Une approche "libre" consiste à donner une suite de représentations complètes de ce fonctionnement à des niveaux de description progressifs.
- Une approche "totalement structurée" consiste à donner une représentation initiale qui situe les "phases principales" de ce fonctionnement, puis à développer de façon séparée la description de chacune de ces phases ; le processus pouvant, éventuellement, être utilisé de façon récursive.

Dans l'une ou l'autre des deux approches, il est important de bien

mettre en évidence le "fil conducteur" qui permette au lecteur du cahier des charges de suivre avec méthode la progression des spécifications.

En pratique, on utilise couramment les deux approches simultanément. En effet, il est souvent utile (sinon nécessaire) de commencer par une représentation synthétique qui situe les principales phases du fonctionnement à décrire. En revanche, il n'est pas toujours possible de procéder, dès cette première représentation, à une décomposition de ce fonctionnement en phases disjointes de telle sorte que chacune de ces phases, représentée à ce niveau par l'équivalent d'un noeud étape, puisse être détaillée ultérieurement de façon indépendante des autres.

On peut donc s'attendre à trouver au niveau d'une telle représentation trois utilisations différentes des noeuds étapes :

. Un noeud étape qui résume un sous-grafcet dont la description détaillée peut être donnée de façon séparée. Le fait de pouvoir séparer cette description apporte une clarté certaine à la représentation. On dira que ce noeud représente une macroétape.

. Un noeud étape qui résume de même un sous-grafcet mais dont la description détaillée ne peut pas être isolée du reste de la représentation. Il faut, pour donner les détails de ce noeud étape, reprendre l'ensemble de la représentation à un niveau de description plus fin. On dira que ce noeud représente une pseudo-macroétape.

. Un noeud étape simple auquel peuvent être associées des actions complètement définies.

Le "fil conducteur" dont nous parlions ci-dessus consiste à donner aux noeuds macroétapes et pseudo-macroétapes des représentations différentes de celles d'une étape simple, de telle sorte que leur présence dans un grafcet renseigne immédiatement le lecteur si une macroétape est détaillée par un sous-grafcet séparé (et dans ce cas, lequel ?) ou si une pseudo-macroétape est amenée à être développée dans une représentation ultérieure plus fine.

Nous allons proposer des conventions précises d'utilisation de macroétapes et de pseudo-macroétapes. Nous illustrerons ces conventions sur des exemples.

1.1. Macroétape

Au noeud marqué M6 dans le grafcet principal de la figure 20-a, on doit substituer le sous-grafcet de la figure 20-b. Le noeud M6 représente une macroétape. Le sous-grafcet de la figure 20-b est appelé grafcet de la macroétape. On obtient le grafcet complet de la figure 20-c.

Les règles d'utilisation sont les suivantes :

1) On représente une macroétape par un cercle inscrit dans un losange. La nom de la macroétape est porté à l'intérieur du cercle.

2) Un grafcet de macroétape a une seule étape de début appelée étape d'entrée et une seule étape de fin appelée étape de sortie. Pour repérer ce sous-grafcet, on fait précéder son étape d'entrée et suivre son étape de sortie par un triangle dans lequel figure le nom de la macroétape (voir figure 20-b).

3) La substitution se fait de la façon suivante :

A la macroétape se substitue le sous-grafcet. A tous les arcs arrivant sur la macroétape correspondent des arcs arrivant sur l'étape d'entrée. A tous les arcs sortant de la macroétape correspondent des arcs partant de l'étape de sortie.

4) Plusieurs macroétapes peuvent avoir le même grafcet de macroétape qui peut alors n'être décrit qu'une seule fois (figures 20-d et 20-e). (Il peut être utile dans ce cas de trouver une notation qui permette de donner à ces macroétapes des noms différents, en y associant par exemple des indices ou toute autre marque de distinction).

Remarques :

i) Une macroétape peut avoir plusieurs transitions d'entrée et/ou de sortie (voir figures 20-f et 20-g).

ii) Une transition d'entrée (resp. de sortie) d'une macroétape peut être transition d'entrée (resp. de sortie) d'une autre étape, ou macroétape (voir figure 20-d).

iii) On peut imaginer que le grafcet de macroétape de la figure 20-b soit tel qu'il n'y ait pas d'action associée à l'étape e et que r_a et r_b soient des conditions toujours vérifiées (notées =1). Dans ce cas, la transition de réceptivité r_1 (figure 20-c) pourrait conduire directement

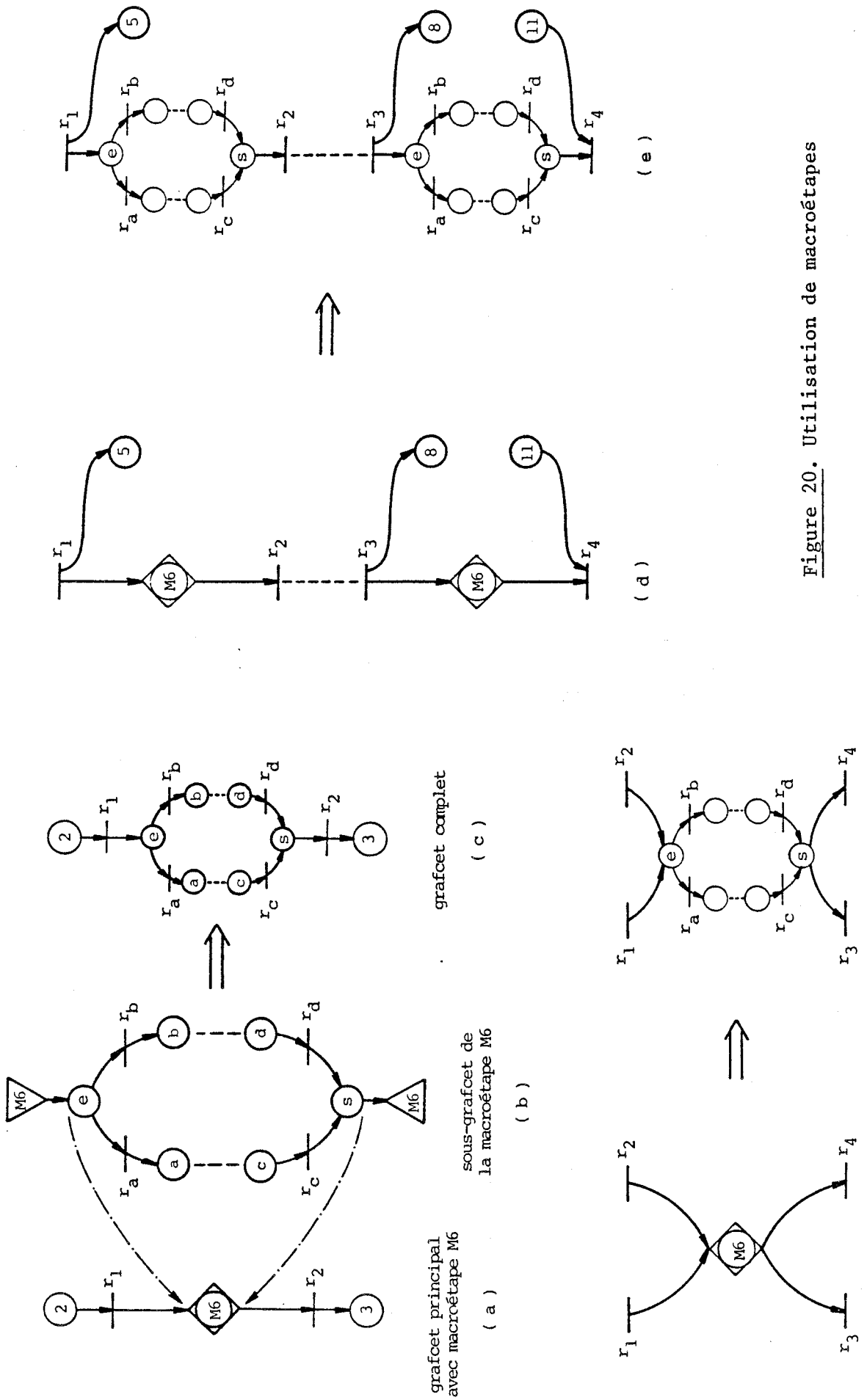


Figure 20. Utilisation de macroétapes

aux étapes a et b. Nous pensons que cela doit éventuellement relever d'une simplification du grafcet obtenu, mais que l'on doit scrupuleusement respecter la 2ème règle d'utilisation, étape d'entrée unique, pour éviter des erreurs.

iv) Rien n'empêche que l'étape d'entrée d'un grafcet de macroétape puisse elle-même servir d'étape de sortie.

v) L'utilisation de macroétapes n'est qu'un artifice de simplification de la représentation. La question de savoir si une macroétape qui apparaît plusieurs fois donnera lieu ou non à la réalisation d'un automate fonctionnant séparément est laissée au concepteur.

III - 1.2. Pseudo-macroétape

Un exemple de description de "niveau 0" (pour signifier un niveau synthétique) des séquences de mise en marche/arrêt d'une machine M est donné dans la figure 21, en traits continus. Les losanges, marqués P_d et P_a sur cet exemple, représentent des pseudo-macroétapes. Ce grafcet de niveau 0 est incomplet, beaucoup de détails manquent, mais le lecteur comprend aisément le "squelette" du grafcet final ainsi résumé par ce "grafcet de niveau 0".

Une pseudo-macroétape est telle que : 1) elle représente un sous-grafcet ; 2) tous les arcs entrants n'arrivent pas nécessairement sur une même étape d'entrée, de même que tous les arcs sortants ne partent pas nécessairement d'une même étape de sortie ; 3) tous les arcs entrants et sortants ne sont pas nécessairement mentionnés au niveau de la représentation où apparaît la pseudo-macroétape.

Macroétapes et pseudo-macroétapes diffèrent du fait des points 2 et 3.

Certaines précisions, qui ne présentent guère d'intérêt au niveau 0 de description, sont apportées ensuite pour obtenir le grafcet de la figure 22. On remarquera que le test < reprise demandée ? > n'est pas effectué à la fin de la pseudo-macroétape d'arrêt. Il s'ensuit que l'on peut sortir de la phase d'arrêt tout de suite après la première étape (étape 9), et que l'on ne recommencera pas toute la phase de démarrage si cette reprise est demandée (reprise à l'étape 5). De même, on remarquera que l'étape 8 et les transitions correspondantes ont été ajoutées.

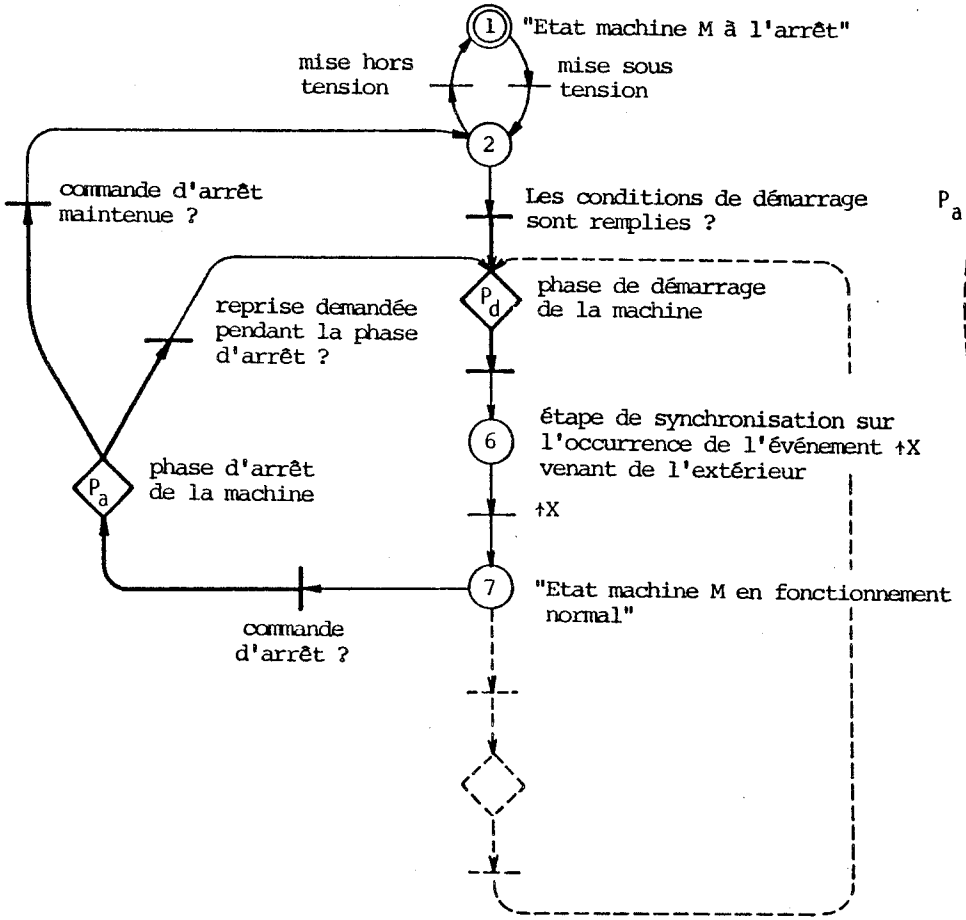


Figure 21. Grafcet de niveau 0, avec des pseudo-macroétapes

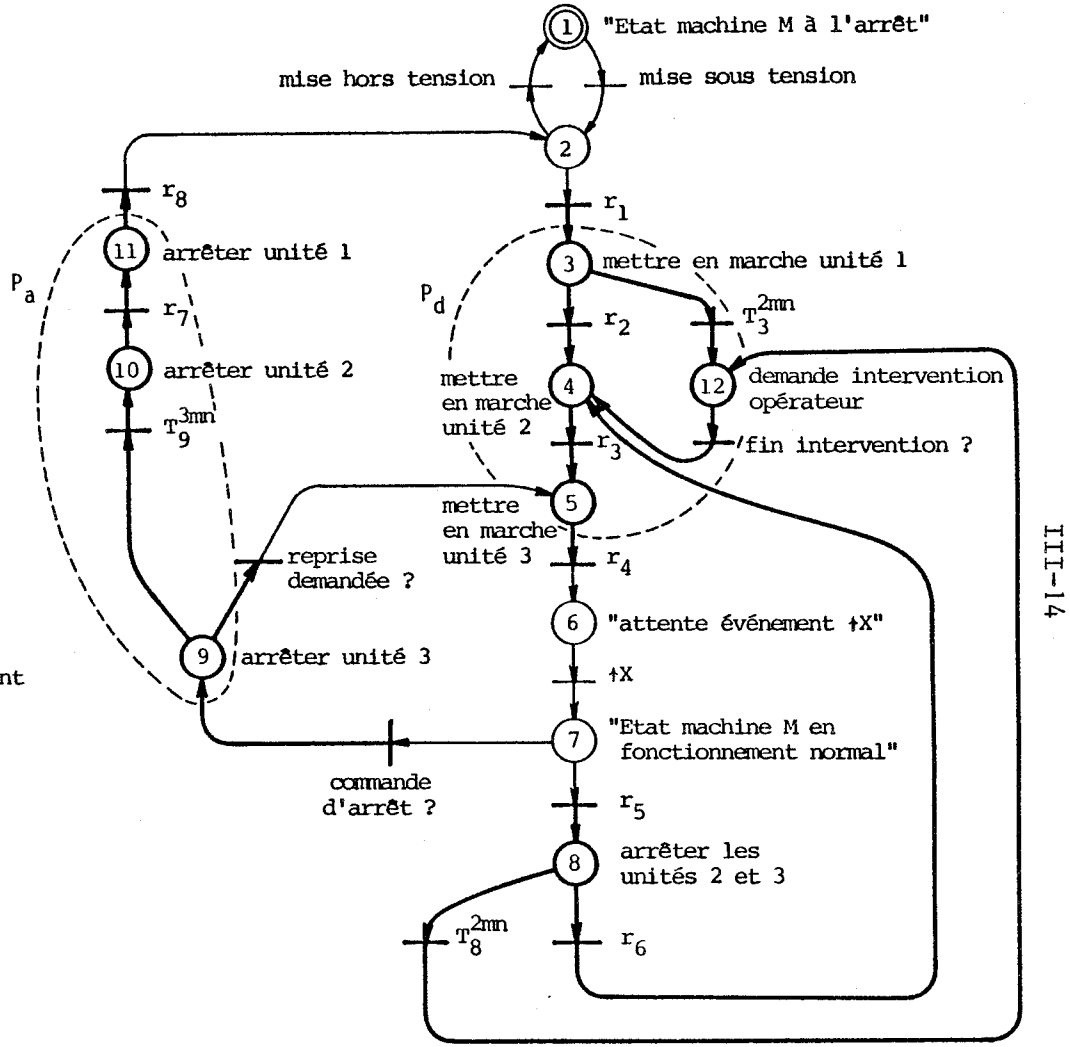


Figure 22. Grafcet obtenu à partir de la figure 21 en développant les pseudo-macroétapes

On peut considérer que cette étape de surveillance et son rebouclage sur la pseudo-macroétape P_d ne sont pas nécessaires au niveau 0.

Une autre personne faisant la description aurait pu inclure l'étape 8 dès la figure 21 (ce qui est représenté en traits pointillés). Ce qu'il faut retenir, c'est que la présence d'une pseudo-macroétape signifie que le grafcet est incomplet, et qu'il doit être repris dans son ensemble, dans une représentation plus fine qui détaille la pseudo-macroétape.

Il est clair, dans le cas de l'exemple traité, que toute autre décomposition qui chercherait à remplacer les deux pseudo-macroétapes par des macroétapes trahirait la démarche naturelle de spécification progressive dans laquelle il y a tout intérêt à maintenir pour chaque "noeud étape" (macro, pseudo-macro, ou étape simple) une interprétation fonctionnelle cohérente et nette.

III - 2. NOTION DE FONCTION

Comme nous l'avons annoncé, la notion de fonction est relative à une hiérarchie entre automates. Pour l'introduire, nous allons considérer un exemple simple : une électrovanne qui assure l'ouverture ou la fermeture d'une canalisation et qui peut être commandée par un ou plusieurs grafcets.

En plusieurs points du fonctionnement décrit par ces grafcets, on veut pouvoir commander l'ouverture ou la fermeture de cette électrovanne, ou s'assurer de son état "ouvert" ou "fermé". L'ouverture (resp. la fermeture) de l'électrovanne n'est pas instantanée et nécessite un temps minimum τ_i. Ainsi, à l'instant où l'on commande l'ouverture, on peut avoir les trois situations suivantes :

. l'électrovanne est fermée : la commande aura pour effet dans ce cas de déclencher la fonction d'ouverture ;

. la fonction d'ouverture est en cours parce que une autre commande d'ouverture a été demandée quelques instants auparavant : dans ce cas, la nouvelle commande n'aura pour effet que de confirmer la commande précédente ;

. l'électrovanne est déjà à l'état ouvert : la nouvelle commande d'ouverture n'aura alors aucun effet réel.

Pour des raisons de sécurité, on peut vouloir soumettre l'ouverture de l'électrovanne à des conditions portant sur l'état du procédé (ou du

système global). On spécifie alors un ensemble d'états tels que :

. tant que le procédé est dans l'un de ces états, on doit inhiber toute commande d'ouverture de l'électrovanne ;

. si pendant que l'électrovanne est ouverte, le procédé atteint l'un de ces états, on doit commander systématiquement la fermeture de l'électrovanne et/ou déclencher une alarme.

Cet exemple illustre le cas d'une fonction simple (ouverture de canalisation), qui est exécutée par un organe technologique —une électrovanne— que l'automatisme peut commander alors qu'elle est fermée (état au repos de la fonction) ou en cours d'ouverture (fonction en cours de mise en marche) ou déjà ouverte (fonction en marche). Cette fonction est soumise à des sécurités dépendant des états du procédé (ou de l'ensemble du système), qui peuvent interdire l'ouverture de l'électrovanne (fonction non autorisée) ou obliger sa fermeture (arrêt de la fonction) avec, éventuellement, déclenchement d'une alarme.

Dans ce cas simple, la commande de mise en marche ou d'arrêt de la fonction se réduit, du côté de l'automatisme, à l'envoi d'un ordre d'ouverture ou de fermeture.

Dans le cas d'un automatisme complexe, on peut trouver des fonctions dont la mise en marche ou l'arrêt nécessite toute une séquence de commandes que l'automatisme est obligé de gérer par lui-même en surveillant les sécurités qui s'imposent. Par exemple, dans le cas d'un procédé de distillation, on peut trouver les fonctions telles que "brassage produit dans un ballon","alimentation colonne", "régulation débit", etc.

L'exemple précisément de la fonction de brassage de produit dans un ballon a déjà été introduit dans le paragraphe II - 1 (figure 17). On peut faire pour cette fonction le même raisonnement que nous venons de faire pour le cas de l'électrovanne : On peut trouver la commande de mise en marche ou d'arrêt de cette fonction en plusieurs endroits différents du fonctionnement décrit et pour des raisons diverses (par exemple, on peut vouloir effectuer le brassage du produit afin d'homogénéiser sa concentration avant de prélever un échantillon de test, ou utiliser le circuit de brassage pour renvoyer le produit vers un autre ballon de stock). Aussi, de la même manière que pour l'électrovanne, on peut vouloir tester l'état "en marche" ou "à l'arrêt" de la fonction. La mise en marche (ou l'arrêt)

nécessitant toute une séquence de commandes,on est amené à distinguer les états "en marche", "en cours de mise en marche", "à l'arrêt" ou "en cours d'arrêt". Enfin, la fonction de brassage peut être inhibée pour des raisons de sécurité : par exemple, on interdit la mise en marche de la fonction de brassage si un niveau minimum de produit n'est pas présent, ou on oblige l'arrêt du brassage avec signalisation d'alarme si le niveau vient à baisser en dessous d'un seuil limite alors que le brassage est en cours.

Nous proposons de donner à la représentation des séquences de mise en marche et d'arrêt d'une fonction des conventions particulières illustrées par la figure 23 : La partie (a), délimitée par le contour en trait pointillé, précise les conditions qui autorisent ou inhibent l'exécution de la fonction ; on peut y inclure, éventuellement, une représentation des alarmes se rapportant directement à l'exécution de cette fonction (c'est-à-dire que la fonction gère elle-même les éventuelles contradictions entre l'état du procédé et les ordres des différents grafcets). La partie (b) explicite les étapes de mise en marche de la fonction. La partie (c) explicite les étapes de la phase d'arrêt.

Un grafcet de commande principal qui utilise cette fonction peut être, par exemple, celui de la figure 24.

Remarques :

1) La différence est nette entre une macroétape et une fonction. Chaque nouvelle activation d'une macroétape oblige la réinitialisation de la séquence complète définie par le grafcet de la macroétape. Par contre, une commande de mise en marche d'une fonction n'a d'effet d'initialisation que si la fonction se trouve à l'état d'arrêt ; sinon, elle ne fait que maintenir l'état de marche ou confirmer le mise en marche en cours.

2) L'ordre que constitue la commande "Marche fonction f_i" est interprété comme une action à niveau ; cet ordre reste effectif tant que l'étape à laquelle il est associé est active. Il en est de même pour la commande d'arrêt.

3) Dans le cas où des ordres de mise en marche et d'arrêt de la même fonction peuvent être effectifs simultanément, les spécifications doivent préciser lequel des deux ordres est prépondérant (la même fonction pouvant être utilisée par plusieurs grafcets "principaux", c-à-d. hiérarchiquement

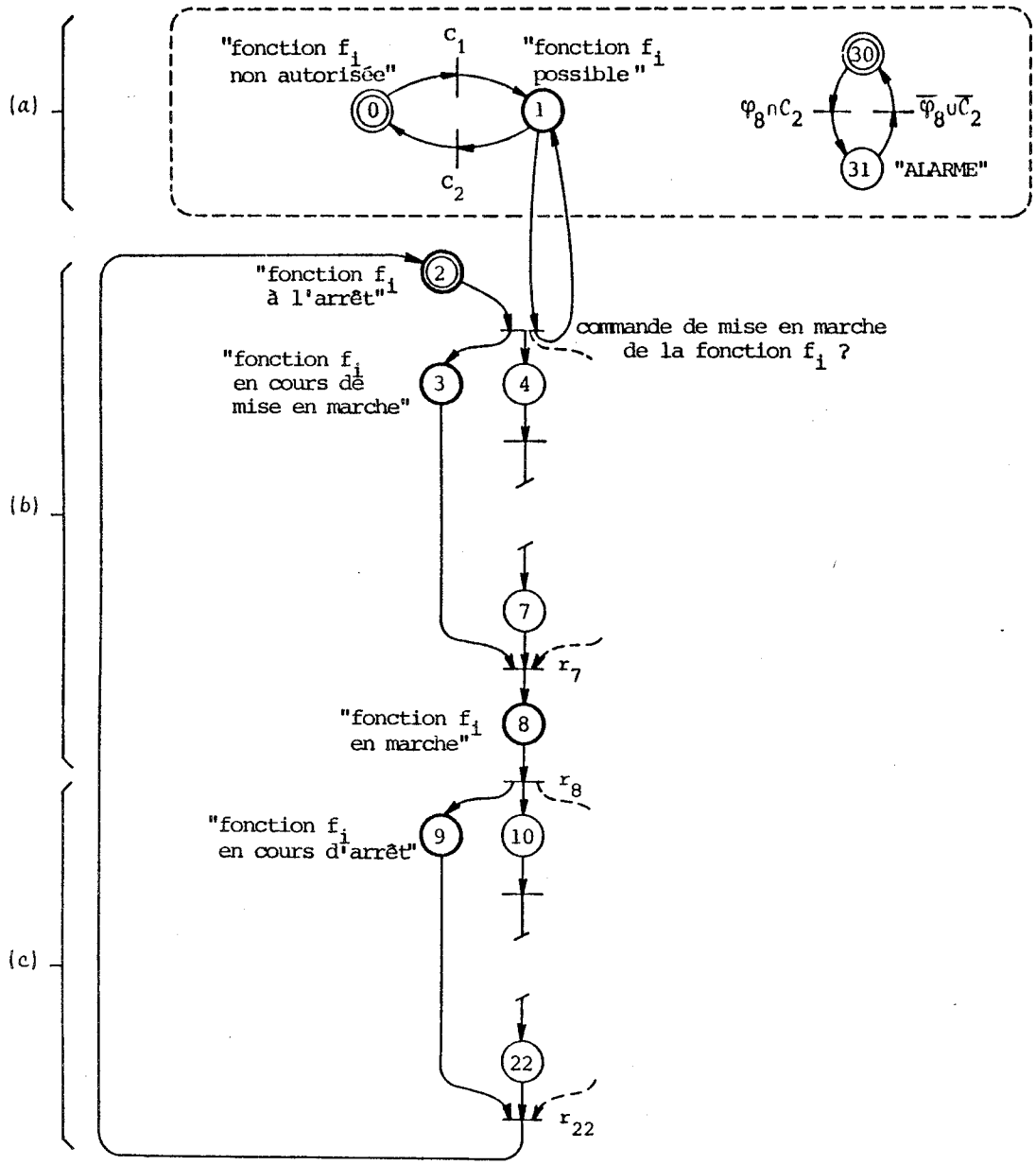


Figure 23. Représentation d'une fonction

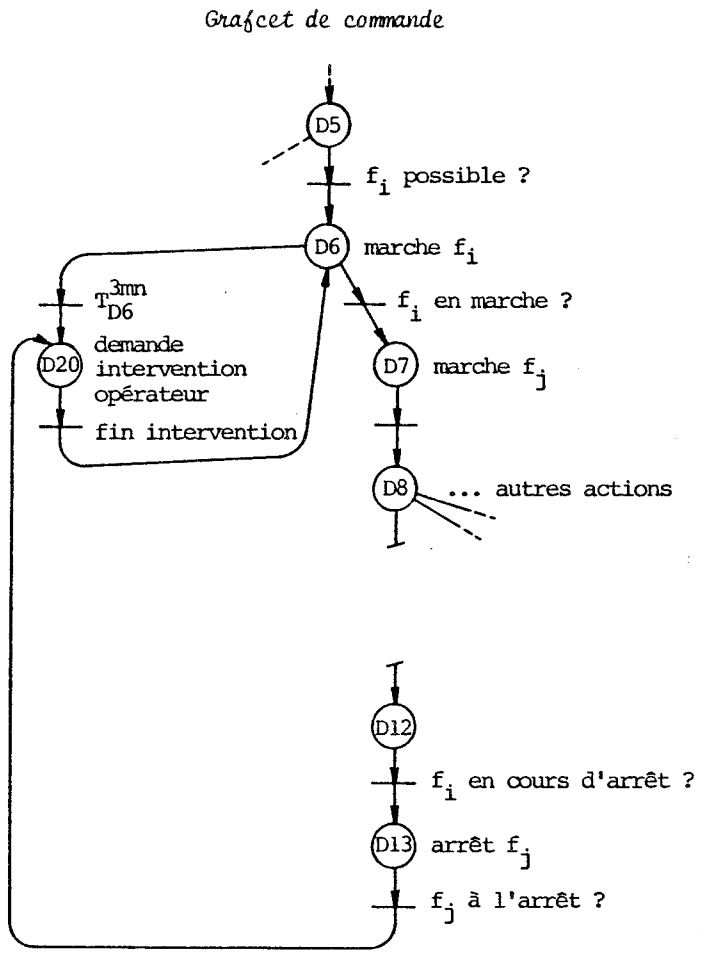


Figure 24. Exemple de grafcet principal utilisant les fonctions f_i et f_j

supérieurs à la fonction).

4) En principe, ni la séquence (b) de mise en marche de la fonction, ni la séquence (c) d'arrêt ne sont interruptibles. Par conséquent, une commande d'arrêt ne peut prendre effet qu'une fois la séquence de mise en marche achevée. Inversement, une commande de mise en marche ne peut prendre effet que si la fonction est à l'arrêt. Si telle n'est pas l'interprétation souhaitée, la description de la fonction doit indiquer explicitement les points d'interruptibilité éventuels des deux séquences.

5) Enfin, on peut imaginer des fonctions plus complexes ; par exemple, des fonctions telles que la mise en marche (ou l'arrêt) comporte plusieurs phases, ou des fonctions dont la mise en marche dépend du mode de fonctionnement en cours : "manuel", "automatique", etc.

IV - CONCLUSIONS GENERALES SUR L'UTILISATION DU GRAFCET POUR LA SPECIFICATION D'UN AUTOMATISME COMPLEXE

La clarté du cahier des charges d'un automatisme complexe repose autant :

. sur une structuration rationnelle du contenu : séparation de la description du procédé à commander des spécifications de l'automatisme, distinction des divers types de spécifications (fonctionnelles, opérationnelles et technologiques), classification des fonctions s'apparentant à un même type de spécifications, ... ;

. sur une hiérarchisation de la description, tant du procédé que de l'automatisme, de façon à rendre le cahier des charges plus pénétrable ;

. sur l'adaptation des outils de représentation à cette description progressive et hiérarchisée.

Trois composantes que nous avons cherché à analyser le long de ces trois premiers chapitres, en proposant tout d'abord une approche de structuration du contenu du cahier des charges, puis en étudiant les critères et outils qui permettent d'assurer une représentation rigoureuse et hiérarchisée de ce contenu.

Cette approche a déjà été expérimentée sur un certain nombre d'exemples réels dont trois présentaient une complexité significative : Le système de commande d'un procédé de distillation [MOA.79] [DAV.79], le système de commande décentralisée d'un atelier flexible de masticage

[CAV.81], et l'automatisme de commande d'une machine d'assemblage [BUG.81]. A titre d'illustration, nous donnons en annexe de larges extraits du cahier des charges du système de commande du procédé de distillation. (Le document complet comporte 130 pages).

Il nous paraît utile d'insister sur les points suivants :

1) La généralité des concepts offerts par le grafcet permet son utilisation aussi bien en tant qu'outil de spécification de cahiers des charges qu'en tant qu'outil de conception et de spécification de solutions. Cependant, les objectifs et la manière dont le grafcet doit être utilisé dans l'un ou l'autre des deux cas sont différents.

La spécification d'un cahier des charges n'a pas besoin d'être réalisable telle quelle. Elle n'a pas besoin non plus d'être optimisée. Elle doit surtout être claire et se limiter à décrire le plus précisément possible les vraies contraintes, en évitant les surspécifications et en laissant au concepteur le maximum de latitude pour décider des solutions de réalisation à adopter.

2) La redondance dans les spécifications n'est pas un mal en soi dans la mesure où elle facilite la présentation de ces spécifications tout en aidant à mieux les comprendre. Chercher à trop optimiser les spécifications conduit souvent à rendre leur compréhension plus difficile et moins sûre.

Il est bien entendu que la façon dont les spécifications sont présentées ne présume en rien de la façon dont l'automatisme doit être conçu. De même, les variables intermédiaires et les divers opérateurs de temporisation et de calcul utilisés dans les spécifications n'ont pas à correspondre (tels quels) à des variables et des opérateurs de la réalisation.

3) L'interface d'un fonctionnement décrit par un grafcet peut comprendre :

- des entrées/sorties liées à l'interface de commande et de contrôle du procédé ;
- des entrées/sorties liées à l'interface utilisateur ;
- des informations d'échange avec d'autres grafcets (variables partagées, étapes dont l'état est testé par d'autres grafcets, ...).

L'expérience montre que le fait de résumer cet interface à côté du grafcet (en indiquant les origines des entrées et les destinations des

sorties et en rappelant, éventuellement, la signification des mnémoniques utilisés) est d'une aide appréciable pour la lecture et l'analyse des spécifications.

4) Il peut arriver que l'on ait à reproduire un même détail en plusieurs points du même grafcet. C'est le cas, par exemple, quand on veut spécifier que l'occurrence d'un événement particulier doit placer immédiatement le grafcet dans une situation donnée quelle que soit la situation en cours, ou quand on veut tenir compte des modalités d'un changement de mode qui peut être opéré à tout moment du fonctionnement ...

Dans ces cas, la représentation peut devenir trop touffue et il serait alors préférable de ne retenir sur le grafcet que les détails essentiels et de compléter cette représentation par un commentaire séparé en langue naturelle.

5) Le grafcet peut être utilisé lors même de la description du procédé ; en particulier, pour représenter les fonctions assurées localement au niveau des unités du procédé.

Exemple : nous avons étudié, dans le pargraphe II - 1 de ce chapitre, la fonction de brassage du produit dans un ballon de stock. Si le schéma d'exécution de cette fonction ainsi que les précautions de sécurité à prendre sont impératifs quel que soit le système de commande, il peut être intéressant de les donner lors même de la description du fonctionnement de l'unité qui comprend ce ballon de stock. On en tire le double avantage de mieux faire apparaître le rôle que remplit cette unité dans le fonctionnement global du procédé tout en préparant la phase de spécification de l'automatisme.

6) Le dernier point concerne la représentation des sécurités fonctionnelles.

D'une façon générale, une sécurité fonctionnelle peut être représentée par une condition qui assujettit l'exécution effective d'une action de commande (rendre conditionnelle une action de commande associée à une étape). Dans certains cas, il peut être intéressant de spécifier une sécurité fonctionnelle plutôt sous la forme d'un "automate de surveillance" lui-même représenté par un grafcet.

Reprenons encore une fois l'exemple de la fonction de brassage dans

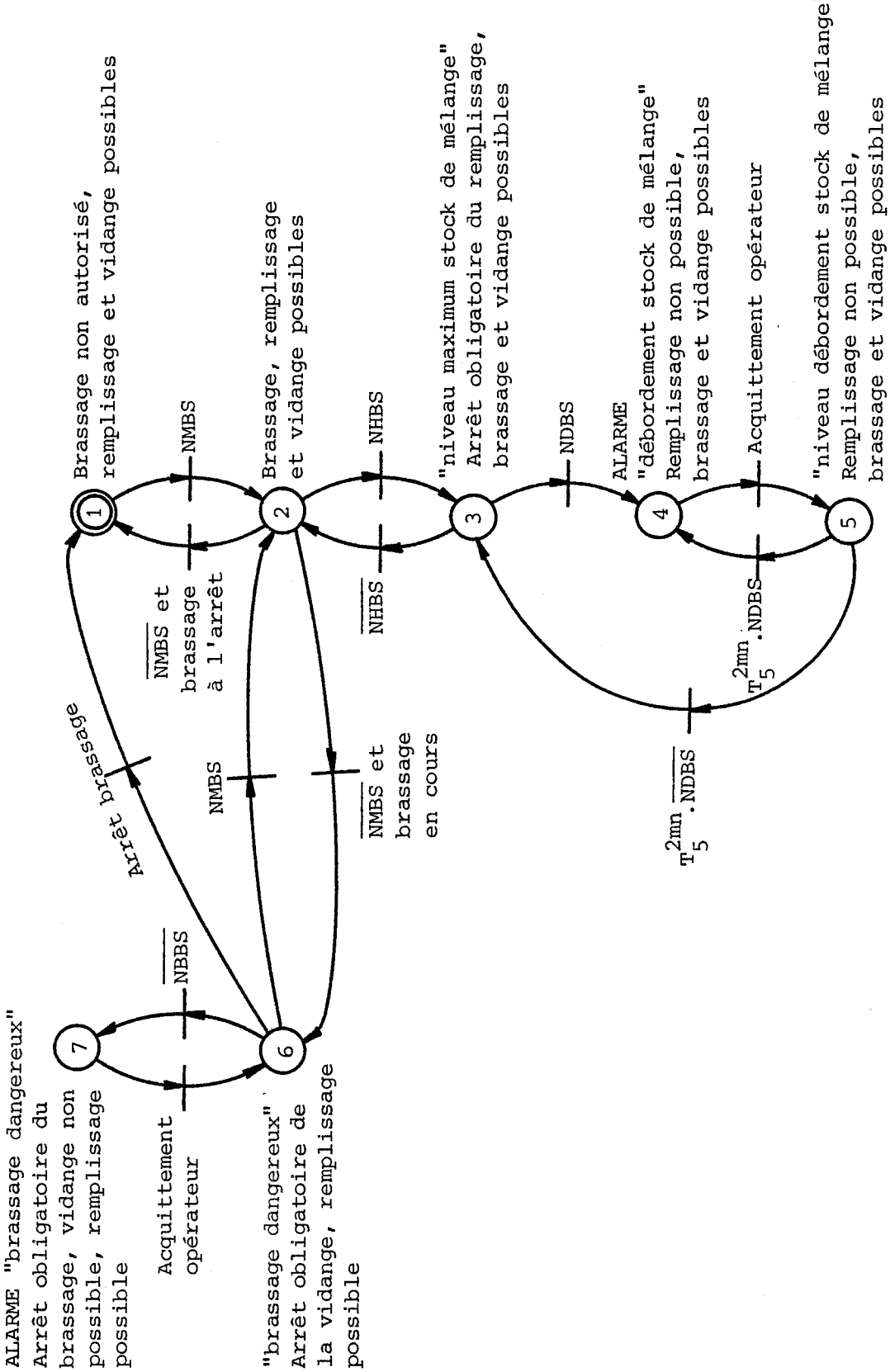


Figure 25. Automate de surveillance du bloc de stock de mélange

un ballon de stock. Supposons que l'on dispose de quatre détecteurs de niveau (NDBS, NHBS, NMBS et NBBS, de haut en bas) qui permettent de situer la quantité du mélange dans le ballon. Supposons aussi que l'on doive assurer, à côté de la fonction de brassage, les fonctions de remplissage et de vidange en observant les sécurités suivantes :

- la fonction de brassage ne peut être initialisée que si le niveau minimum NMBS est présent. Pendant que le brassage est en cours, on tolère que le niveau du mélange descende en dessous du niveau NMBS jusqu'au niveau bas NBBS. Si le niveau NBBS vient à son tour à disparaître, le brassage doit être arrêté et un signal d'alarme doit être déclenché pour avertir l'opérateur ;
- la fonction de remplissage peut être initialisée tant que le niveau du mélange est en dessous du niveau haut NHBS. Dès la détection de ce niveau, le remplissage doit être arrêté. La détection du niveau de débordement NDBS doit déclencher une alarme jusqu'à acquittement par l'opérateur ;
- enfin, la fonction de vidange peut toujours avoir lieu sauf si le niveau du mélange est en dessous du niveau NMBS pendant que le brassage est en cours.

On peut répartir ces sécurités dans les représentations associées à chaque fonction. On peut aussi résumer ces sécurités sur un seul grafcet comme le montre la figure 25.

Toutes ces remarques sont illustrées dans l'exemple du cahier des charges présenté en annexe.

BIBLIOGRAPHIE (*Partie I*)

[ABR.78] ABRIAL J.R. : Z : A specification language, IFIP Congress, TOKYO, 1978.

[BAR.75] BARBACCI M.R. : A comparison of register transfer languages for describing computers and digital systems, IEEE Trans. on Computers, Vol. C-24, n° 2, Février 1975.

[BLA.76] BLANCHARD M., CAVARROC J.C., GILLON J., THUILLIER G. : Conception modulaire d'automatismes séquentiels asynchrones, DERA - Télémécanique Electrique, Rapport DGRST n° 71-7-2912-01, Janvier 1976.

[BLA.2.79] BLANCHARD M. : Comprendre, maîtriser et appliquer le GRAFCET, Editions CEPADUES, 1979.

[BUG.81] BUGAUD J.P., DOLLE F., GUERIN P., GOUTIERRE F., MOALLA M. : Expérimentation du grafcet à la spécification de l'automatisme de commande d'une machine d'assemblage, Rapport interne DAST-RENAULT, Avril 1981.

[CAS.80] CASPI P., HALBWACHS N., MOALLA M. : Approche comportementale pour la spécification des systèmes temps réel, Journées d'étude AFCET : la spécification des problèmes, des programmes et des systèmes, TOULOUSE, Octobre 1980. (texte paru dans le bulletin GLOBULE de l'AFCET, n° 2, 1981).

[CAP.78] CAPLAIN M. : Langage de spécification, Thèse ès-Sciences, USMG-INPG, GRENOBLE, Mai 1978.

[CAV.81] CAVANNA B., DOLLE F., MOALLA M. : Outils C.A.O pour l'analyse, la spécification et la réalisation de l'automatisation d'un équipement de production mécanique, 3èmes Journées Scientifiques et Techniques de la Production Automatisée (ADEPA), TOULOUSE, Juin 1981 (et note interne DAST-RENAULT, Août 1980).

[DAV.79] DAVID R., MOALLA M., SAUCIER G., SILVA M. : Conception d'automatismes logiques répartis - Outils et mise en oeuvre, Rapport de contrat DGRST, n° 77-7-0646, Octobre 1979.

[GRA.77] Rapport final de la commission AFCET "Normalisation de la représentation du cahier des charges d'un automatisme logique", publié intégralement dans la revue Automatique et Informatique Industrielles, n° 61-62, Novembre-Décembre 1977.

[GRA.79] Document ADEPA : Le GRAFCET, diagramme fonctionnel des automatismes séquentiels, Avril 1979.

[HEN.79] HENINGER K.L. : Specifing software requirements for complex systems : New techniques and their application, Conf. on Specifications of Reliable Software, IEEE n° 79 CH 1401-9C.

[MOA.3.76] MOALLA M. : L'approche fonctionnelle dans la vérification des systèmes informatiques - Proposition d'un ensemble de méthodologies, Thèse Doc. Ingénieur, INPG-ENSIMAG, GRENOBLE, Décembre 1976.

[MOA.79] MOALLA M. : Modèle de présentation du cahier des charges d'un automatisme complexe, Rapport de recherche IMAG, n° 168, GRENOBLE, Septembre 1979.

[MOA.81] MOALLA M., DAVID R. : Extension du grafcet pour la représentation de systèmes temps réel complexes, Revue RAIRO-Automatique, Vol. 15, n° 2, 1981.

[PRU.74] PRUNET F., DUMAS J.M. : Introduction à la modélisation naturelle des structures de commande : l'organiphase, Revue RAIRO-AUTOMATIQUE, Juillet 1974, pp 45-75.

[RAM.75] RAMOS NIEMBRO G. : Contribution à l'étude de la commande séquentielle des procédés complexes, Thèse 3ième cycle, INPG-LAG, GRENOBLE, Septembre 1975.

[SIF.74] SIFAKIS J. : Modèles temporels des systèmes logiques, Thèse Doc. Ingénieur, USMG-INPG, GRENOBLE, Mars 1974.

[SUR.76] Mémoire de définition du projet pilote "Sûreté de fonctionnement des systèmes informatiques (SURF)", Janvier 1976.

[VAL.76] VALETTE R. : Sur la description, l'analyse et la validation des systèmes de commande parallèle, Thèse ès-Sciences, Université Paul Sabatier, TOULOUSE, Novembre 1976.

- ANNEXE -

MODÈLE DE PRÉSENTATION DU CAHIER DES CHARGES D' UN AUTOMATISME COMPLEXE.

UN EXEMPLE D'APPLICATION A LA SPÉCIFICATION DU CAHIER DES CHARGES DU SYSTÈME DE COMMANDE D ' UN PROCÉDÉ DE DIDTILLATION.

S O M M A I R E

I - PRESENTATION GENERALE DU PROCEDE

I-1. Principe de la distillation 1

I-2. Constitution du procédé 2

II - DESCRIPTION ET FONCTIONNEMENT DES UNITES

II-1. Conventions 3

II-2. Description de l'unité de stockage 5

II-3. Description de l'unité de mélange en ligne 7

II-4. Description des unités colonnes de distillation 9

II-5. Description de l'unité de remélange 15

III - STRUCTURE ET MOYENS DE COMMANDE DES DIFFERENTES UNITES

III-1. Inventaire global des moyens de commande et remarques préliminaires 19

III-2. Structure et moyens de commande de l'unité de mélange en ligne 27

III-3. Structure et moyens de commande d'une unité colonne de distillation 44

III-4. Structure et moyens de commande de l'unité de remélange 76

IV - SPECIFICATIONS DE L'AUTOMATISME

IV-1. Prélude 90

IV-2. Spécifications de la commande séquentielle 95

IV-2.1. Spécifications fonctionnelles 95

IV-2.2. Spécifications opérationnelles 126

IV-2.3. Spécifications technologiques 129

Avertissement :

Le cahier des charges complet comporte 130 pages et peut être trouvé aux références [MOA.79][DAV.79]. Nous n'en donnons ici que des extraits que nous pensons suffisants pour illustrer clairement la méthodologie de spécification présentée dans les trois premiers chapitres de ce mémoire.

Le premier extrait reprend tout le début du document complet jusqu'à la page 9, à savoir : (voir sommaire ci-contre)

I - PRESENTATION GENERALE DU PROCEDE

I-1. Principe de la distillation

I-2. Constitution du procédé

II - DESCRIPTION ET FONCTIONNEMENT DES UNITES

II-1. Conventions

II-2. Description de l'unité de stockage

II-3. Description de l'unité de mélange en ligne

Les paragraphes II-4 et II-5 utilisent la même approche que dans II-2 et II-3 pour donner respectivement la description des unités colonnes de distillation et celle de l'unité de remélange. Ils ne sont pas repris ici.

Nous avons choisi d'axer le reste de l'annexe autour de l'unité de mélange en ligne décrite dans le paragraphe II-3.

L'extrait suivant reprend les pages 19 à 44 à savoir :

III - STRUCTURATION ET MOYENS DE COMMANDE DES DIFFERENTES UNITES

III-1. Inventaire global des moyens de commande et remarques préliminaires

III-2. Structure et moyens de commande de l'unité de mélange en ligne

Les paragraphes III-3 et III-4 utilisent la même démarche que dans III-2 pour décrire les structures et moyens de commande respectivement d'une unité colonne de distillation et de l'unité de remélange. Ils ne sont pas repris ici.

Enfin, le troisième extrait reprend tout le début du volet réservé aux spécifications de l'automatisme et ce jusqu'à la page 100, à savoir :

IV - SPECIFICATIONS DE L'AUTOMATISME

IV-1. Prélude

IV-2. Spécifications de la commande séquentielle

IV-2.1. Spécifications fonctionnelles

Nous reprenons complètement les grafcets suivants :

G1 : Coordination de la commande des différentes unités lors de la phase de démarrage du procédé

G3 : Grafcet de commande de l'unité de mélange en ligne en phase de démarrage

G7 : Grafcet de la macroétape d'arrêt normal dans le grafcet de commande de l'unité de mélange en ligne (G3)

G10 : Automate de surveillance d'un bloc du module de stock primaire de l'unité de mélange en ligne

G13 : Grafcets des contraintes de sécurité fonctionnelle à respecter absolument.

I - PRESENTATION GENERALE DU PROCEDE

I - 1. PRINCIPE DE LA DISTILLATION

Le procédé en place est une chaîne de distillation. Les produits traités sont essentiellement des mélanges de méthanol, d'éthanol et d'eau distillée. La distillation utilise alors la propriété que ces différents constituants ont des températures d'ébullition différentes : si l'on porte à une certaine température un mélange de deux composants placé dans un bloc fermé et à pression constante, la vapeur que l'on obtient a une forte concentration du composant ayant le point d'ébullition bas, alors que dans le liquide on trouve plus du composant au point d'ébullition élevé.

Toutefois, la séparation que l'on obtient ainsi n'est, en général, pas suffisante. Pour cette raison, on est amené à construire des cascades de bacs de distillation formant ce que l'on appelle une *colonne à distiller* (Figure 1). Le produit traité, qui peut se présenter sous la forme d'un liquide ou d'un mélange gaz-liquide est introduit par la section d'alimentation située à une hauteur intermédiaire de la colonne. Le long de la colonne, il se produit un échange de matière entre une phase liquide et une phase vapeur circulant à contre-sens. Le liquide qui descend à travers la cascade de bacs entre en contact intime avec la vapeur montante et participe à des échanges massiques et thermiques; il perd ainsi graduellement sa composante la plus légère tandis que la vapeur, en montant, en gagne progressivement.

Au pied de la colonne, une partie du liquide qui arrive est vaporisée par un apport d'énergie calorifique dans le bouilleur. L'autre partie qui est riche en la composante la plus lourde est retirée de la colonne, constituant ce qu'on appelle le "produit de pied".

La vapeur montante est condensée en tête de la colonne. Une partie du liquide obtenu, qui est riche en la composante la plus légère, est recueillie, constituant ce qu'on appelle le "produit de tête". L'autre partie redescend à travers les bacs permettant d'améliorer la séparation.

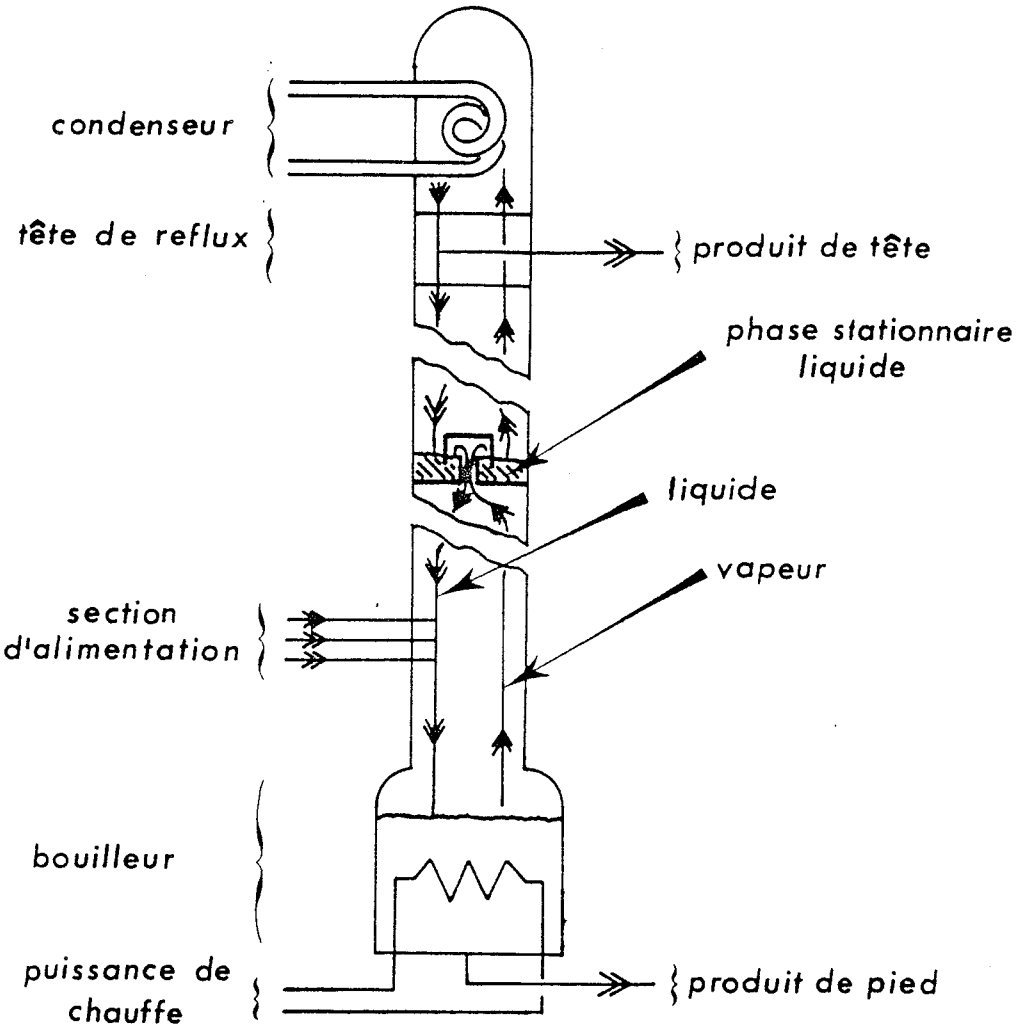


Figure 1. Schéma d'une colonne à distiller

I - 2. CONSTITUTION DU PROCEDE

L'installation complète se compose de 5 unités (Figure 2).

- Une unité de stockage (1)
- Une unité de mélange en ligne
- Deux unités colonne de distillation (colonne-1 et colonne-2)
- Une unité de remélange.

L'ensemble a été conçu de façon à ce que chaque unité puisse fonctionner indépendamment des autres. Cela donne la possibilité de changer la structure du procédé en cours de fonctionnement (colonnes de distillation en série, colonnes de distillation en parallèle, arrêt provisoire d'une unité pour intervention...).

---

(1) Cette unité n'étant pas encore construite, elle n'apparaît pas dans la figure 2. Sa description sera donnée dans § II-2.

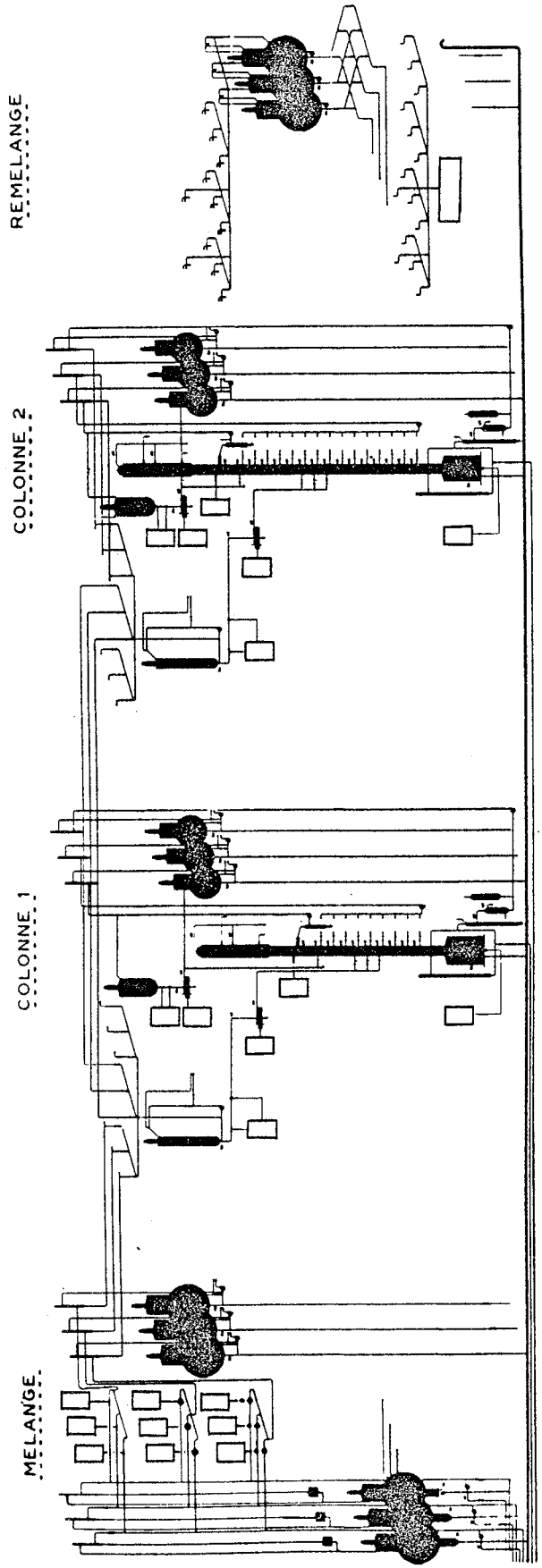


Figure 2. Panneau synoptique du procédé pilote de distillation

II - DESCRIPTION ET FONCTIONNEMENT DES UNITES

II - 1. CONVENTIONS

Afin de faciliter la description de la constitution physique du procédé et de son fonctionnement, les différentes unités sont présentées séparément. On montrera, pour chaque unité, quelles sont ses interconnexions avec les autres unités.

Chaque unité est composée d'un certain nombre de modules eux-mêmes composés de blocs. Chaque bloc est constitué d'un certain nombre d'éléments (Figure 3).

On adoptera dans la schématisation du procédé physique le symbolisme suivant :

Vanne tout-ou-rien

Vanne analogique

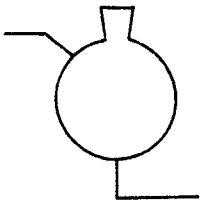
Pompe tout-ou-rien

Pompe volumétrique

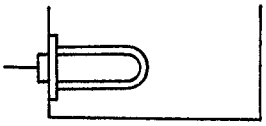
Pompe double sens

Réservoir

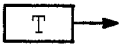
Ballon



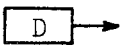
Préchauffage



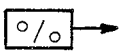
Mesure de température



Mesure de débit

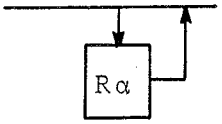


Mesure de concentration



Régulateur Rα

(α = D pour Débit, PC pour Puissance de Chauffe,
C pour Concentration, T pour Température)



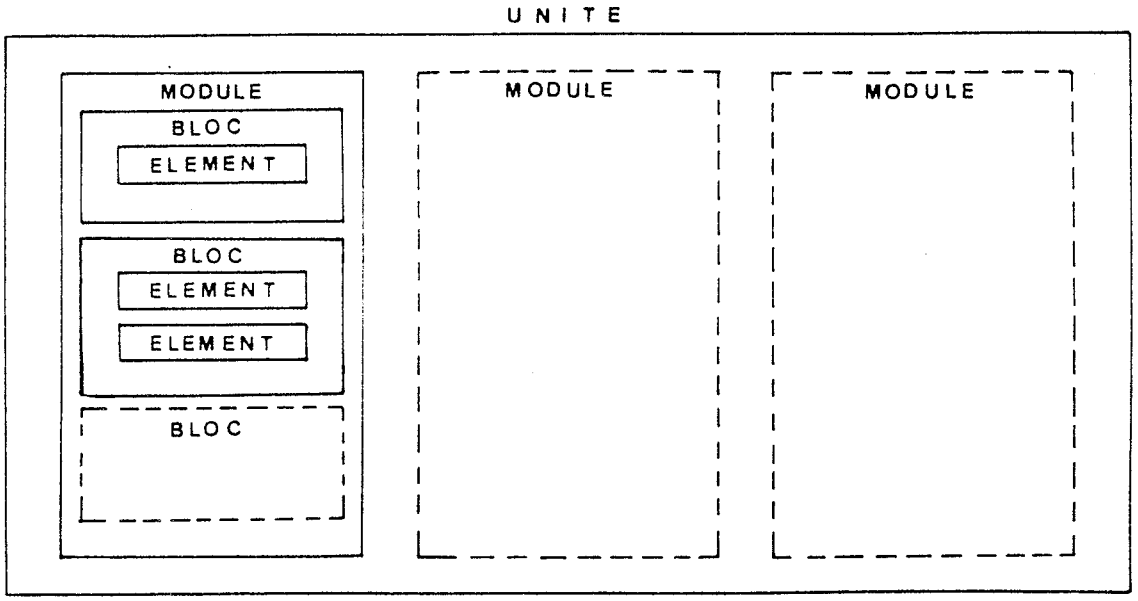


Figure 3. Nomenclature des niveaux de décomposition d'une unité

II - 2. DESCRIPTION DE L'UNITE DE STOCKAGE

Cette unité sert au stockage des produits quand le procédé est à l'arrêt, ou lorsqu'il s'agit de produits de réserve ou de produits prélevés au cours de la distillation pour des fins de test. Les produits traités sont inflammables. Par mesure de sécurité, ils sont gardés dans des réservoirs enterrés à une profondeur et à une distance du reste du procédé fixées par les normes de sécurité.

L'unité est composée d'un module de stockage et d'un module d'interconnexion (Figure 4).

1. MODULE DE STOCKAGE

Ce module est composé de 15 blocs, chaque bloc étant constitué d'un réservoir de stock. Les 15 réservoirs sont distribués de la façon suivante :

- 3 réservoirs numérotés de 1 à 3 sont affectés à l'unité de mélange en ligne;
- 2 réservoirs numérotés 4 et 5 sont affectés aux unités colonne;
- 10 réservoirs numérotés de 6 à 15, non affectés de façon permanente, sont utilisés pour stock secondaire.

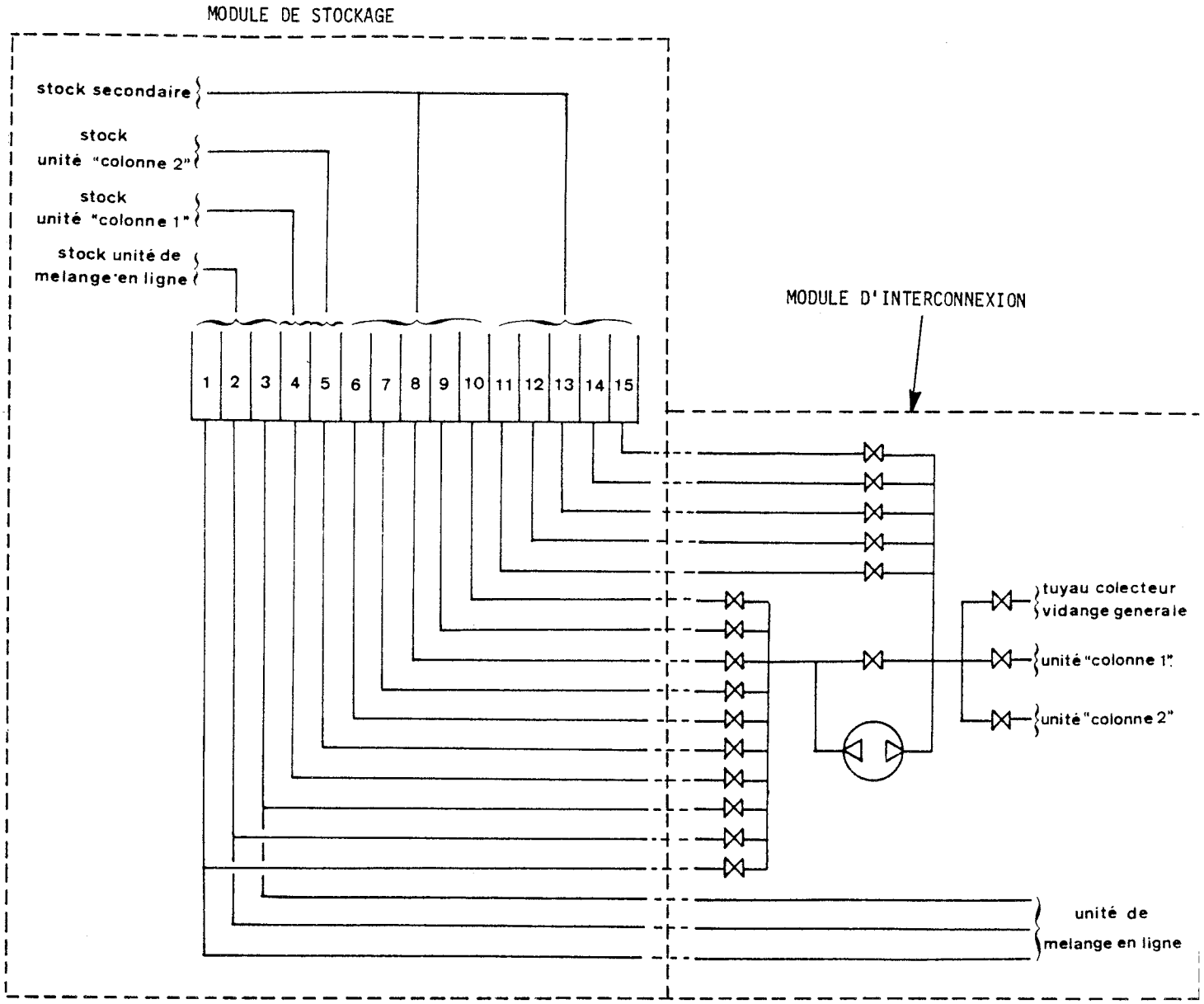


Figure 4. Description de l'unité de stockage

2. MODULE D'INTERCONNEXION

Chacun des réservoirs numérotés 1 à 3 est connecté à l'unité de mélange en ligne par l'intermédiaire d'une canalisation et d'une pompe indépendantes (les 3 pompes se trouvent dans l'unité de mélange en ligne). Il est ainsi possible de les remplir ou de les vidanger sans avoir à tenir compte du fonctionnement du reste de l'unité.

Les 12 autres réservoirs sont connectés au procédé par l'intermédiaire d'un réseau de canalisations et de vannes tout-ou-rien constituant le module d'interconnexion. Le remplissage et la vidange de ces réservoirs sont assurés par une même pompe à deux sens donnant les possibilités suivantes :

- transfert du produit de l'un quelconque des réservoirs de stock secondaire numérotés de 11 à 15 à l'un des réservoirs 1 à 10 et inversement;
- remplissage de l'unité colonne 1 ou colonne 2 à partir des réservoirs affectés 4 et 5 ou à partir des réservoirs de stock secondaire numérotés 6 à 10.

Le niveau de l'unité de stockage étant inférieur au reste du procédé, le remplissage des différents réservoirs peut se faire par gravité. D'une façon générale, la pompe est utilisée seulement pour le remplissage ou la vidange des unités colonne-1 et colonne-2; l'utilisation des réservoirs de stock secondaire est plutôt circonstancielle :

- stockage des produits à des quantités supérieures à celles prévues dans les réservoirs normaux;
- prélèvement de produits intermédiaires pendant le processus de distillation et stockage pour des fins de test;
- stockage pour essai de produits différents de ceux utilisés normalement;
- vidange générale rapide en cas d'alarme par l'intermédiaire d'un tuyau collecteur qui traverse toute l'installation.

II - 3. DESCRIPTION DE L'UNITE DE MELANGE EN LIGNE

Le rôle de cette unité est de préparer les mélanges qui vont alimenter les deux colonnes de distillation. Comme on l'a déjà mentionné, le procédé étudié, bien qu'il soit d'échelle industrielle, est utilisé uniquement pour des fins expérimentales de recherche. Ce sont, par conséquent, les mêmes produits qui sont utilisés dans un cycle fermé de mélange puis séparation. L'unité de mélange en ligne permet d'obtenir des mélanges à différents degrés de concentration des trois produits de base. Le rôle de l'unité de remélange, que l'on décrira plus tard, est de récupérer les produits séparés aux sorties des colonnes et de les renvoyer à l'entrée de l'unité de mélange en ligne.

L'unité de mélange en ligne est composée de 3 modules (Figure 5).

- un module de stock primaire,
- un module de mixage,
- un module de stock secondaire.

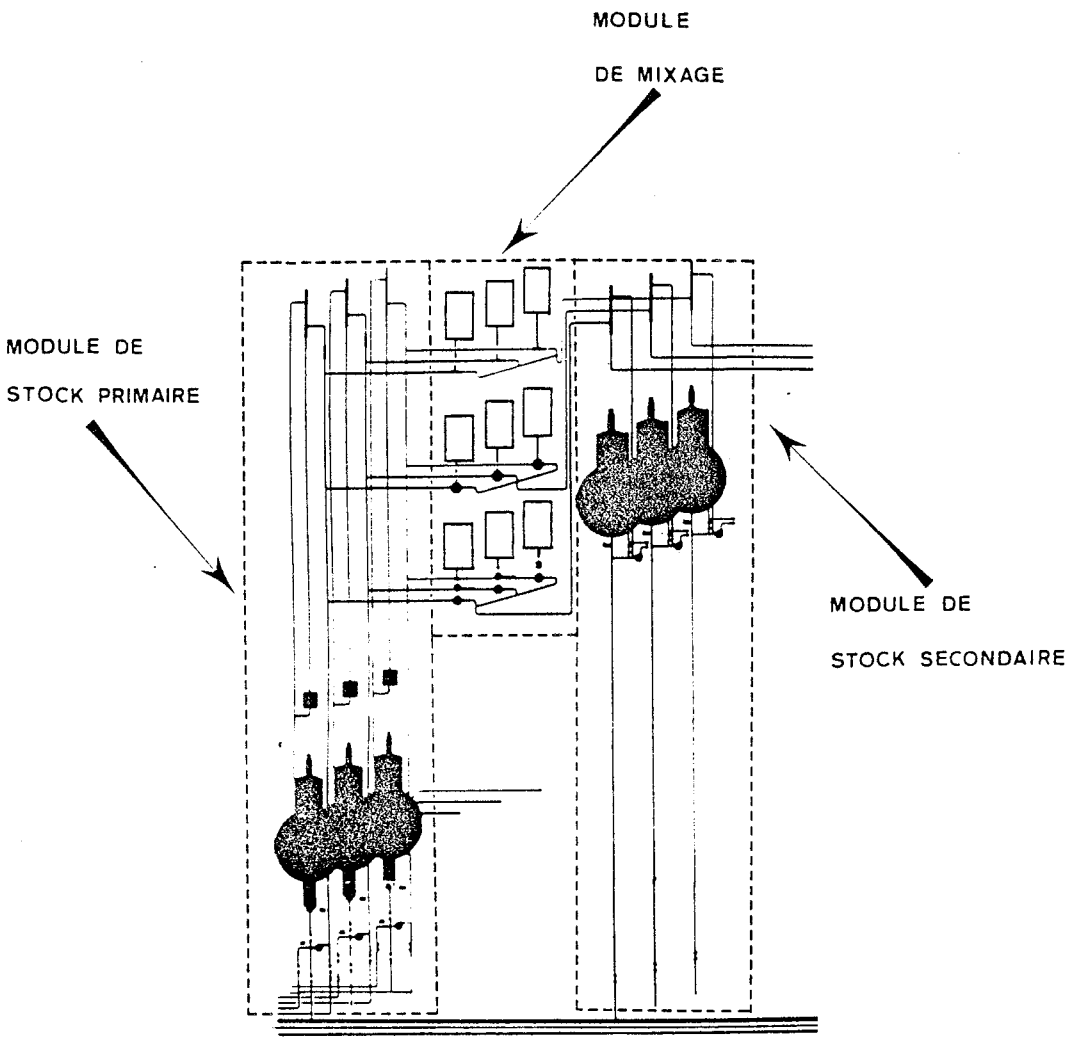


Figure 5. Unité de mélange en ligne

1. MODULE DE STOCK PRIMAIRE

Ce module est constitué de 3 blocs de stockage identiques où sont admis séparément les trois produits de base provenant, soit de l'unité de stockage, soit de l'unité de remélange. Chaque bloc de stockage a comme élément principal un ballon. Pour maintenir les produits dans des conditions de température et de concentration homogènes, chaque ballon est équipé d'un échangeur thermique et d'une pompe qui assure un brassage continu. Ce brassage permet en même temps d'envoyer le produit vers le module de mixage.

2. MODULE DE MIXAGE

Ce module est constitué de 3 blocs de mixage dont les éléments principaux diffèrent. Le premier bloc a trois vannes tout-ou-rien; le deuxième bloc a trois vannes analogiques et le troisième trois pompes volumétriques. Il y a, ainsi, trois façons différentes d'obtenir des mélanges. Les trois blocs sont alimentés à partir des ballons du module de stock primaire, et leurs sorties sont connectées avec le module de stock secondaire.

3. MODULE DE STOCK SECONDAIRE

Ce module est constitué de 3 blocs de stockage identiques ayant chacun, comme éléments principaux, un trop-plein et un ballon de stock. Les trop-pleins recueillent les mélanges obtenus en sortie des blocs de mixage, soit pour les acheminer directement vers les unités colonne, soit pour les envoyer dans les ballons de stock. Chaque ballon de stock est équipé d'une pompe qui permet d'effectuer un brassage du mélange stocké en le renvoyant dans le trop-plein d'où il peut être acheminé vers les unités colonne. Ce brassage peut être court ou long. Le circuit utilisé pour le brassage long va jusqu'à l'unité de remélange d'où l'on peut effectuer des prélèvements du mélange pour mesurer sa concentration. Ce circuit sert aussi pour vidanger le bloc de stock dans l'unité de remélange (voir plus loin dans § II-5 : Description de l'unité de remélange).

Grâce à ces blocs de stockage, l'unité de mélange en ligne peut avoir un fonctionnement indépendant des autres unités.

II - 4. DESCRIPTION DES UNITES COLONNES DE DISTILLATION

La constitution et le fonctionnement d'une colonne de distillation ont été déjà introduits dans § I-1. Les deux unités colonne-1 et colonne-2 sont de ces points de vue identiques à deux différences près : (Figures 6 et 7).

- Le nombre de plateaux (étages de bacs) est de 9 dans l'unité colonne-1 alors qu'il est de 15 dans l'unité colonne-2;

- La puissance de chauffe maximum dans le bouilleur de la colonne-1 est de 72 KW alors qu'elle n'est que de 56 KW dans la colonne-2.

Les deux unités peuvent être interconnectées avec les autres unités du procédé pour travailler en parallèle ou en série (Figure 8). Dans le premier cas, les deux unités sont alimentées à partir de l'unité mélange en ligne

Nous sautons ici les paragraphes :

II - 4. DESCRIPTION DES UNITES COLONNES DE DISTILLATION

II - 5. DESCRIPTION DE L'UNITE DE REMELANGE

Rappelons que ces paragraphes utilisent la même approche que dans II-2 et II-3 pour donner, respectivement, la description des unités colonnes de distillation et celle de l'unité de remélange.

III - STRUCTURE ET MOYENS DE COMMANDE DES DIFFERENTES UNITES

On a donné dans les paragraphes I et II une description globale du procédé physique, de la constitution de ses unités et de leur fonctionnement. Dans ce paragraphe, on va présenter de façon plus détaillée la structure de chaque unité, les fonctions réalisées par l'unité et les moyens de commande et de contrôle disponibles.(ou envisageables). On indiquera quels sont, parmi ces moyens de commande, ceux qui sont destinés à être pilotés par l'automatisme et on donnera alors les spécifications technologiques de l'interface qu'ils présentent. Toute contrainte sur les moyens de commande impliquée par des considérations fonctionnelles, technologiques ou de sécurité sera soulignée.

III - 1. INVENTAIRE GLOBAL DES MOYENS DE COMMANDE ET REMARQUES PRELIMINAIRES

1. L'unité de stockage (§II.2) n'est pas encore construite et son architecture n'est pas arrêtée de façon défintive, il n'est par conséquent pas possible de faire un inventaire précis de son interface de commande et des fonctions qu'elle réalise. Toutefois, cette lacune ne gênera nullement la description détaillée des autres unités puisqu'il est prévu que la commande de cette unité restera strictement manuelle. Dans la description de l'interface de commande du procédé, on fera référence à deux commandes globales sur cette unité :

- "Sélection de réservoir" : chaque fois qu'une opération de remplissage ou de vidange est effectuée en fonctionnement normal, l'opérateur doit initialiser les commandes nécessaires (ouverture/fermeture de vannes, marche/arrêt pompe double sens) afin d'assurer la bonne exécution de cette opération.

- "Remise à zéro sélection de réservoir" : l'état au repos de l'unité de stockage doit être tel que toutes les canalisations de remplissage des réservoirs de stock secondaire soient ouvertes et la pompe double sens arrêtée. Cela permet de maintenir l'unité de stockage prête pour la vidange d'urgence en cas d'alarme générale. Toute opération de remplissage ou de vidange en fonctionnement normal doit, par conséquent, commencer par une commande de sélection de réservoir et se terminer par une remise à zéro de cette sélection.

2. Les commandes suivantes sont prévues au niveau du procédé global :

. Marche/Arrêt pression air
. Marche/Arrêt pression azote
. Marche/Arrêt secteur (alimentation courant)
. Test présence secteur.

Toutefois, ces commandes ne sont effectives que si elles sont confirmées par des commandes locales au niveau de chaque unité (sauf la commande M/A secteur) :

. Marche/Arrêt local pression air
. Marche/Arrêt local pression eau
. Marche/Arrêt local pression azote
. Test présence pression air
. Test présence pression eau
. Test présence pression azote

Ces commandes ne figureront pas dans l'inventaire des moyens de commande que l'on va établir pour chaque unité.

3. Un certain nombre de régulateurs sont utilisés :

. Régulateurs débit-concentration dans l'unité de mélange en ligne,
. Régulateurs débit et régulateurs de puissance de chauffe dans les unités colonne de distillation.

L'utilisateur dispose, sur chaque régulateur, de deux disjoncteurs de commande :

- un disjoncteur pour la commande Marche/Arrêt du régulateur (commande locale de mise sous tension du régulateur),
- un disjoncteur pour la sélection du mode de fonctionnement Manuel (MN)/ Automatique (AU).

Le circuit de commande du régulateur est conçu de telle façon que :

a) Tant que l'alimentation générale est coupée, les deux disjoncteurs sont ouverts, ce qui met le régulateur à l'état "arrêt, mode de fonctionnement manuel" La commande de mise en marche du régulateur doit donc être réeffectuée après chaque coupure (Arrêt-Marche) de l'alimentation.

b) La commutation du mode de fonctionnement manuel au mode de fonctionnement automatique ne peut être effectuée que si le régulateur est à l'état "marche" (sous alimentation). L'opérateur commande cette commutation en utilisant le disjoncteur de sélection. L'automatisme vient alors lire l'affichage manuel en cours pour ajuster en conséquence l'affichage automatique avant de commander un relais rendant cette commutation effective.

c) La commutation du mode de fonctionnement automatique au mode de fonctionnement manuel est immédiate dès action sur le disjoncteur de sélection.

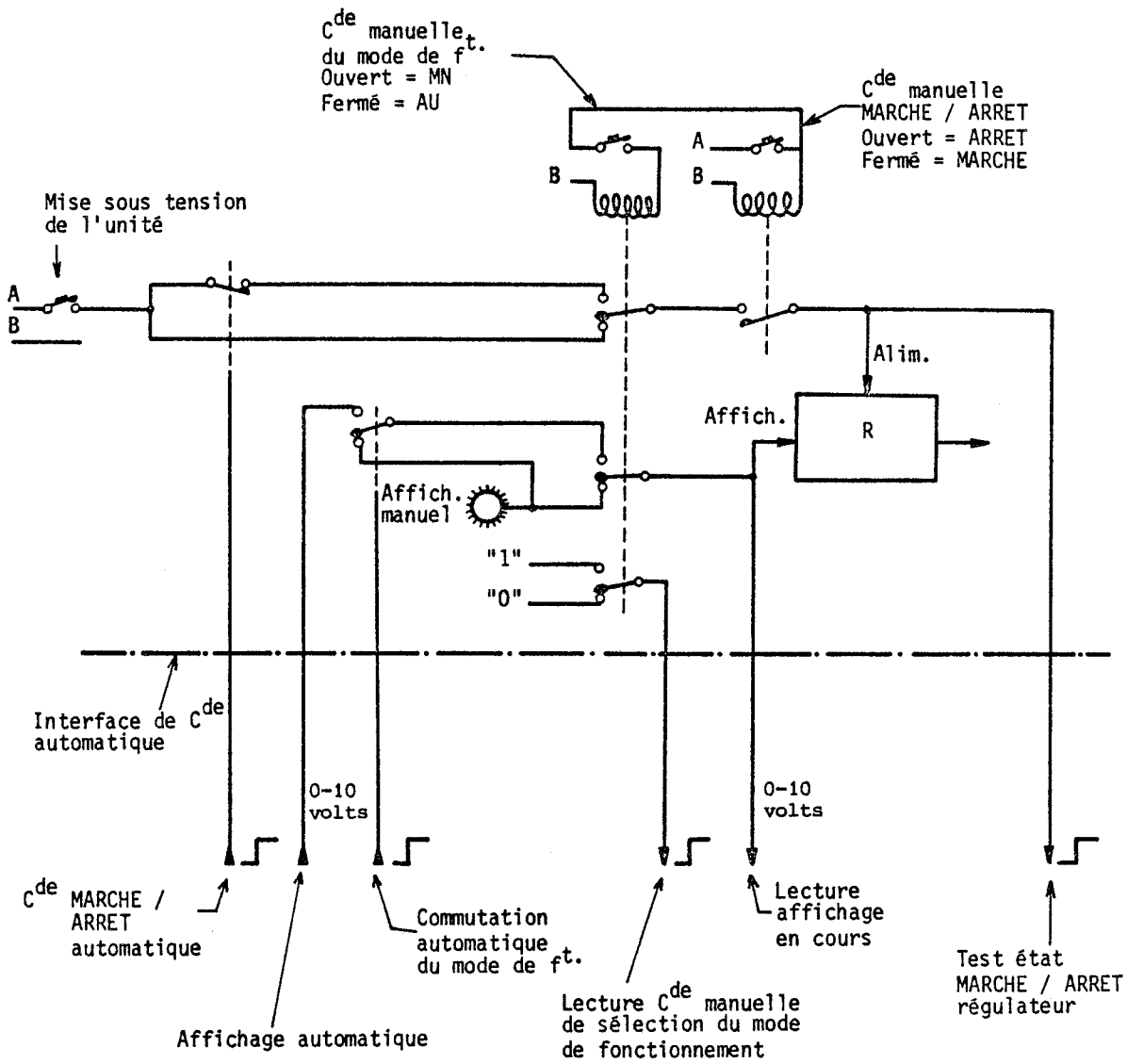


Figure 10. Schéma fonctionnel du circuit de commande d'un régulateur

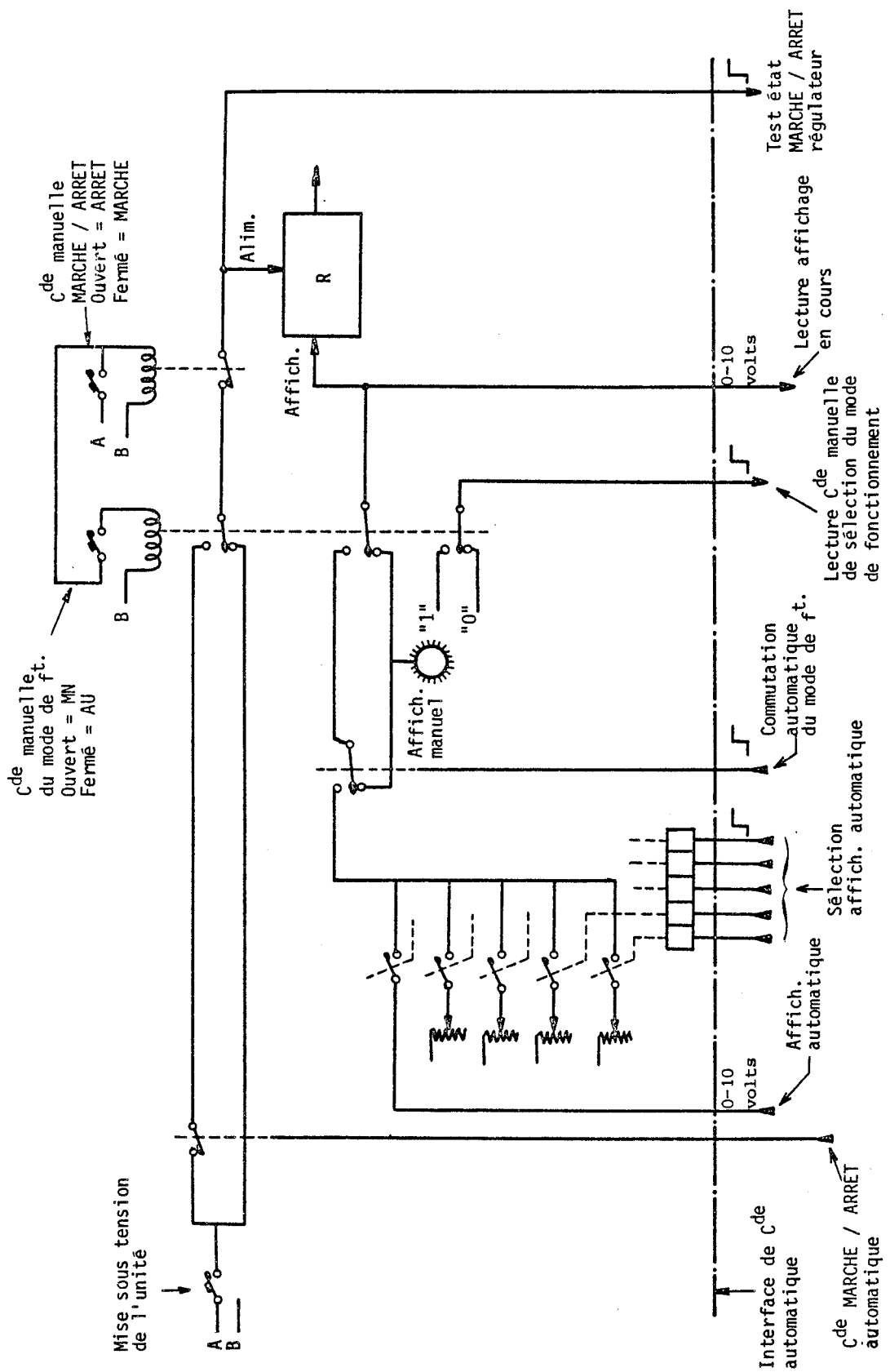


Figure 11. Schéma fonctionnel du circuit de commande du régulateur de puissance de chauffe.

d) L'automatisme dispose d'une commande Marche/Arrêt du régulateur, mais cette commande est non effective tant que le disjoncteur de commande manuelle est en position "Arrêt".

Ceci est illustré sur le schéma fonctionnel donné en figure 10. Ce schéma est commun pour tous les circuits de commande des régulateurs.

4. Pour commander l'affichage automatique des régulateurs de puissance de chauffe, l'automatisme peut soit échantillonner un signal analogique entre 0 et 10 volts, soit commander la commutation sur un affichage préréglé. Il est possible de prérégler (manuellement) quatre affichages différents. L'affichage en mode de fonctionnement manuel est fait à l'aide d'un potentiomètre indépendant (voir schéma fonctionnel donné en figure 11).

5. La mise en marche des appareils de mesure de température, de concentration et de débit est enclenchée en même temps que l'alimentation générale. Les mesures sont disponibles en permanence sous forme de signaux analogiques 0 - 10 volts (pour la mesure de débit : sortie analogique 4 - 20 mA).

6. En cas d'alarme générale, la coupure de secteur est systématique entraînant la mise de toutes les électrovannes à l'état "fermé" et de tous les régulateurs, pompes et appareils de mesure à l'état "arrêt". Seule la commande des électrovannes de vidange générale reste possible, grâce à un circuit d'alimentation indépendant du circuit d'alimentation principal.

7. Le tableau ci-dessous donne un inventaire global des types des moyens de commande à l'interface du procédé.

Légende : C = Commande
M = Mesure
T = Test niveau logique
MN = MaNuelle
AU = AUtomatique
⎍⎍ = Signal à niveau

INVENTAIRE DES TYPES DE MOYENS DE COMMANDE A L'INTERFACE DU PROCEDE

Accès	Type	Fonction réalisée	Mnémonique commande	Interface de commande manuelle	Interface de commande automatique				
					ref. signal entrée	ref. signal sortie	log. analog.	Valeur	Temps de réponse
MN/AU	C	Ouverture/Fermeture d'électrovanne	OUV(Vxy) FERM(Vxy)	Bouton poussoir (Λ) (à 1 position)	Vxy		⎍⎍, TTL	1=ouverture 0=fermeture	100 ms
MN/AU	T	Test état Ouvert/Fermé d'électrovanne	TVxy ?	Voyant lumineux		TVxy	⎍⎍, TTL	1=ouvert 0=fermé	
MN/AU	C	Marche/Arrêt pompe tout ou rien	MARCHE(Pxy) ARRET(Pxy)	Bouton à 1 position (Λ)	Pxy		⎍⎍, TTL	1=marche 0=arrêt	0,5 à 1 s
MN/AU	T	Test état Marche/Arrêt pompe tout ou rien	TPxy ?	Voyant lumineux		TPxy	⎍⎍, TTL	1=en marche 0=à l'arrêt	
MN/AU	C	Marche/Arrêt régulateur débit-concentration	MARCHE(RDCxy) ARRET(RDCxy)	Bouton à 1 position (Λ)	RDCxy		⎍⎍, TTL	1=marche 0=arrêt	
MN/–	C	Sélection du mode de fonctionnement Manuel/Auto.		Bouton à 2 positions (⎍⎍)					
MN/AU	C	Affichage consigne débit	AFFD(RDCxy,d)	Potentiomètre gradué	ADRDCxy		analogique	0 - 10 volts	
MN/AU	C	Affichage consigne concentration	AFFC(RDCxy,c)	Potentiomètre gradué	ACRDCxy		analogique	0 - 10 volts	
MN/AU	T	Test état Marche/Arrêt du régulateur	TRDCxy ?	Témoin vert		TRDCxy	⎍⎍, TTL	1=en marche 0=à l'arrêt	
MN/AU	T	Test fonctionnement normal du régulateur	TFNRDCxy ?	Témoin rouge en cas de fonctionnement anormal		TFNRDCxy	⎍⎍, TTL	1=ft.anormal 0=ft.normal	
MN/AU	M	Lecture affichage débit	DAFF(RDCxy)	Lecture visuelle sur voltmètre		LDRDCxy	analogique	0 - 10 volts	
MN/AU	M	Lecture affichage concentration	CAFF(RDCxy)			LCRDCxy	analogique	0 - 10 volts	
MN/AU	T	Lecture sélection du mode de fonctionnement (Manuel/Auto.)	MN?/AU?	Position du bouton de commande		MNAUxy	⎍⎍, TTL	1=mode AU 0=mode MN	
MN/AU	C	Marche/Arrêt régulateur puissance de chauffe	MARCHE(RPCxy) ARRET(RPCxy)	Bouton à 1 position (Λ)	RPCxy		⎍⎍, TTL	1=marche 0=arrêt	
MN/–	C	Sélection du mode de fonctionnement Manuel/Auto.		Bouton à 2 positions (⎍⎍)					
MN/AU	C	Affichage consigne puissance de chauffe	AFFP(RPCxy,p)	Potentiomètre gradué	APRPCxy		analogique	0 - 10 volts	
MN/AU	T	Test état Marche/Arrêt du régulateur	TRPCxy ?	Témoin vert		TRPCxy	⎍⎍, TTL	1=en marche 0=à l'arrêt	
MN/AU	T	Test fonctionnement normal du régulateur	TFNRPCxy	Témoin rouge en cas de ft. anormal		TFNRPCxy	⎍⎍, TTL	1=ft.anormal 0=ft.normal	
MN/AU	M	Lecture affichage puissance de chauffe	PAFF(RPCxy)	Lecture visuelle sur voltmètre		LPRPCxy	analogique	0 - 10 volts	
MN/AU	T	Lecture sélection du mode de fonctionnement	MN?/AU?	Position du bouton de commande		MNAUxy	⎍⎍, TTL	1=mode AU 0=mode MN	
MN/AU	C	Marche/Arrêt régulateur de température	MARCHE(RTxy) ARRET(RTxy)	Bouton à 1 position (Λ)	RTxy		⎍⎍, TTL	1=marche 0=arrêt	
MN/–	C	Sélection du mode de fonctionnement Manuel/Auto.		Bouton à 2 positions (⎍⎍)					
MN/AU	C	Affichage consigne t° préchauffage	AFFT(RTxy,t)	Potentiomètre gradué	ATRTxy		analogique	0 - 10 volts	
MN/AU	T	Test état Marche/Arrêt du régulateur	TRTxy ?	Témoin vert		TRTxy	⎍⎍, TTL	1=en marche 0=à l'arrêt	
MN/AU	T	Test fonctionnement normal du régulateur	TFNRTxy ?	Témoin rouge en cas de ft. anormal		TFNRTxy	⎍⎍, TTL	1=ft.anormal 0=ft.normal	

.
.
.
.
.
.
.
.
.
.

Nous sautons ici la suite de l'inventaire des types de moyens de commande à l'interface du procédé.

III - 2. STRUCTURE ET MOYENS DE COMMANDE DE L'UNITE DE MELANGE EN LIGNE

La figure 12 illustre la structure des 3 modules constituant l'unité de mélange en ligne.

1. MODULE DE STOCK PRIMAIRE (constitué de 3 blocs de stockage identiques, voir figure 13)

Commandes :

MN/AU Ouverture/Fermeture d'électrovannes

bloc 1 : V11/V12/V13/V14

bloc 2 : V41/V42/V43/V44

bloc 3 : V71/V72/V73/V74

MN/AU Marche/Arrêt pompes tout ou rien

bloc 1 : P11

bloc 2 : P41

bloc 3 : P71

MN/- Sélection mode de fonctionnement Manuel/Automatique (commande globale pour le module de stock primaire)

Mesures et tests

MN/AU Mesures de température

bloc 1 : T11/T12

bloc 2 : T41/T42

bloc 3 : T71/T72

MN/AU Mesures de concentration

bloc 1 : C13

bloc 2 : C43

bloc 3 : C73

MN/AU Test état Ouvert/Fermé d'électrovannes

bloc1 : TV11?/TV12?/TV13?

bloc 2 : TV41?/TV42?/TV43?

bloc 3 : TV71?/TV72?/TV73?

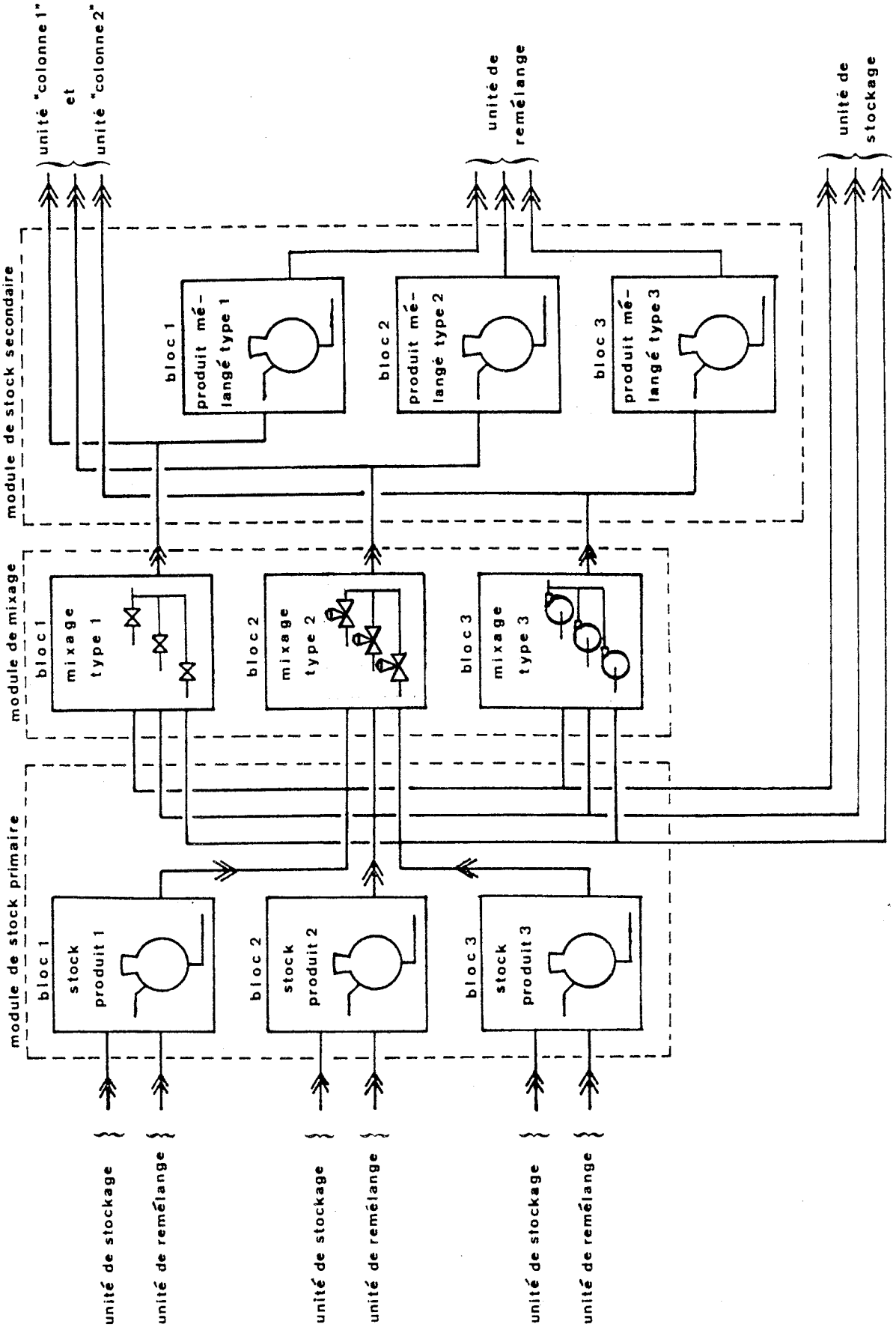


Figure 12. Structure de l'unité de mélange en ligne.

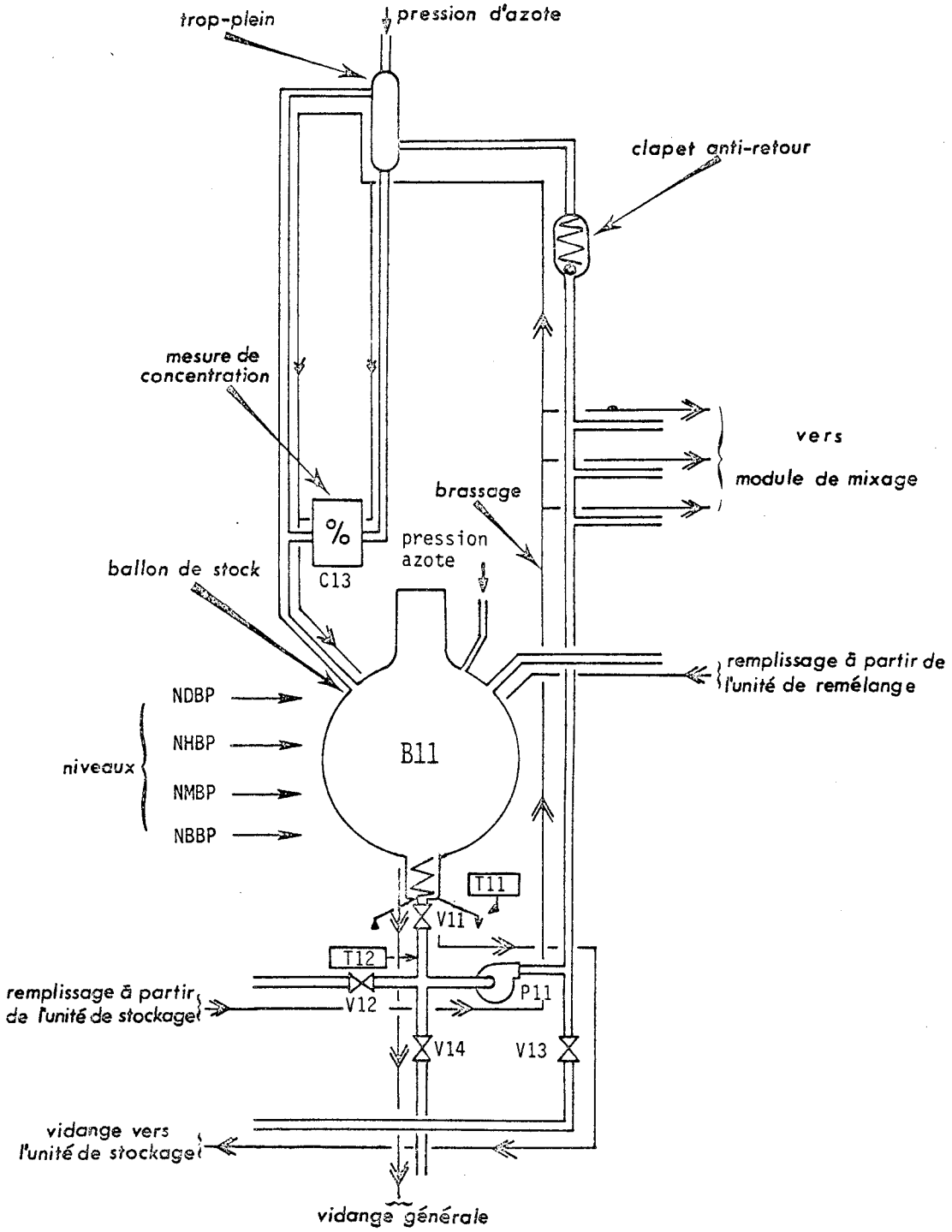


Figure 13. Unité de mélange en ligne.

Schéma détaillé du fonctionnement d'un bloc de stockage du module de stock primaire.

MN/AU Test état Marche/Arrêt pompes tout ou rien

bloc 1 : TP11?

bloc 2 : TP41?

bloc 3 : TP71?

-/AU Niveaux dans les ballons de stock primaire

bloc 1 : NBBP1?/NMBP1?/NHBP1?/NDBP1?

bloc 2 : NBBP4?/NMBP4?/NHBP4?/NDBP4?

bloc 3 : NBBP7?/NMBP7?/NHBP7?/NDBP7?

MN/AU Lecture sélection mode de fonctionnement du module (NM/AU)

Fonctions assurées

Remarque 1 : Les 3 blocs étant identiques, la description des fonctions ne sera donnée que pour le bloc 1 (sauf pour la fonction vidange générale d'urgence).

Remarque 2 : La vidange générale d'urgence est prioritaire sur toute autre fonction.

Remarque 3 : La description de chaque fonction comprendra :

* Une spécification du *mode fonctionnement* dans lequel l'unité (ou le module) doit être pour que la fonction puisse être exécutée. Certaines fonctions (comme, par exemple, le remplissage des ballons de stock à partir de l'unité de stockage) ne peuvent être exécutées qu'en mode manuel; l'automatisme n'aura pas, par conséquent, à conduire l'exécution de ces fonctions.

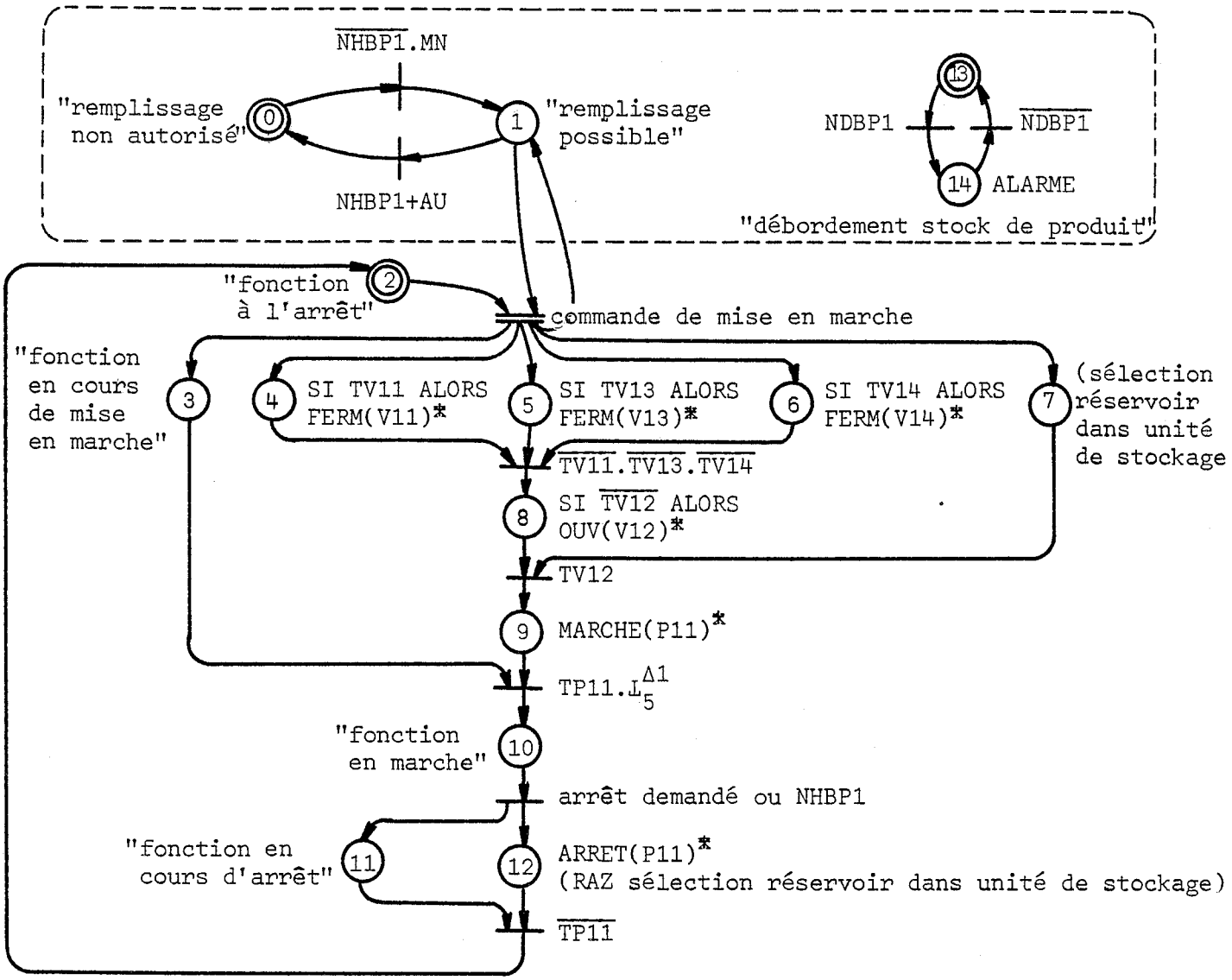
* Une spécification des *contraintes* qui peuvent empêcher le lancement (ou la poursuite) de l'exécution de la fonction.

* Le *grafcet de commande*. On encadrera par des pointillés les étapes représentant les états acceptables du procédé au moment de l'initialisation de l'exécution de la fonction.

1) Remplissage ballon de stock à partir de l'unité de stockage

* Mode de fonctionnement : manuel
* Contraintes :
  - le remplissage ne peut être effectué (/doit être arrêté) si le niveau NHBP1 est atteint
  - alarme sur détection niveau NDBP1.

* Grafcet de commande :



2) Remplissage ballon de stock à partir de l'unité de remélange

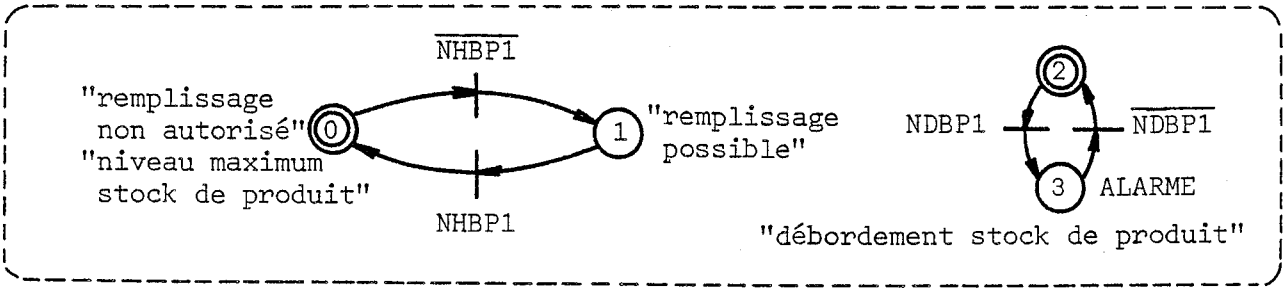
* Mode de fonctionnement : manuel/automatique

* Contraintes :

- le remplissage ne peut être effectué (/doit être arrêté) si le niveau NHBP1 est atteint ;
- alarme sur détection niveau NDBP1

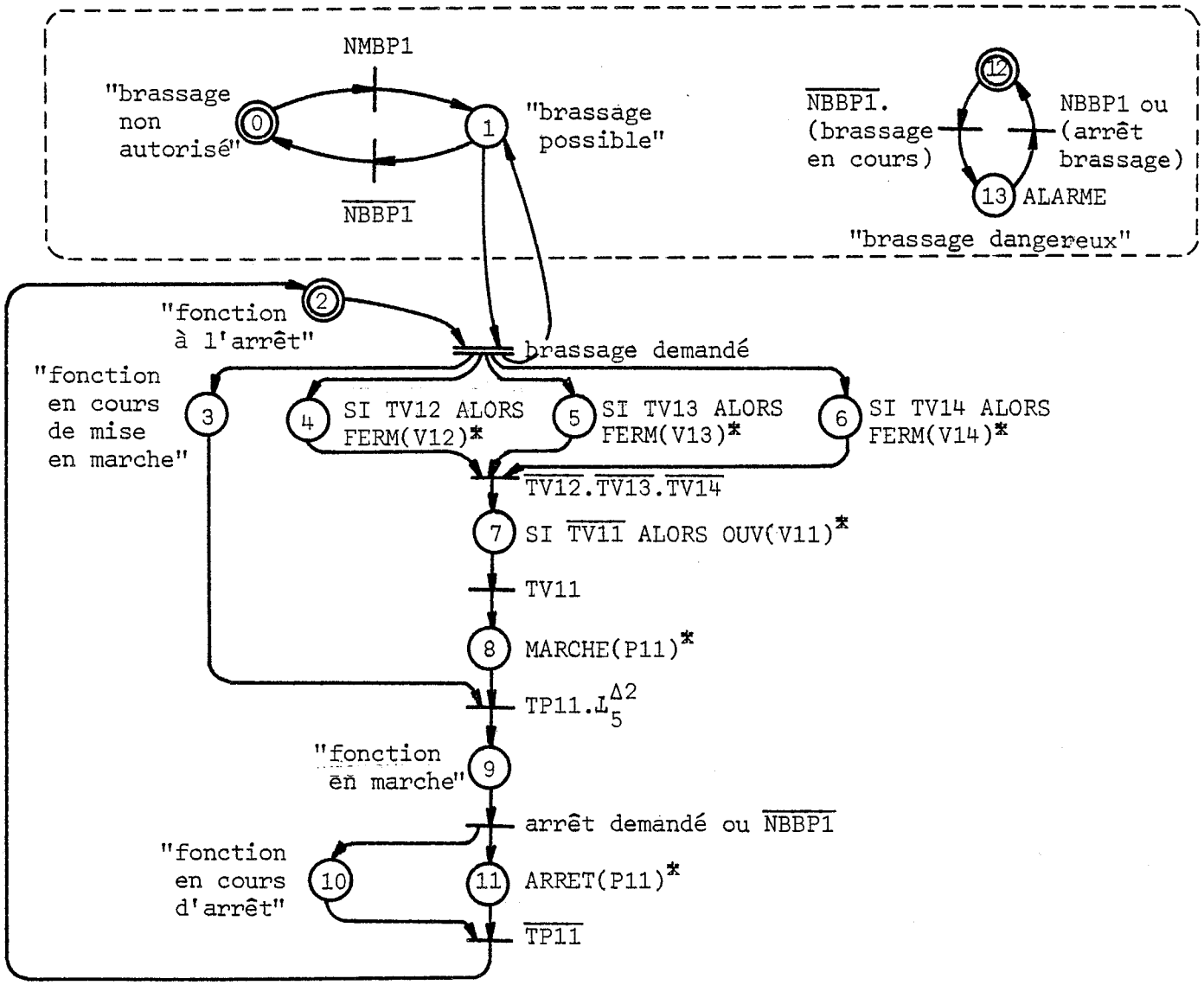
* Grafcet de commande :

Cette fonction est commandée à partir de l'unité de remélange



3) Brassage du produit dans ballon de stock

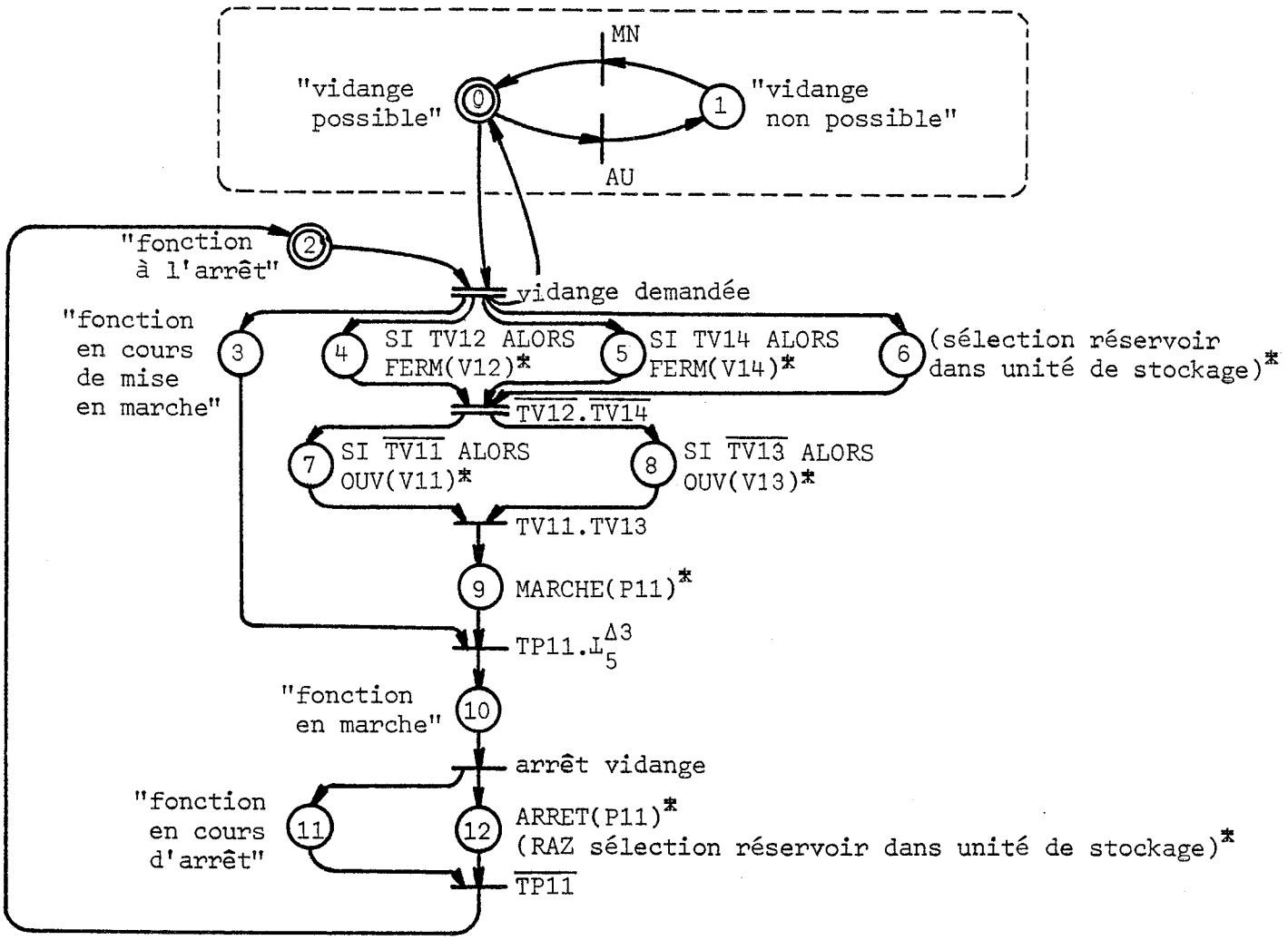
* Mode de fonctionnement : manuel/automatique
* Contraintes :
  - le brassage ne peut être effectué si, au départ, le niveau minimum NMBP1 n'est pas présent ;
  - alarme sur disparition du niveau NBBP1.
* Grafcet de commande :



4) Vidange ballon de stock en fonctionnement normal

* Mode de fonctionnement : manuel
* Contraintes : aucune

* Grafcet de commande

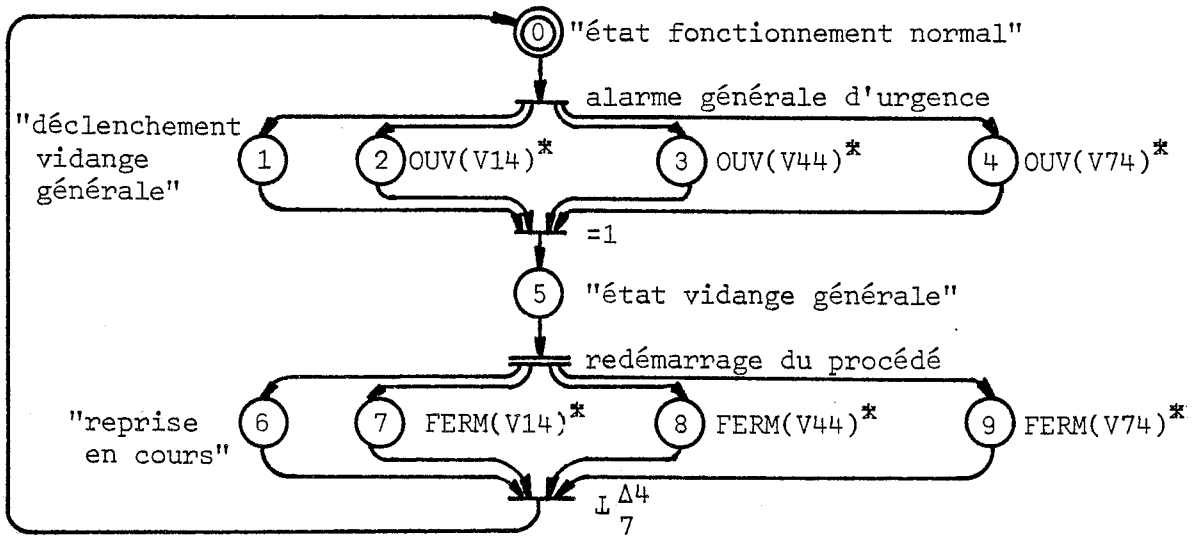


5) Vidange générale d'urgence

* Mode de fonctionnement : manuel/automatique

* Contraintes : aucune

* Grafcet de commande :



6) Mesure température du produit : (T12)

Cette mesure est effectuée en permanence dès la mise sous tension de l'unité. Le résultat de la mesure est livré sous la forme d'un signal analogique 0-10 volts.

7) Mesure température d'eau à la sortie de l'échangeur thermique : (T11), idem à 6.

8) Mesure de concentration sur boucle de brassage : (C13), idem à 6.

Contraintes de commande à respecter absolument

Quelle que soit la fonction exécutée, la pompe P11 (resp.P41, P71) ne peut être mise en marche tant que l'une au moins des électrovannes V11 ou V12 (resp. V41 ou V42, V71 ou V72) n'est pas ouverte.

.
.
.
.
.
.
.
.
.
.
.

Nous sautons ici les paragraphes :

III - 3. STRUCTURE ET MOYENS DE COMMANDE D'UNE UNITE COLONNE DE DISTILLATION

III - 4. STRUCTURE ET MOYENS DE COMMANDE DE L'UNITE DE REMELANGE

Rappelons que ces paragraphes utilisent la même démarche que dans III-2 pour décrire les structures et moyens de commande, respectivement, d'une unité colonne de distillation et de l'unité de remélange.

IV - SPECIFICATION DE L'AUTOMATISME

IV - 1. PRELUDE

On distingue quatre phases dans le fonctionnement du procédé :

- la phase de démarrage,
- la phase de fonctionnement normal,
- la phase d'arrêt,
- la phase d'alarme générale.

L'automatisation doit tenir compte des particularités de chaque phase. Elle doit aussi permettre le passage d'une commande manuelle à une commande automatisée et inversement, et ce :

. à tout moment du fonctionnement, quelle que soit la phase en cours,

. au niveau de chaque unité et de façon tout à fait indépendante de la commande des autres unités.

L'architecture modulaire du procédé et les possibilités d'interconnexion donnent plusieurs possibilités de choix de configurations de fonctionnement. Une configuration de fonctionnement est caractérisée par :

- le (les) bloc(s) de mixage mis en jeu dans l'unité de mélange en ligne;
- les unités colonne de distillation impliquées : une seule unité colonne, les deux unités colonnes mises en parallèle ou en série;
- pour chaque unité colonne impliquée, le type de recyclage de produit utilisé : pas de recyclage, recyclage du produit de tête, recyclage d'un produit intermédiaire, recyclage du produit de pied. Dans la pratique , les deux premiers types de recyclage sont plutôt rares. Le recyclage du produit de pied est le plus couramment utilisé;
- en cas de fonctionnement en série des deux unités colonnes, la façon dont il faut les connecter : alimentation de la deuxième colonne à partir du produit de tête en sortie de la première, à partir d'un produit intermédiaire ou à partir d'un produit de pied. En fonctionnement courant, l'alimentation de la deuxième colonne est faite à partir du produit de tête en sortie de la première colonne. L'alimentation à partir d'un produit intermédiaire est aussi utilisée mais beaucoup moins fréquemment. L'alimentation à partir du produit de pied n'est quasiment jamais utilisée;

- toujours en cas de fonctionnement en série des deux unités colonnes, le type de rebouclage utilisé : pas de rebouclage, rebouclage du produit de tête de la deuxième colonne vers l'entrée d'alimentation de la première colonne, rebouclage d'un produit intermédiaire, rebouclage du produit de pied;
- les paramètres d'initialisation et de fonctionnement de façon génér consignes régulateurs au démarrage, temporisations, seuils de tempé rature... (ces paramètres apparaîtront dans la description de l'automatisme).

Le choix d'une configuration de fonctionnement est généralement fait au moment du démarrage du procédé, et les commandes conséquentes sont effectuées progressivement durant cette phase. Toutefois, un changement de configuration peut être opéré à tout moment au cours du fonctionnement nor du procédé.

La spécification de l'automatisme comporte 3 parties (figure 25) :

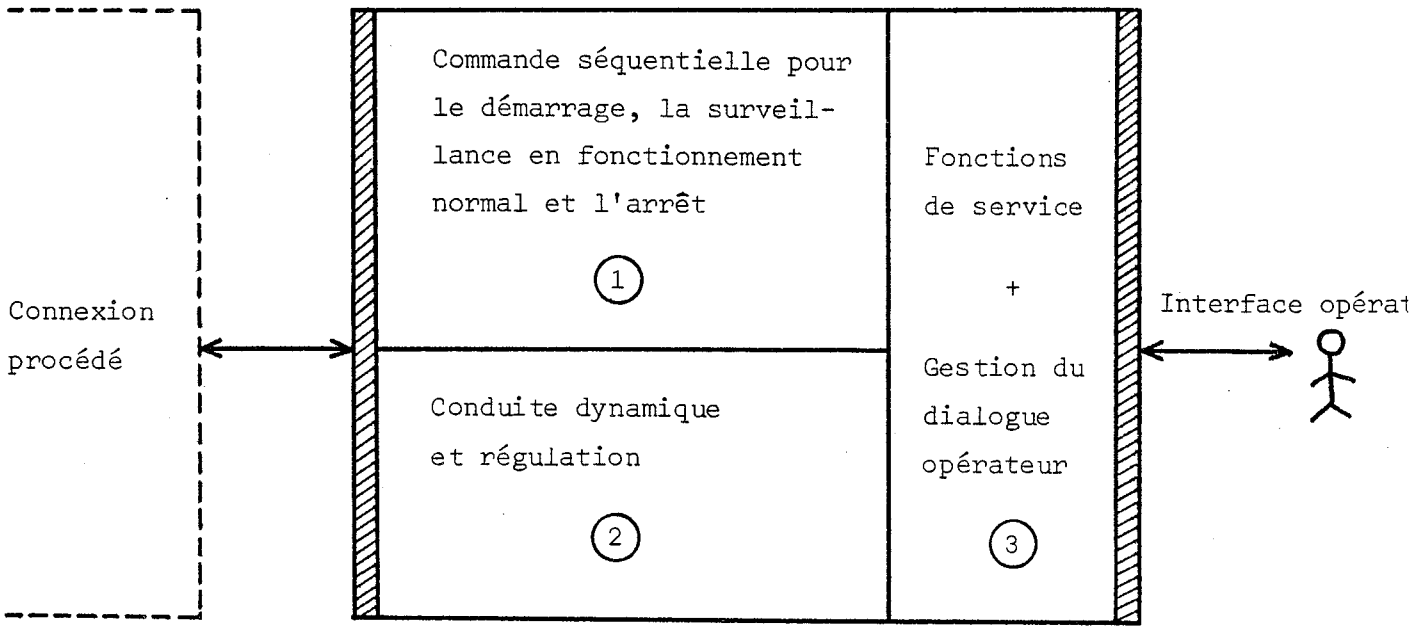


Figure 25.

1 - La partie commande séquentielle chargée de la surveillance permanente des états du procédé et de l'ordonnancement des commandes discrètes pour les diverses phases de fonctionnement.

2 - La partie concernant la commande continue. C'est la partie chargée de déterminer, de façon dynamique le long du fonctionnement du procédé, les corrections à apporter aux consignes de commande dans le but de réaliser un ensemble d'objectifs de consommation et de production (figure 26). Les fonctions de régulation peuvent être intégrées dans cette partie ou réalisées par des dispositifs autonomes.

3 - Une partie concernant la gestion du dialogue opérateur et les fonctions de service (comptabilisation des principaux événements, élaboration des bilans..) que l'automatisme doit assurer.

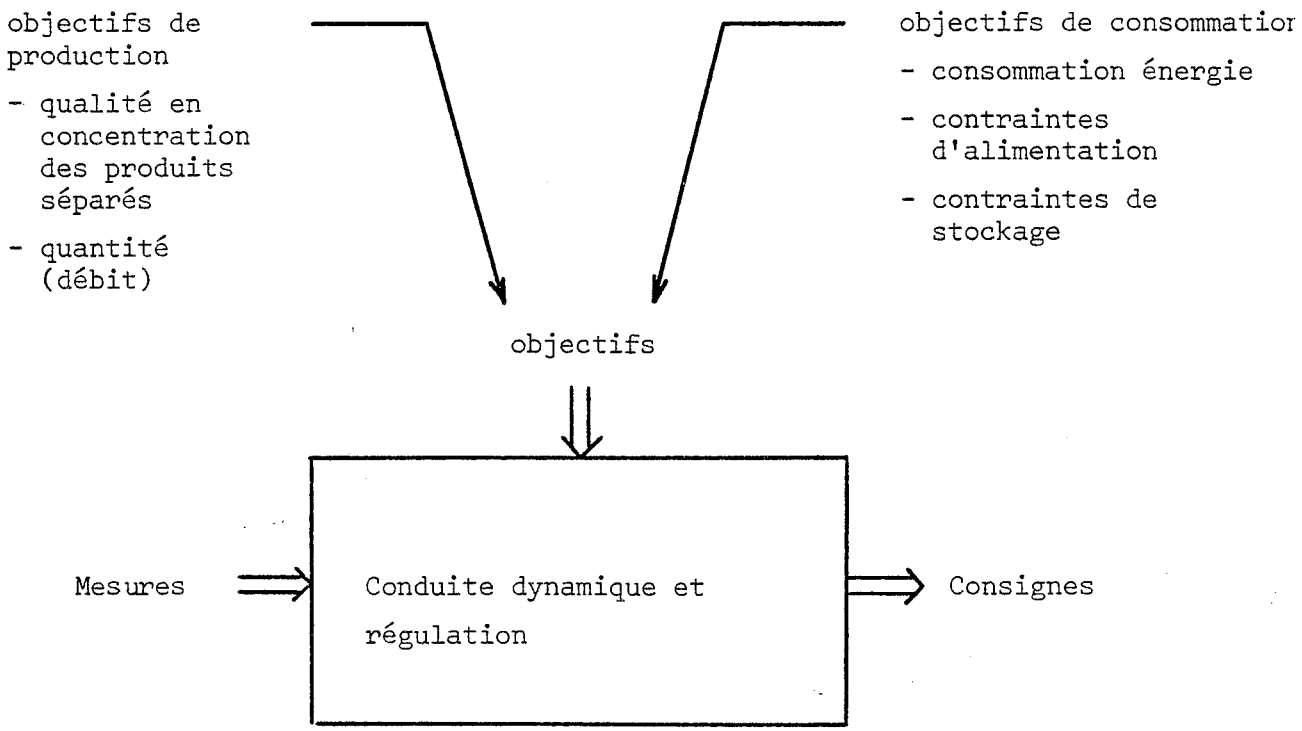


Figure 26.

Ce cahier des charges ne concerne que la partie commande séquentielle. La spécification de la partie commande continue et de la partie dialogue opérateur est donnée dans un document annexe.

L'interface de la partie commande séquentielle avec le procédé a été entièrement défini dans le paragraphe III : STRUCTURE ET MOYENS DE COMMANDE DES DIFFERENTES UNITES.

L'interface avec la partie commande continue comprend les informations suivantes :

dans le sens partie commande séquentielle → partie commande continue :

- les données définissant la configuration de fonctionnement en cours;
- les principaux états d'activité des différentes unités du procédé :
  - . unité de mélange en ligne à l'arrêt
  - . unité de mélange en ligne en cours de démarrage
  - . unité de mélange en ligne en fonctionnement normal
  - . unité colonne-1 à l'arrêt
  - . unité colonne-1 en cours de démarrage
  - . unité colonne-1 prête
  - . unité colonne-1 en fonctionnement normal
  - . unité colonne-2 à l'arrêt
  - . unité colonne-2 en cours de démarrage
  - . unité colonne-2 prête
  - . unité colonne-2 en fonctionnement normal.
- niveaux minima présents/absents dans les ballons de stock d'alimentation;
- mesure de concentration prête (mesure de concentration du produit contenu dans l'un des ballons munis d'un circuit de brassage long). Cette mesure est toujours effectuée par l'unité de remélange sous le contrôle de la commande séquentielle. Elle peut être ordonnée soit de l'extérieur par l'opérateur, soit par la partie commande continue;
- commandes de mise en marche/arrêt des régulateurs. Les fonctions de régulation peuvent être intégrées dans la partie commande continue ou réalisées par des dispositifs autonomes. Dans tous les cas, il importe à la partie commande continue de connaître l'état mis en jeu/mis hors jeu des régulateurs.

dans le sens partie commande continue → partie commande séquentielle :

- commandes de mise en marche/arrêt des unité du procédé (la partie commande continue peut intervenir pour demander la mise en service/ hors service des unités du procédé, opérant ainsi à une modification de la configuration de fonctionnement);
- commande de mesure de concentration à affectuer par l'unité de remélange;
- acquittement d'une mesure de concentration;
- état de fonctionnement normal/anormal des régulateurs.

Cette information peut être obtenue directement à partir de l'interface du procédé, au cas où les fonctions de régulation sont réalisées par des dispositifs autonomes.

Le dialogue opérateur comporte deux types d'interactions :

dans le sens partie commande séquentielle → opérateur :

- demande d'intervention : certaines situations requièrent l'intervention de l'opérateur, soit pour exécuter des fonctions qui ne peuvent être commandées que manuellement (remplissage des ballons de stock primaire ou des bouilleurs à partir de l'unité de stockage,..), soit à la suite d'une alarme (fonctionnement anormal d'un régulateur, débordement de produits dans un ballon,...). L'intervention de l'opérateur doit se terminer par un acquittement "fin intervention";

dans le sens opérateur → partie commande séquentielle :

- commandes d'exécution de fonctions particulières telles que : brassage du produit dans les ballons de stock, actions préventives de remplissage ou de vidange de ces ballons, etc;
- commandes de modification de la configuration de fonctionnement;
- commandes de mesure de concentration des produits contenus dans les ballons munis d'un circuit de brassage long. Ce sont, en fait, les seules mesures qui requièrent une action directe sur le procédé (brassage → prélèvement → mesure); les mesures de température, de débit et de concentration sont effectuées en permanence.

IV - 2. SPECIFICATION DE LA COMMANDE SEQUENTIELLE

IV - 2.1. SPECIFICATIONS FONCTIONNELLES

Ces spécifications concernent les fonctions que doit réaliser l'automatism dans les 4 phases prédéfinies. Ces fonctions sont définies distinctement pour chaque phase et représentées par des grafcets, commentés, éventuellement, par un ensemble de remarques.

SF1 - Phase de démarrage

ref.grafcet
↓

G1 - Coordination de la commande des différentes unités lors de la phase de démarrage du procédé. Il s'agit, essentiellement, d'illustrer les points de synchronisation qui interviennent dans la coordination de la commande des différentes unités du procédé.

G2 - Mise en marche/Arrêt de l'alimentation générale.

G3 - Commande de l'unité de mélange en ligne en phase de démarrage.

G4 - Commande de l'unité colonne-1 en phase de démarrage.

G5 - Commande de l'unité colonne-2 en phase de démarrage.

G6 - Commande des blocs de reflux externe.

SF2 - Phase d'arrêt normal

G7 - Commande de l'unité de mélange en ligne en phase d'arrêt normal.

G8 - Commande de l'unité colonne-1 (colonne-2) en phase d'arrêt norma

G9 - Commande de la phase d'arrêt normal des blocs de reflux externe.

SF3 - Phase de fonctionnement normal

La plupart des fonctions regroupées dans ce paragraphe sont des fonctions que l'automatisme doit assurer le long du fonctionnement du procédé, systématiquement ou sur commande opérateur, quelle que soit la phase en cours. Ces fonctions sont de trois types :

1) Des fonctions de surveillance permanente : il s'agit de surveiller les conditions limites de fonctionnement normal du procédé. Dès que l'une de ces conditions cesse d'être remplie, l'automatisme doit intervenir pour prendre les mesures d'urgence qui s'imposent avant d'en avertir l'opérateur. Ces conditions limites concernent essentiellement les niveaux maximum et minimum des produits dans les ballons et des seuils de température. Elles sont résumées dans les grafcets suivants :

G10 - Automates de surveillance des modules de l'unité de mélange en ligne.

G11 - Automates de surveillance des modules de l'unité colonne-1 (colonne-2).

G12 - Automates de surveillance des modules de l'unité de remélange.

G13 - Contraintes de sécurité intrinsèque à respecter absolument.

2) Des fonctions commandées "occasionnellement" par l'opérateur. Ce sont :

- d'une part, des fonctions touchant directement à la gestion de la configuration de fonctionnement :
  - . marche/arrêt d'une unité,
  - . modification des paramètres de la configuration de fonctionnement (consignes, électrovannes d'alimentation, seuils de température..)
  - . marche/arrêt reflux externe,
  - . recyclage/arrêt de recyclage des produits dans unités colonnes,
  - . connexion/rupture de connexion en série des deux unités colonnes,
  - . bouclage/arrêt de bouclage des produits en fonctionnement en série.
- d'autre part, certaines des fonctions qui ont été définies dans le paragraphe III (STRUCTURE ET MOYENS DE COMMANDE DES DIFFERENTES UNITES - *Fonctions assurées*)pour lesquelles on a spécifié un mode de fonctionnement automatique possible. Ces fonctions sont les suivantes

pour l'unité de mélange en ligne :

- module de stock primaire
  - . remplissage ballon de stock à partir de l'unité de remélange
  - . brassage du produit dans ballon de stock

- module de mixage
  - . mise en marche régulation débit-concentration
- module de stock secondaire
  - . brassage court du produit dans ballon de stock
  - . brassage long du produit dans ballon de stock

pour les unités colonne de distillation

- module d'interconnexion :
  - . alimentation à partir de l'unité de mélange en ligne
- module d'alimentation - bloc ballon tampon
  - . brassage court du produit dans ballon tampon
  - . brassage long du produit dans ballon tampon.
- module d'alimentation - bloc de reflux externe
  - . remplissage ballon de reflux externe
  - . vidange ballon de reflux externe dans ballon de stock secondair du produit de tête.
- module colonne à distiller
  - . soutirage produit de tête
  - . soutirage produit intermédiaire
  - . soutirage produit de pied.
- module de stock secondaire
  - . brassage court du produit dans ballon de stock
  - . brassage long du produit dans ballon de stock.

pour l'unité de remélange

- modules de remplissage, de stockage et de vidange
  - . vidange d'un bloc du module de stock secondaire de l'unité de mélange en ligne dans l'un des blocs de stockage de l'unité de remélange
  - . vidange d'un bloc du module de stock secondaire d'une unité colonne dans l'un des blocs de l'unité de remélange
  - . vidange du bloc ballon tampon d'une unité colonne dans l'un des blocs de stockage de l'unité de remélange
  - . brassage du produit dans bloc de stockage

. transvasage du produit d'un bloc de l'unité de remélange à un autre bloc
. vidange d'un bloc de l'unité de remélange dans l'un des blocs de l'unité de mélange en ligne.

- module de mesure de concentration
  . mesure de concentration du produit dans l'un des blocs de stock de l'unité de mélange en ligne ou des unités colonne à distiller.

L'automatisation de ces fonctions a pour but de réduire au maximum la nécessité pour l'opérateur d'accéder directement au procédé. Par ailleurs, certaines de ces fonctions interviennent (et sont commandées automatiquement) dans les phases de démarrage et d'arrêt normal.

Chaque commande d'exécution de l'une de ces fonctions doit évidemment être accompagnée des paramètres qui la définissent complètement (blocs concernés, régulateurs, ..) de même qu'à chaque fonction est associée une commande d'arrêt.

3) Des fonctions d'acheminement des produits entre les diverses unités. Il s'agit, d'une part, de réguler l'alimentation des ballons tampon des unités colonne, d'autre part d'assurer le rebouclage des produits depuis les ballons de sortie des unités colonne jusqu'aux ballons de stock primaire dans l'unité de mélange en ligne, en passant par l'unité de remélange.

a) Alimentation des ballons tampon des unités colonne :

L'alimentation du ballon tampon d'alimentation d'une unité colonne peut se faire, selon la configuration de fonctionnement choisie, soit directement à partir de l'unité de mélange en ligne, soit par recyclage des sorties de l'unité elle-même, soit par rebouclage des sorties de la deuxième unité colonne en fonctionnement en série. L'alimentation du ballon tampon de reflux externe se fait à partir du ballon de stock du produit de tête.

- La configuration de fonctionnement choisie détermine quels sont les électrovannes à ouvrir à l'entrée des ballons tampon pour assurer l'acheminement des produits. Dans la suite, on désignera par DVpqUCi la variable booléenne qui vaut "1" ssi la configuration de fonctionnement en cours implique l'ouverture de l'électrovanne Vpq dans l'unité colonne UCi (i=1,2).

- L'alimentation ne doit être effectuée que si l'unité colonne est en mesure de consommer. Ceci est déterminé par l'état actif ou non des étapes U18 et U116 dans l'unité colonne-1, U28 et U216 dans l'unité colonne-2.
- Afin d'assurer une meilleure distribution des produits, si deux ballons tampon sont alimentés à partir d'une même source alors l'alimentation se fera de façon prioritaire pour le ballon dans lequel le niveau de remplissage normal (NN) n'est pas atteint.

Les conditions régissant en conséquence l'ouverture ou la fermeture des électrovannes de remplissage des ballons tampon de l'unité colonne-1 sont explicitées ci-dessous. Celles se rapportant à l'unité colonne-2 se désuisent systématiquement.

OUV (V12) ssi $DV12UC1.\varphi_{U18}.\overline{NHBT1}.(\overline{NNBT1} + \overline{DV12UC2} + NNBT2 + \overline{\varphi}_{U28}$

OUV (V42) ssi $DV42UC1.\varphi_{U28}.\overline{NHBT1}.(\overline{NNBT1} + \overline{DV42UC2} + NNBT2 + \overline{\varphi}_{U28}$

OUV (V72) ssi $DV72UC1.\varphi_{U18}.\overline{NHBT1}.(\overline{NNBT1} + DV72UC2 + NNBT2 + \overline{\varphi}_{U28}$

OUV (V13) ssi $DV13UC1.\varphi_{U116}.\overline{NHBT1}.(\overline{NNBT1} + (DV15UC1 + NNBR1)(\overline{DV13UC2} + NNBT2 + \overline{\varphi}_{U28}))$

OUV (V43) ssi $DV43UC1.\varphi_{U116}.\overline{NHBT1}.(\overline{NNBT1} + \overline{DV43UC2} + NNBT2 + \overline{\varphi}_{U2}$

OUV (V73) ssi $DV73UC1.\varphi_{U116}.\overline{NHBT1}.(\overline{NNBT1} + \overline{DV73UC2} + NNBT2 + \overline{\varphi}_{U2}$

OUV (V14) ssi $DV14UC1.\varphi_{U18}.\overline{NHBT1}.(\overline{NNBT1} + (\overline{DV14UC2} + NNBT2 + \overline{\varphi}_{U216})(\overline{DV15UC2} + NNBR2 + \overline{\varphi}_{U216}))$

OUV (V44) ssi $DV44UC1.\varphi_{U18}.\overline{NHBT1}.(\overline{NNBT1} + \overline{DV44UC2} + NNBT2 + \overline{\varphi}_{U21}$

OUV (V74) ssi $DV74UC1.\varphi_{U18}.\overline{NHBT1}.(\overline{NNBT1} + \overline{DV74UC2} + NNBT2 + \overline{\varphi}_{U21}$

OUV (V15) ssi $DV15UC1.\varphi_{U116}.\overline{NHBT1}.(\overline{NNBT1} + (\overline{DV13UC1} + NNBT1)(\overline{DV13UC2} + NNBT2 + \overline{\varphi}_{U28}))$

b) Rebouclage des produits vers l'unité de mélange en ligne.

Les ballons de stock dans l'unité de remélange sont banalisés. En effet, les circuits d'interconnexion permettent :

- le remplissage de n'importe lequel de ces ballons à partir de n'importe lequel des ballons de stock secondaire ou des ballons tampon d'alimentation des unités colonne;

- le transvasage des produits de l'un quelconque de ces ballons dans un autre;
- la vidange de n'importe lequel de ces ballons dans n'importe lequel des ballons de stock primaire dans l'unité de mélange en ligne.

On peut prévoir un mode de fonctionnement automatique (il s'agit d'une option qui reste à confirmer) dans lequel chacun des trois ballons est réservé exclusivement pour la collecte de l'un des trois produits (de tête, intermédiaire ou de pied) en sortie des unités colonne. Les produits collectés sont ensuite acheminés vers les ballons correspondants dans l'unité de mélange en ligne. On peut convenir, par exemple, que le produit dépassant le niveau normal dans les ballons de sortie des unités colonne est renvoyé dans le ballon réservé dans l'unité de remélange. Celui-ci renverrait, à son tour, le produit vers le ballon correspondant de l'unité de mélange en ligne si le niveau haut dans ce ballon n'est pas atteint.

SF4 - Phase d'arrêt d'urgence

L'alarme générale est déclenchée par l'opérateur et provoque systématiquement la vidange d'urgence des blocs de stockage de produit dans les différentes unités. Les séquences de commande de la vidange d'urgence ont été définies dans le paragraphe III (STRUCTURE ET MOYENS DE COMMANDE DES DIFFERENTES UNITES).

.
.
.
.
.
.

Nous ne reprenons de la suite que les grafcets suivants :

G1 : Coordination de la commande des différentes unités lors de la phase de démarrage du procédé

G3 : Grafcet de commande de l'unité de mélange en ligne en phase de démarrage

G7 : Grafcet de la macroétape d'arrêt normal dans le grafcet de commande de l'unité de mélange en ligne (G3)

G10 : Automate de surveillance d'un bloc du module de stock primaire de l'unité de mélange en ligne

G13 : Grafcets des contraintes de sécurité fonctionnelle à respecter absolument

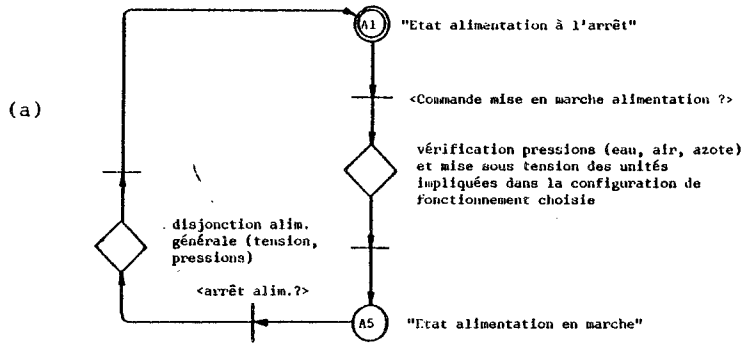
Le grafcet G1 illustre l'utilisation des macroétapes et pseudo-macroétapes pour donner une vue synthétique de l'ensemble de la commande en phase de démarrage. Les étapes A5, M7, U18, U116, U28 et U216 jouent un rôle important dans la coordination de la commande des différentes unités (Exemple : aucune unité (grafcets b, c et d) ne peut démarrer tant que l'étape A5 (grafcet a) n'a pas été activée).

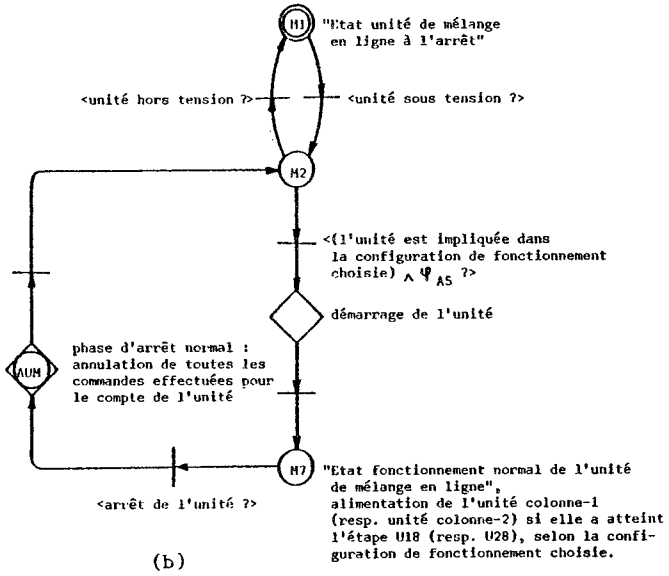
Le grafcet G3 illustre une façon de passer d'un grafcet de niveau "0" à un grafcet de niveau "1" en développant les pseudo-macroétapes.

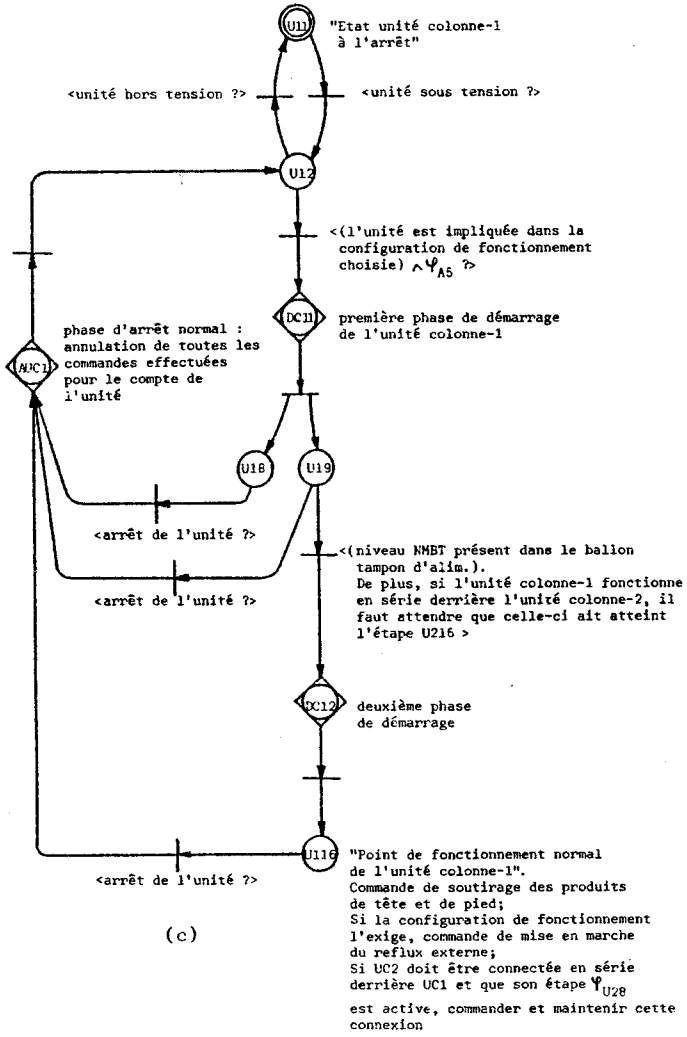
Le grafcet G7 donne un exemple simple de développement d'une macro-étape.

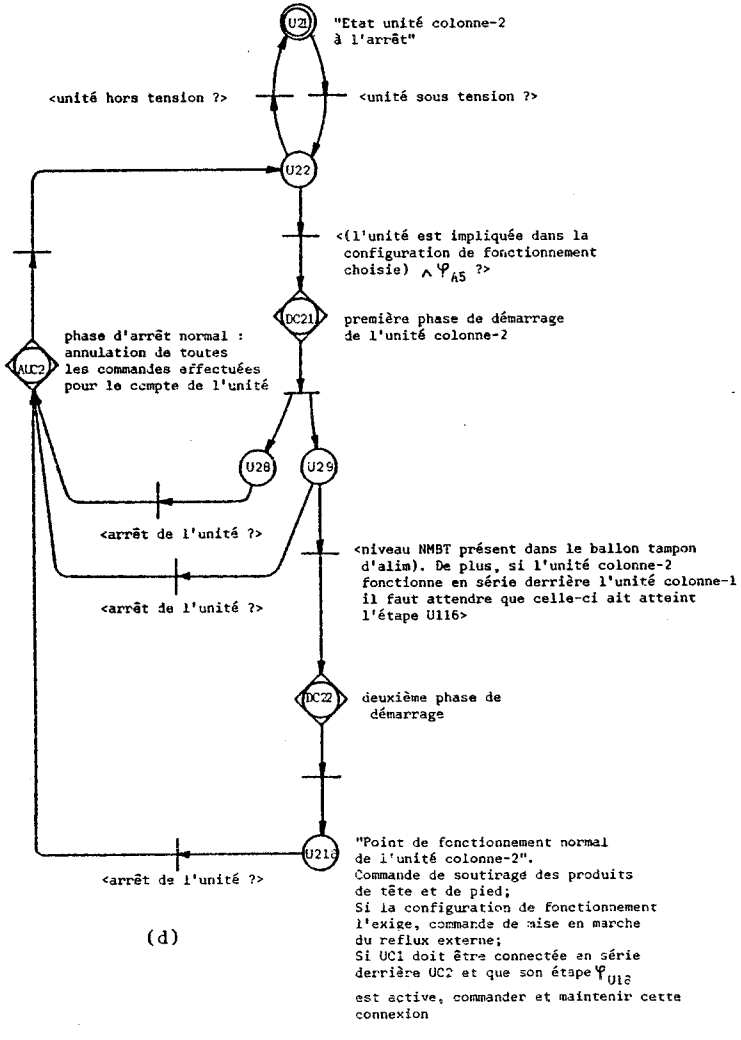
Le grafcet G10 donne un exemple de description des sécurités fonctionnelles sous la forme d'un automate de surveillance.

Enfin, le grafcet G13 résume un ensemble de sécurités fonctionnelles liées de façon intrinsèque aux normes d'exploitation normales du procédé.









G1. Coordination de la commande des différentes unités lors de la phase de démarrage du procédé.

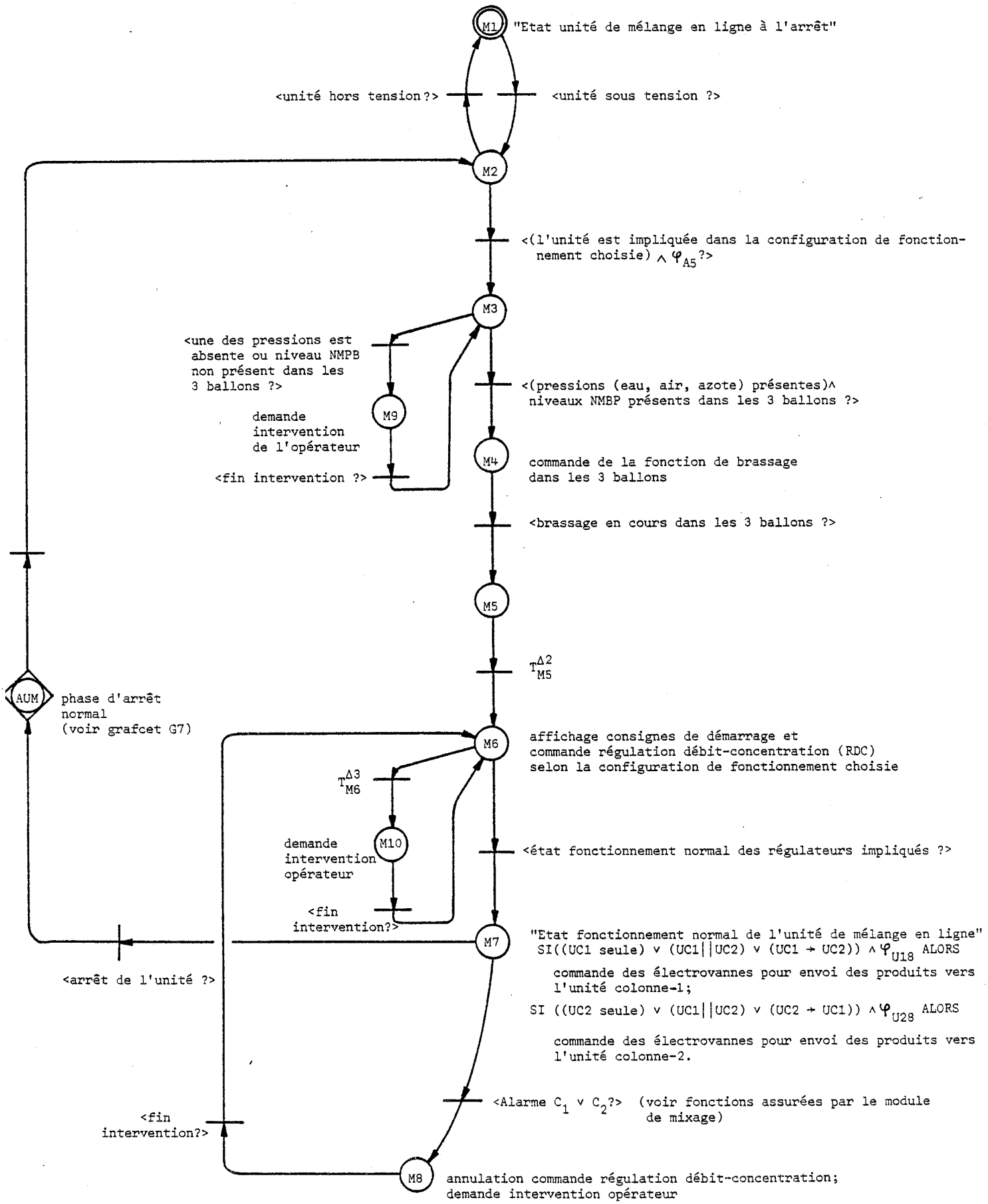
Remarques

1. La phase de démarrage ne concerne que l'unité de mélange en ligne et le unités colonne de distillation. L'unité de remélange est activée occasi nellement, au cours du fonctionnement du procédé, pour réaliser l'une des fonctions décrites en III-4 (prélèvements pour mesure de concentration, vidanges).

2. La mise en marche et l'arrêt des alimentations (tension, pressions), tant au niveau global (alim. générale) qu'au niveau local des unités, n peuvent être effectuées que manuellement par l'opérateur. La commande proprement dite de la phase de démarrage des trois unités (grafcets b, c et d), doit pouvoir s'effectuer dans les deux modes manuel et automatique. La mise hors tension annule tout effet de commande et ramène t les éléments (électrovannes, pompes, régulateurs,...) à l'état dans leq ils doivent se trouver à l'instant initial du démarrage.

3. L'événement <arrêt de l'unité ?> correspond à une commande de mise hor service (provisoire ou définitive) de l'unité, impliquant une modificat de la configuration de fonctionnement. Cet événement peut survenir à to moment du fonctionnement et engage la phase d'arrêt de l'unité,quelle que soit l'étape en cours.

4. De la façon dont est organisée la commande de chaque unité, un changement de configuration peut intervenir à tout moment de la phase de déma rage et de la phase de fonctionnement normal.



C_1 = conditions de brassage non remplies ou brassage non en cours

C_2 = alarme "niveau maximum stock de mélange" (voir fonctions assurées par le module de mixage)

G3. Grafcet de commande de l'unité de mélange en ligne lors de la phase de démarrage

Remarques

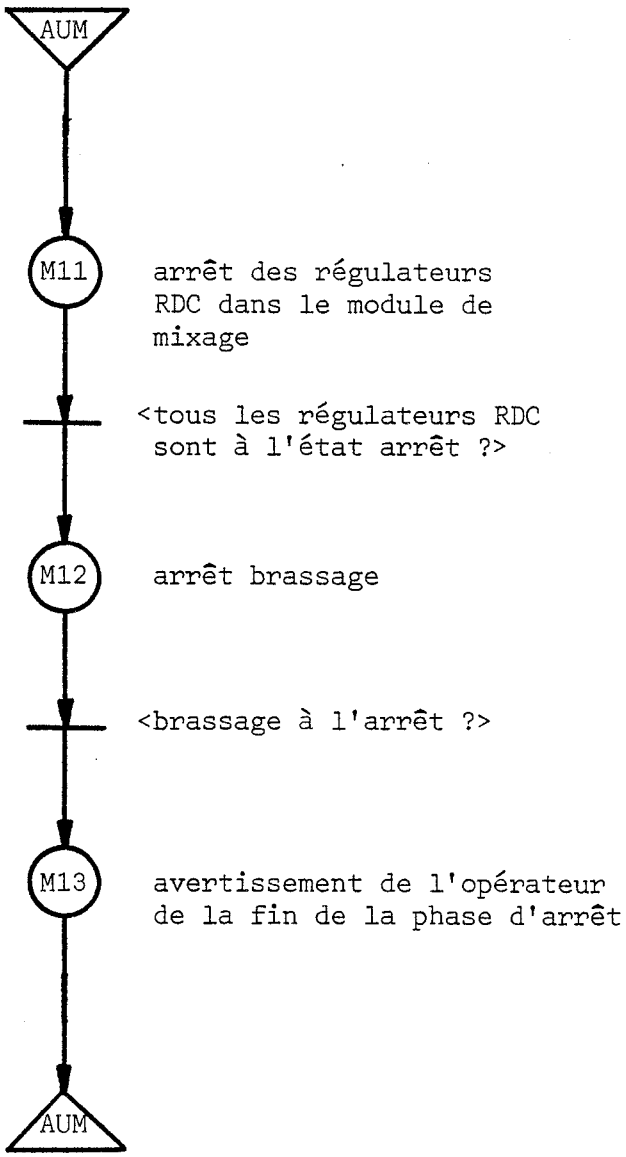
1. L'unité de mélange en ligne ne peut être démarrée tant que la phase de mise en marche de l'alimentation n'a pas atteint l'étape A5.

2. L'alimentation du ballon tampon de l'unité colonne-1 (resp. ballon tamp de l'unité colonne-2) à partir de l'unité de mélange en ligne ne peut être effectuée tant que l'unité réceptrice n'a pas atteint l'étape U18 (resp.U28).

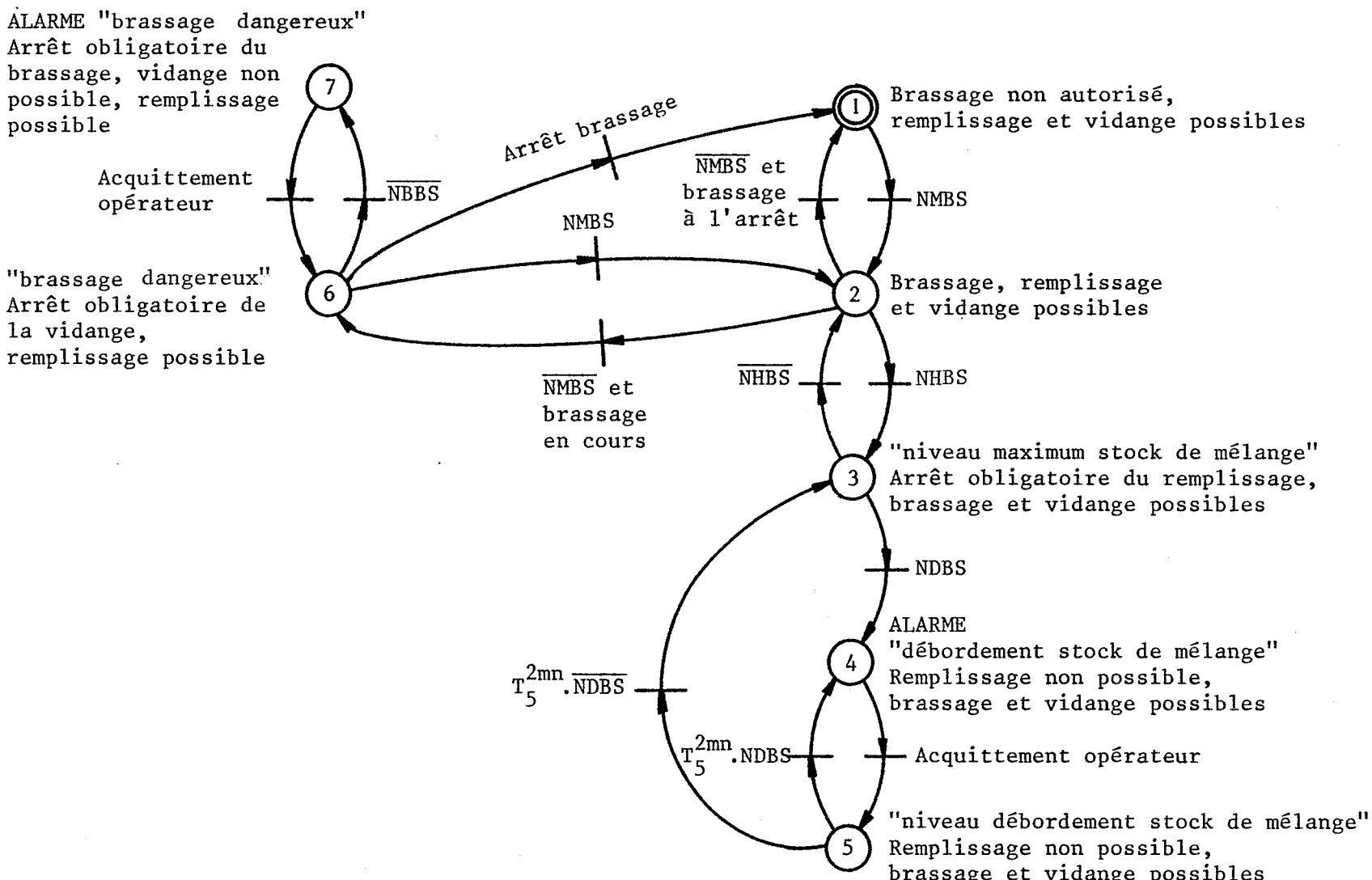
3. L'événement <arrêt de l'unité> engage l'unité dans la phase d'arrêt (AU quelle que soit l'étape où elle se trouve (excepté les étapes M1 et M2).

4. Les valeurs des temporisations $\Delta 2$ et $\Delta 3$ font partie des paramètres de fonctionnement et sont ajustables.

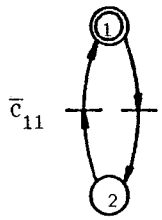
5. Le test <((UC1 seule) $\vee$ (UC1||UC2) $\vee$ (UC1 $\rightarrow$ UC2)) $\wedge \varphi_{U18}$?> signifie : l'unité colonne-1 travaille seule, ou en parallèle avec l'unité colonne ou en série devant l'unité colonne-2, et elle a atteint l'étape φ_{U18}.



G7. Grafcet de la macroétape d'arrêt normal dans le grafcet de commande de l'unité de mélange en ligne (G3)



G10. Automate de surveillance du bloc de stock de mélange



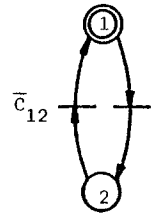
ne peut être mise en marche
/doit être arrêtée

C_{11}

pompe P11 (resp. P41,P71)
peut être mise en marche

C_{11} = TV11 ∨ TV12 (resp. V41 ∨ V42, V71 ∨ V72)

(a). Unité de mélange en ligne.
Module de stock primaire.



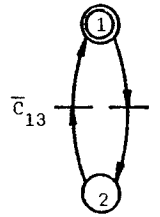
les régulateurs RDC0, RDC1 et RDC2
ne peuvent être mis en marche
/doivent être arrêtés

C_{12}

les régulateurs RDC0, RDC1 et RDC2
peuvent être mis en marche

C_{12} = TP11 ∧ TP41 ∧ TP71

(b). Unité de mélange en ligne.
Module de mixage.



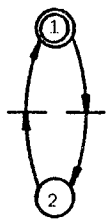
pompe P10 (resp. P40 , P70)
ne peut être mise en marche
/doit être arrêtée

C_{13}

pompe P10 (resp. P40, P70)
peut être mise en marche

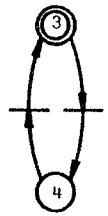
C_{13} = TV10 ∨ TV100 (resp. TV40 ∨ TV400, TV70 ∨ TV700)

(c). Unité de mélange en ligne.
Module de stock secondaire.



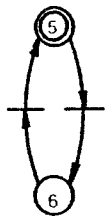
pompe P12 ne peut
être mise en marche
/ doit être arrêtée

pompe P12 peut être
mise en marche



régulation de préchauffage
non autorisée

régulation de préchauffage
autorisée



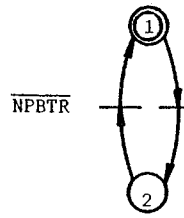
régulation de débit
non autorisée

régulation de débit
autorisée

C'_3 = (l'une au moins des électrovannes
V17, V18 et V19 est ouverte) ∨ $\overline{NPBTA}$

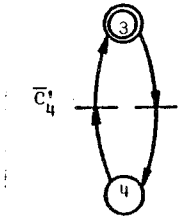
(d). Unité colonne de distillation
Module d'alimentation.
Bloc ballon tampon.

G13. Grafcets des contraintes de sécurité
fonctionnelle à respecter absolument.



régulation de préchauffage
non autorisée

régulation de préchauffage
autorisée

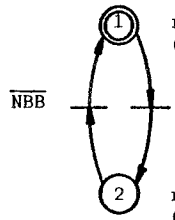


régulation de débit
non autorisée

régulation de débit
autorisée

C'_4 = (l'une au moins des électrovannes
V16 et V20 est ouverte) ∨ $\overline{NPBTR}$

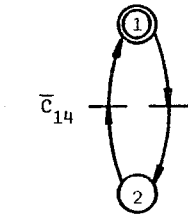
(e). Unité colonne de distillation
Module d'alimentation.
Bloc de reflux externe.



régulation puissance de chauffe
(bouilleur) non autorisée

régulation puissance de chauffe
(bouilleur) autorisée

(f). Unité colonne de distillation.
Module colonne à distiller.



pompe P10 (resp. P40, P70)
ne peut être mise en marche
(/doit être arrêtée)

C_{14}

pompe P10 (resp. P40, P70)
peut être mise en marche

C_{14} = TV10 ∨ TV100 (resp. TV40 ∨ TV400, TV70 ∨ TV700)

(g) Unité colonne de distillation
Module de stock secondaire

PARTIE II

LES RESEAUX DE PETRI ET LEUR UTILISATION POUR LA DESCRIPTION ET L'ANALYSE DES SYSTEMES DISCRETS A EVOLUTIONS PARALLELES

Cette deuxième partie comporte quatre chapitres et est consacrée à l'étude des réseaux de PETRI (RdP) en tant que modèle pour la description et l'analyse des systèmes au cours de leur conception.

Rappelons que la définition du grafcet s'est beaucoup inspirée des principes de fonctionnement des RdP.

Les réseaux de PETRI portent le nom de leur auteur, C.A. PETRI, qui en a donné les définitions de base dans sa thèse en 1962 [PET.62]. Ces réseaux sont introduits comme un nouveau modèle pour la représentation graphique du cheminement de l'information dans les systèmes asynchrones. Mais ce n'est que beaucoup plus tard, grâce notamment aux travaux conduits par A.W. HOLT [HOL.68] [HOL.70] et ceux développés dans le cadre du projet MAC au MIT [PAT.70] [HAC.72] [RAM.73], que ces réseaux ont commencé à être utilisés de façon plus large pour la modélisation et l'analyse du comportement des systèmes à évolutions parallèles.

Les travaux de HOLT, comme la plupart des premiers travaux qui les ont suivis, ont surtout utilisé les RdP pour décrire le fonctionnement de systèmes "asynchrones" et "autonomes", selon une démarche analogue à celle ayant utilisé autrefois les graphes d'états pour représenter les états d'un système et les transitions possibles entre ces états. Dans cette utilisation, ni le facteur temps, ni l'interprétation associée aux états n'interviennent dans la relation de transition entre états. On parle de RdP "autonomes" (ou de RdP à fonctionnement autonome). Par rapport aux graphes d'états, les RdP permettent une représentation condensée des états tout en facilitant l'expression de la relation entre états.

Plusieurs recherches ont été consacrées, dans le cadre de ces travaux, à l'exploration des propriétés mathématiques de ce modèle et à l'étude de différentes variantes de fonctionnements autonomes des RdP ; ces variantes étant comparées entre elles en considérant les RdP comme des automates générateurs de langages.

De façon particulière en France, on s'est intéressé, à partir du début des années 70, à utiliser des extensions des RdP pour la description de systèmes non-autonomes ; c'est-à-dire, des systèmes dont le fonctionnement doit prendre en compte des considérations temporelles, et dont les évolutions sont soumises à des contraintes dépendant d'une

interprétation associée aux états et d'une interaction avec l'environnement externe. Les RdP Interprétés (RdPI), que nous avons choisis d'utiliser comme modèle de travail, cumulent ces trois extensions : temporisation, synchronisation avec l'environnement externe et interprétation.

Dans le premier chapitre de cette deuxième partie (chapitre IV du mémoire), nous allons rappeler les définitions de base des RdP autonomes et leurs propriétés caractéristiques, puis nous étudierons quelques variantes intéressantes dérivées des RdP autonomes.

Dans le chapitre V, nous étudierons les RdP non-autonomes. Nous introduirons progressivement les différentes extensions qui nous ont conduits à la définition du modèle RdPI. Nous montrerons, au fur et à mesure, les modifications que ces extensions induisent sur les propriétés caractéristiques des réseaux.

Il faut souligner ici que si plusieurs variantes de fonctionnement des RdP (autonomes et non-autonomes) sont aujourd'hui utilisées pour la modélisation des systèmes discrets et pour l'élaboration d'outils de conception, peu de ces variantes ont fait l'objet d'une formalisation mathématique rigoureuse. Cette formalisation est cependant nécessaire pour vérifier la cohérence de ces variantes et définir les hypothèses de base de toute investigation théorique. Nous lui accorderons, dans ces chapitres, le plus grand intérêt.

Dans le chapitre VI, nous aborderons le problème de l'analyse et de la vérification des systèmes décrits par les RdP. Ce problème a aussi suscité de nombreuses recherches, et l'on dispose aujourd'hui de résultats intéressants à ce sujet. Cependant, la plupart de ces résultats ont été démontrés pour le cas des RdP autonomes, et leur application aux RdP non-autonomes n'est que partielle. Nous ferons le bilan de ces résultats et nous étudierons sous quelles conditions l'application de ces résultats peut être étendue aux cas des RdP non-autonomes.

Nous terminerons, dans le chapitre VII, par une comparaison des modèles GRAFCET et RdPI.

CHAPITRE IV

RÉSEAUX DE PETRI AUTONOMES

I - DEFINITIONS DE BASE 3

I - 1. RESEAUX DE PETRI ORDINAIRES 3

I - 2. SOUS-CLASSES PARTICULIERES DE RdP ORDINAIRES 3

I - 3. PROPRIETES D'UN RdP 4

I - 3.1. Marquages d'un RdP 4

I - 3.2. Opérations sur les marquages 5

I - 3.3. Propriétés liées aux marquages d'un RdP 6

II - PRINCIPALES EXTENSIONS DES RdP AUTONOMES 8

II - 1. RESEAUX DE PETRI A ARCS INHIBITEURS 8

II - 2. RESEAUX DE PETRI GENERALISES 9

II - 3. RESEAUX DE PETRI A PRIORITES 9

III - RELATION D'EQUIVALENCE SUR LES RdP 10

. *Réseau de Petri étiqueté* 10

. *Relation de réalisation* 10

. *Relation d'équivalence* 11

. *Propositions.* 11

I - DEFINITIONS DE BASE

I - 1. RESEAUX DE PETRI ORDINAIRES [HOL.70] [HAC.72] [MOA.78]

Définition :

Un réseau de PETRI (RdP) est un quadruplet $R = (P,T,\alpha,\beta)$ où :

- P est un ensemble d'objets appelés places, $P \neq \emptyset$
- T est un ensemble d'objets appelés transitions, $T \neq \emptyset$ et $P \cap T = \emptyset$
- $\alpha \subseteq P \times T$ est une relation d'incidence avant
- $\beta \subseteq T \times P$ est une relation d'incidence arrière

Représentation :

On représente un RdP par un graphe biparti orienté. Les noeuds du graphe sont les places et les transitions, représentées respectivement par des cercles et des barres (figure 1). Il y a un arc de la place p_r à la transition t_i ssi $(p_r,t_i)\in\alpha$. De même, il y a un arc de la transition t_j à la place p_s ssi $(t_j,p_s)\in\beta$.

Conventions de notation :

Les notations suivantes sont adoptées :

a) $t \in T$, ${}^\bullet t = \{p \in P \mid (p,t) \in \alpha\}$
   ${}^\bullet t$ est dit ensemble des places d'entrée de t

b) $t \in T$, $t^\bullet = \{p \in P \mid (t,p) \in \beta\}$
   $t^\bullet$ est dit ensemble des places de sortie de t

c) $p \in P$, ${}^\bullet p = \{t \in T \mid (t,p) \in \beta\}$
   ${}^\bullet p$ est dit ensemble des transitions d'entrée de p

d) $p \in P$, $p^\bullet = \{t \in T \mid (p,t) \in \alpha\}$
   $p^\bullet$ est dit ensemble des transitions de sortie de p.

I - 2. SOUS-CLASSES PARTICULIERES DE RdP ORDINAIRES

- Un réseau de Petri simple est un RdP tel que :

$$\forall t_i \in T, \left| \bigcup_{t_j \in T,\, j \neq i} ({}^\bullet t_i \cap {}^\bullet t_j) \right| \leq 1$$

autrement dit, tel que toute transition a au plus une place d'entrée qui est partagée avec d'autres transitions.

- Un réseau de Petri libre choix est un RdP tel que :

$$\forall p \in P \text{ , si } |p^\bullet| > 1 \text{ alors } \forall\, t_i \in p^\bullet,\ |{}^\bullet t_i| = 1$$

autrement dit, tel que si deux transitions t_i et t_j ont en commun une

place d'entrée p alors p est la seule place d'entrée de t_i et t_j.

- Un graphe de transitions (ou graphe marqué [HOL.68]) est un RdP tel que :

$\forall p \in P, |{}^\bullet p| \leq 1$ et $|p^\bullet| \leq 1$

autrement dit, tel que toute place a au plus une transition d'entrée et une transition de sortie.

- Un graphe d'états est un RdP tel que :

$\forall t \in T, |{}^\bullet t| \leq 1$ et $|t^\bullet| \leq 1$

Nous verrons, dans le chapitre VI, que les restrictions qui caractérisent ces sous-classes de RdP facilitent l'analyse et la vérification des systèmes que ces sous-classes permettent de décrire.

I - 3. PROPRIETES D'UN RdP

I - 3.1. Marquages d'un RdP

Un marquage M d'un RdP $R = (P,T,\alpha,\beta)$ est une application de P dans $\mathbb{N}$, $\mathbb{N}$ étant l'ensemble des entiers naturels.

Si n est le nombre de places de R, ($|P| = n$), un marquage M de R peut être défini par la donnée d'un vecteur de $\mathbb{N}^n$. On note

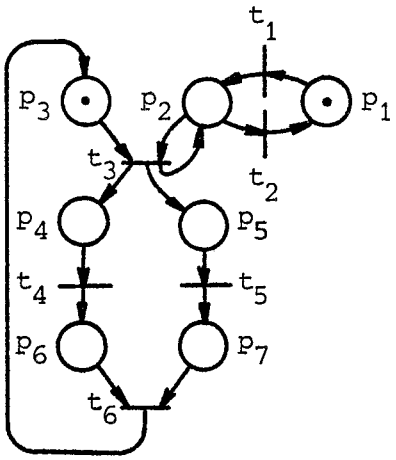
$$M = \begin{bmatrix} m_1 \\ m_2 \\ \vdots \\ m_n \end{bmatrix} \quad \text{et } m_i = M(p_i)$$

Sur le graphe associé à R, on peut représenter un marquage M par une distribution dans les places d'objets appelés marques. Une marque est représentée par un point ; chaque place p_i aura donc m_i marques (figure 1).

Une transition t de R est dite validée par le marquage M ssi : $\forall p \in {}^\bullet t$, on a $M(p) \geq 1$; autrement dit, ssi toute place d'entrée de t contient au moins une marque.

$\alpha = \{(p_1,t_1),(p_2,t_2),(p_2,t_3),$
$(p_3,t_3),(p_4,t_4),(p_5,t_5),$
$(p_6,t_6),(p_7,t_6)\}$

$\beta = \{(t_1,p_2),(t_2,p_1),(t_3,p_2),$
$(t_3,p_4),(t_3,p_5),(t_4,p_6),$
$(t_5,p_7),(t_6,p_3)\}$



$$M = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

t_1 est validée par M

Figure 1.

I - 3.2. Opérations sur les marquages

Soient M l'ensemble des marquages de R, et t une transition de R. On définit la mise à feu de t comme étant l'opération qui à un marquage $M_i \in M$ fait correspondre le marquage M_j tel que :

$$\forall p \in P,\ M_j(p) = \begin{cases} M_i(p)-1 \text{ si } p \in {}^\bullet t - t^\bullet \\ M_i(p)+1 \text{ si } p \in t^\bullet - {}^\bullet t \\ M_i(p) \text{ sinon} \end{cases}$$

On note $MAF_t(M_i) = M_j$. L'opération MAF_t est définie seulement pour les marquages M qui valident t.

Sur la représentation graphique associée à R avec un marquage M, l'opération MAF_t consiste à enlever une marque de toute place d'entrée de t et à déposer une marque dans toute place de sortie de t. Cette opération n'est possible que si chaque place d'entrée de t contient au moins une marque.

On désigne par T^* l'ensemble des séquences construites à partir de symboles de T. Soient $\sigma = t_{i_1}\ t_{i_2} \ldots t_{i_s}$, $\sigma \in T^*$, et M_0 un marquage de R. On dira que σ est une séquence de simulation ou séquence de mises à feu à partir de M_0 ssi il existe une suite de marquages $M_1, M_2, \ldots, M_s$ telle que :

$$\forall j = 1,2,\ldots s,\ MAF_{t_{i_j}}(M_{j-1}) = M_j$$

On note $M_0 \overset{\sigma}{\rightarrow} M_s$. M_s est le marquage atteint par application de σ à partir de M_0.

La séquence $\sigma = \lambda_T, \lambda_T$ mot vide de T^*, est dite séquence de simulation vide. On note $M_0 \xrightarrow{\lambda_T} M_0$.

On appelle classe des marquages successeurs de M_0 l'ensemble $\vec{M}_0 = \{M \in M \mid \exists\, \sigma \in T^* \text{ et } M_0 \overset{\sigma}{\rightarrow} M\}$

I - 3.3. Propriétés liées aux marquages d'un RdP

Soient M_0 un marquage de R, et p et t respectivement une place et une transition de R.

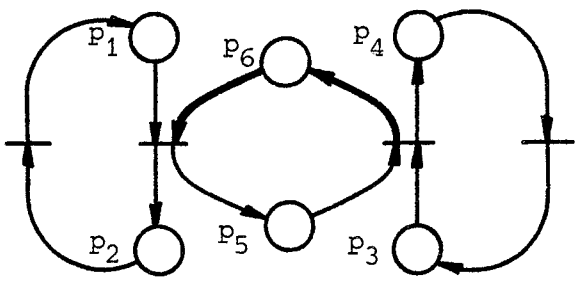
Propriété de borné :

- On dit que la place p est bornée pour M_0 ssi $\exists\, k \in \mathbb{N}$ tel que : $\forall M \in \vec{M}_0$, on a $M(p) \leq k$; on appelle alors borne de la place p le plus petit entier k vérifiant cette propriété.

- On dit que R est borné pour M_0 ssi toutes ses places sont bornées pour M_0.

- Le réseau R est dit sauf pour M_0 ssi toutes ses places ont une borne inférieure ou égale à 1.

Exemples : Le réseau donné par la figure 2 (sans la partie en trait fort) n'est pas borné pour le marquage M_0 : la place p_5 n'est pas bornée pour ce marquage. Le réseau complet est borné pour le marquage M_1, mais il n'est pas sauf pour ce marquage : la borne de la place p_5, comme celle de la place p_6, est égale à 2. Le réseau complet est borné et sauf pour le marquage M_2.



$$M_0 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}, \quad M_1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

Figure 2.

Propriété de vivacité :

- On dit que la transition t est vivante pour M_0 ssi : $\forall M \in \vec{M}_0$, il existe une séquence de mises à feu à partir de M qui contient t.

- On dit que R est vivant pour M_0 ssi toutes ses transitions sont vivantes pour M_0.

Exemples : Le réseau complet de la figure 2 est vivant pour les marquages M_1 et M_2. Le réseau donné par la figure 3 n'est vivant pour aucun marquage : quel que soit le marquage considéré, on peut toujours trouver une séquence de simulation qui rassemble toutes les marques dans la place p_2.

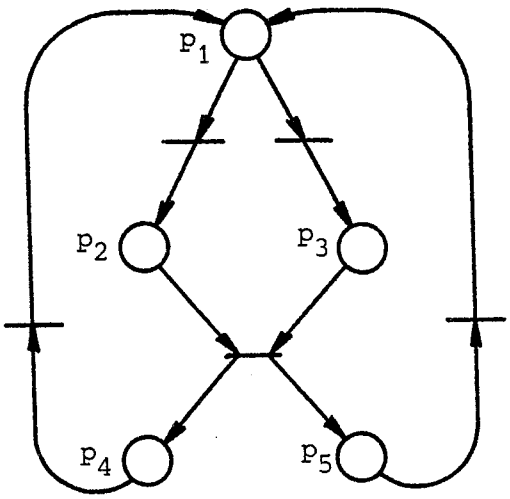
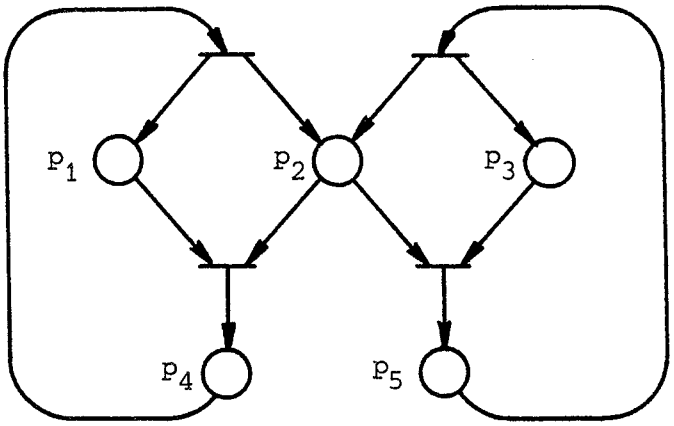


Figure 3.

Propriété de persistance :

- Le réseau R est dit persistant pour M_0 ssi : si deux transitions quelconques t_i et t_j sont validées par $M \in \vec{M}_0$ alors les opérations MAF_{t_i} et MAF_{t_j} peuvent être exécutées l'une après l'autre, dans n'importe quel ordre, à partir de M.

Exemples : Le réseau donné par la figure 4 n'est pas persistant pour le marquage M_2, mais il est persistant pour le marquage M_3.



$$M_2 = \begin{bmatrix} 1 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \quad , \quad M_3 = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

Figure 4.

II - PRINCIPALES EXTENSIONS DES RdP AUTONOMES

Pour bon nombre d'applications à la modélisation des systèmes autonomes, les RdP ordinaires se sont avérés insuffisants [AGE.73] [KOS.73]. C'est ainsi que certaines extensions ont été étudiées dont nous allons rappeler les trois principales.

II - 1. RESEAUX DE PETRI A ARCS INHIBITEURS (RdPZ) [AGE.73]

Définition :

Un RdPZ est un doublet $R_Z = (R, I)$ où :

. R est un RdP (P,T,α,β), et

. $I \subseteq P \times T$ est un ensemble d'arcs inhibiteurs, $I \cap \alpha = \emptyset$.

Sur la représentation graphique associée à un RdPZ, on distinguera les arcs inhibiteurs des arcs ordinaires dans R en plaçant un petit cercle aux extrémités incidentes aux transitions (voir exemple en figure 5).

Opérations sur les marquages :

Une transition t est validée par un marquage M ssi chaque place d'entrée reliée à t par un arc ordinaire contient au moins une marque et chaque place d'entrée reliée à t par un arc inhibiteur est vide. L'opération mise à feu de t est définie pour M ssi M valide t, et son exécution consiste à enlever une marque de toute place d'entrée reliée par un arc ordinaire et à déposer une marque dans toute place de sortie.

II - 2. RESEAUX DE PETRI GENERALISES (RdPG) [HAC.75]

Définition :

Un RdPG est un doublet $R_G = (R,W)$ où :

- R est un RdP (P,T,α,β), et
- W est une application de l'ensemble $U = \alpha \cup \beta$ des arcs de R dans $\mathbb{N}-\{0\}$.

Sur la représentation graphique associée à un RdPG, chaque arc $u \in U$ porte comme renseignement la valeur W(u). Cette valeur est appelée poids de l'arc. (voir exemple en figure 6).

Opérations sur les marquages :

Une transition t est validée par un marquage M ssi chacune de ses places d'entrée contient un nombre de marques au moins égal au poids de l'arc qui la connecte à t. L'opération mise à feu de t est définie pour M ssi M valide t. Son exécution consiste alors à enlever de chaque place d'entrée p_r un nombre de marques égal à $W(p_r,t)$ et à déposer dans chaque place de sortie p_s un nombre de marques égal à $W(t,p_s)$.

II - 3. RESEAUX DE PETRI A PRIORITES (RdPP) [HAC.75]

Définition :

Un RdPP est un doublet $R_P = (R,\mathcal{O})$ tel que :

- R est un RdP (P,T,α,β), et
- $\mathcal{O}$ est une relation d'ordre partiel sur T.

Si $(t,t') \in \mathcal{O}$, $t \neq t'$, on dira que t est plus prioritaire que t'.

Opérations sur les marquages :

Une transition t est validée par un marquage M ssi chacune de ses places d'entrée contient au moins une marque. L'opération mise à feu de t est définie pour M ssi M valide t et ne valide aucune transition plus prioritaire. Son exécution s'effectue comme pour le RdP R.

Un réseau de Petri à priorité élémentaire (RdPPE) est un RdPP $(R,\mathcal{O})$ tel que $\mathcal{O}$ définit deux classes de transitions T_λ et T_0 avec :
$T_\lambda \cup T_0 = T$, $T_\lambda \cap T_0 = \emptyset$, et $(t_\lambda, t_0) \in T_\lambda \times T_0 \Leftrightarrow (t_\lambda, t_0) \in \mathcal{O}$, $t_\lambda \neq t_0$.

Ces extensions ont été apportées pour permettre de décrire, de façon directe, certains types de fonctionnements dont la modélisation à l'aide des RdP ordinaires s'avère trop compliquée ou impossible. Cependant, nous allons montrer que, sur le plan théorique, la seule sous-classe constituée par les RdP à priorité élémentaire suffit à couvrir l'ensemble des possibilités de modélisation offertes par les trois extensions.

Pour comparer la puissance de description de deux classes de réseaux, nous adoptons le critère classique qui consiste à considérer un réseau comme un automate générateur de langage. Nous comparons alors les classes de réseaux en comparant les classes de langages qu'elles permettent de générer.

III - RELATION D'EQUIVALENCE SUR LES RdP [MOA.78]

Réseau de Petri étiqueté :

L'étiquetage d'un RdP $R = (P,T,\alpha,\beta)$ est défini par la donnée d'un vocabulaire V et d'une application $f : T \to V \cup \{\lambda_V\}$ où λ_V est le mot vide du monoïde libre V^* construit sur V. Le réseau étiqueté est noté $R_E = (R,V,f)$.

Tout RdP peut être considéré comme étant étiqueté de façon triviale en prenant V = T et f = application identité.

On note par f^* l'extension naturelle de f, $f^* : T^* \to V^*$, définie de la façon suivante : si $t \in T$, $\sigma \in T^*$ et λ_T est le mot vide de T^* on a $f^*(t\sigma) = f(t)\ f^*(\sigma)$ et $f^*(\lambda_T) = \lambda_V$.

On note par $L_T(R_E,M)$ l'ensemble des séquences de simulation possibles à partir d'un marquage M de R_E.

Le langage généré par R_E à partir de M, que l'on note par $L_V(R_E,M)$, est le sous-ensemble de V^* défini par :

$$L_V(R_E,M) = \{f^*(\sigma) \mid \sigma \in L_T(R_E,M)\}$$

Relation de réalisation :

Soient $R_{E1} = (R_1,V,f_1)$ et $R_{E2} = (R_2,V,f_2)$ deux RdP étiquetés. On dira que R_{E1} réalise R_{E2} ssi pour tout marquage M_2 de R_2 il existe un marquage M_1 de R_1 tel que : $L_V(R_{E1},M_1) = L_V(R_{E2},M_2)$.

Relation d'équivalence :

On dira que deux RdP étiquetés R_{E1} et R_{E2} sont équivalents ssi chacun des deux réseaux réalise l'autre.

Proposition 1 :

Etant donné un RdPZ R_Z (considéré comme étant étiqueté de façon triviale), on peut toujours construire un RdPPE étiqueté R_E qui le réalise.

Démonstration :

Un arc inhibiteur (p,t) dans un RdPZ représente la condition : la transition t ne peut être validée par un marquage M que si le test <place p vide pour M?> est vrai. Ce test peut être transformé en <place $\overline{p}$ contient une marque?>, où $\overline{p}$ est une nouvelle place ajoutée au réseau, si l'on peut garantir la propriété de "complémentarité" suivante : pour tout marquage de départ M_0 et pour tout marquage successeur $M \in \vec{M}_0$ pour lequel l'opération MAF_t est définie, $\overline{p}$ contient une marque ssi p est vide.

Ce principe est appliqué pour la construction du RdPPE étiqueté R_E qui réalise le RdPZ R_Z. Si $R_Z = (R, I)$ avec $R = (P,T,\alpha,\beta)$, alors $R_E = ((R_e, \mathcal{O}), V_e, f_e)$ avec $R_e = (P_e, T_e, \alpha_e, \beta_e)$ construit selon les règles suivantes :

r_1) $\forall\ p \in P \Rightarrow p \in P_e$

r_2) $\forall\ t \in T \Rightarrow t \in T_e$

r_3) $\forall\ (p,t) \in \alpha \Rightarrow (p,t) \in \alpha_e$

r_4) $\forall\ (t,p) \in \beta \Rightarrow (t,p) \in \beta_e$

r_5) pour toute place $p \in P$ de laquelle est issu un arc inhibiteur $(p,t) \in I$, on crée une place $\overline{p}$ et une transition $t_{\lambda p}$ telles que :

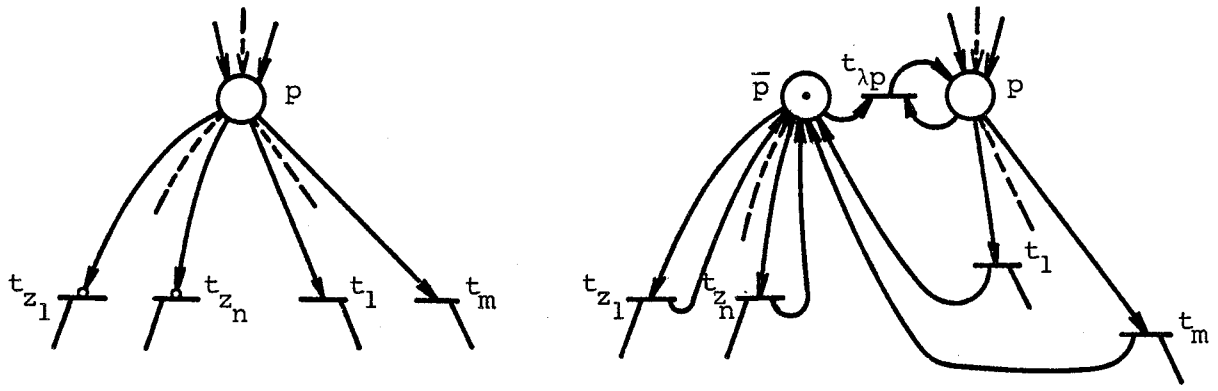
- $\overline{p} \in P_e$, $t_{\lambda p} \in T_e$, $(p,t_{\lambda p}) \in \alpha_e$, $(\overline{p},t_{\lambda p}) \in \alpha_e$, $(t_{\lambda p},p) \in \beta_e$
- $\forall\ t_i \in T$, $(p,t_i) \in \alpha \Rightarrow (t_i,\overline{p}) \in \beta_e$
- $\forall\ t_j \in T$, $(p,t_j) \in I \Rightarrow (\overline{p},t_j) \in \alpha_e$, $(t_j,\overline{p}) \in \beta_e$

r_6) $(t_i,t_j) \in \mathcal{O}$, $t_i \neq t_j \Leftrightarrow t_i \in T_e - T$ et $t_j \in T$

r_7) $V_e = T$

r_8) $\forall\ t \in T_e$, $f_e(t) = t$ si $t \in T$ et λ_V sinon.

Cette construction est illustrée par la figure 5.



(a) dans R_Z (b) dans R_E

Figure 5.

Soient M l'ensemble des marquages de R_Z et M_e l'ensemble des marquages de R_E. On définit l'application $g : M \to M_e$ telle que si $g(M) = M_e$ alors :

- $\forall\, p_i \in P$, $M_e(p_i) = M(p_i)$
- $\forall\, \bar{p}_j \in P_e - P$, $M_e(\bar{p}_j) = 0$ si $M(p_j) > 0$ et 1 sinon.

Les déductions suivantes sont triviales :

d1) $\forall\, t \in T$, si t est validée par M_0 dans R_Z alors t est validée par $M_{e_0} = g(M_0)$ dans R_E.

d2) si $MAF_t(M_i) = M_j$ dans R_Z alors $MAF_t(g(M_i)) = M'_{e_j}$ dans R_E tel qu'il existe une séquence de simulation unique (à une permutation près) $\sigma = t_{\lambda p1}\, t_{\lambda p2} \ldots t_{\lambda pr}$, $\sigma \in (T_e - T)^*$, telle que :
- $M'_{e_j} \overset{g}{\neq} M_{e_j} = g(M_j)$
- aucune transition $t_{\lambda p} \in T_e - T$ n'est validée par M_{e_j}.

d3) si R_Z est borné (resp. vivant, sauf) pour M_0 alors R_E est borné (resp. vivant, sauf) pour $M_{e_0} = g(M_0)$.

d4) si R_Z est persistant pour M_0 alors R_E est persistant pour $M_{e_0} = g(M_0)$ en tenant compte toutefois de la priorité des transitions $t_{\lambda p}$.

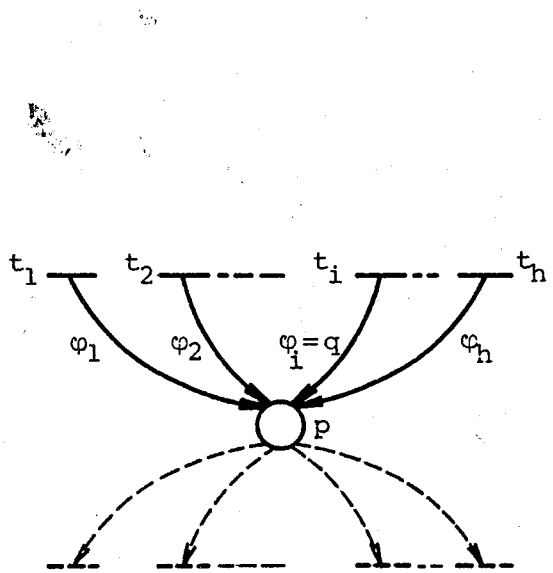
d5) $L_V (R_E, M_{e_0}) = L_V (R_Z, M_0)$.

Proposition 2 :

Etant donné un RdPG R_G (considéré comme étant étiqueté de façon triviale), on peut toujours construire un RdPPE étiqueté R_E qui le réalise.

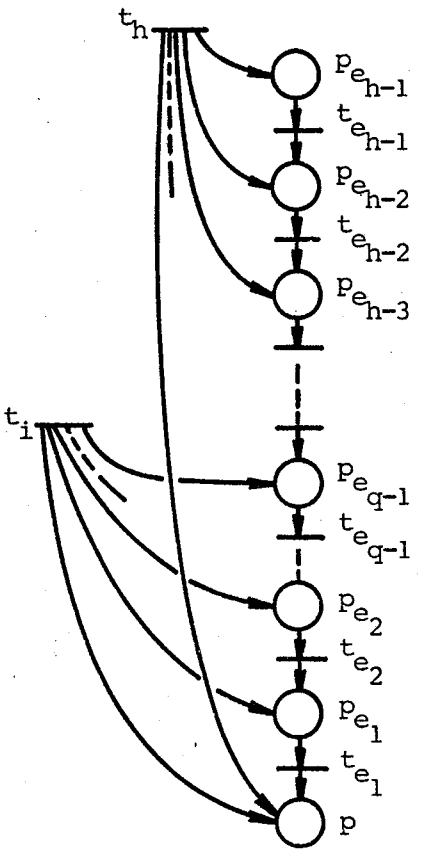
Démonstration :

Soit $R_G = (R, W)$ avec $R = (P,T,\alpha,\beta)$. On considère le réseau R_e obtenu à partir de R_G en effectuant pour chaque place p les deux transformations illustrées en figure 6 et figure 7.



$h = \max \{W_i = W(t_i,p) \mid t_i \in {}^\bullet p\}$

(a) dans R_G



(b) dans R_e

Figure 6.

La transformation en figure 6 introduit h-1 places p_{e_1}, p_{e_2}, ... $p_{e_{h-1}}$

et h-1 transitions t_{e_1}, t_{e_2}, ... $t_{e_{h-1}}$ avec $h = \max \{W_i = W(t_i,p) \mid t_i \in {}^{\bullet}p\}$. Chaque transition t_{e_j} a comme place d'entrée p_{e_j} et comme place de sortie $p_{e_{j-1}}$ pour j = 1,2,... h-1 en prenant $p_{e_0} = p$. Pour toute transition $t_i \in {}^{\bullet}p$, si $W(t_i,p) = q$ alors l'arc (t_i,p) dans R_G est remplacé par q arcs connectant t_i aux places p, p_{e_1}, p_{e_2}, $p_{e_{q-1}}$.

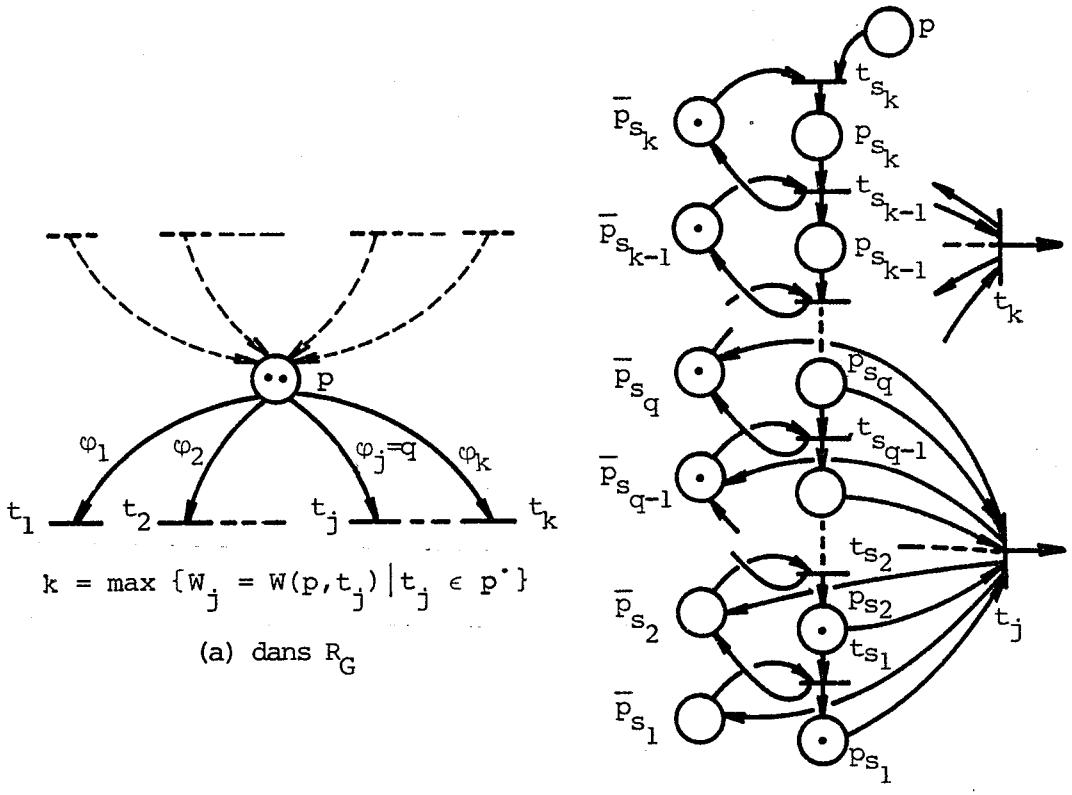


Figure 7.

La transformation en figure 7 introduit 2k places p_{s_1}, $\overline{p}_{s_1}$, p_{s_2}, $\overline{p}_{s_2}$, ..., p_{s_k}, $\overline{p}_{s_k}$ et k transitions t_{s_1}, t_{s_2}, ... t_{s_k} avec $k = \max \{W_j = W(p,t_j) \mid t_j \in p^{\bullet}\}$. Ces transitions sont connectées de façon que ${}^{\bullet}t_{s_j} = \{\overline{p}_{s_j}, p_{s_{j+1}}\}$ et $t_{s_j}^{\bullet} = \{p_{s_j}, \overline{p}_{s_{j+1}}\}$ pour j = 1,2,... k-1. La transition t_{s_k} a comme place d'entrée p et $\overline{p}_{s_k}$, et comme place de

sortie p_{s_k}. Pour toute transition $t_j \in p^\bullet$ dans R_G, si $W(p,t_j) = q$ alors l'arc (p,t_j) de R_G est remplacé par q arcs connectant les places p_{s_1}, $p_{s_2}, \ldots, p_{s_q}$ à t_j et q arcs connectant t_j aux places $\overline{p}_{s_1}$, $\overline{p}_{s_2}, \ldots, \overline{p}_{s_q}$.

Soit $R_E = ((R_e, \mathcal{O}), V_e, f_e)$ le RdPPE étiqueté défini tel que :

- $(t_i,t_j) \in \mathcal{O}$, $t_i \neq t_j \iff t_i \in T_e - T$ et $t_j \in T$
- $V_e = T$
- $\forall\, t \in T_e$, $f_e(t) = t$ si $t \in T$ et λ_V sinon.

Soient M l'ensemble des marquages de R_G et M_e l'ensemble des marquages de R_E. On définit l'application $g : M \to M_e$ telle que si $g(M) : M_e$ alors pour toute place p on a :

- si $M(p) = m$ avec $0 < m \leq k = \max \{W_j = W(p,t_j) \mid t_j \in p^\bullet\}$ alors $M_e(p_{s_1}) = M_e(p_{s_2}) \ldots = M_e(p_{s_m}) = 1$
- si $M(p) = m$ avec $m > k$ alors $M_e(p_{s_1}) = M_e(p_{s_2}) \ldots = M_e(p_{s_k}) = 1$ et $M_e(p) = m - k$
- pour toutes les autres places p_{s_i}, $M_e(p_{s_i}) = 0$
- $\forall\, j = 1,2, \ldots,k$, $M_e(\overline{p}_{s_j}) = 1$ si $M_e(p_{s_j}) = 0$ et 0 sinon.

Il est aisé de vérifier les déductions suivantes :

d1) $\forall\, t \in T$, si t est validée par M_0 dans R_G alors t est validée par $M_{e_0} = g(M_0)$ dans R_E.

d2) si $MAF_t(M_i) = M_j$ dans R_G alors $MAF_t(g(M_i)) = M'_{e_j}$ dans R_E tel qu'il existe une séquence de simulation unique (à une permutation près) $\sigma = t_{\lambda_1} t_{\lambda_2} \ldots t_{\lambda_n}$, $\sigma \in (T_e - T)^*$, telle que :
- $M'_{e_j} \overset{\sigma}{\to} M_{e_j} = g(M_j)$
- aucune transition $t_\lambda \in T_e - T$ n'est validée par M_{e_j}.

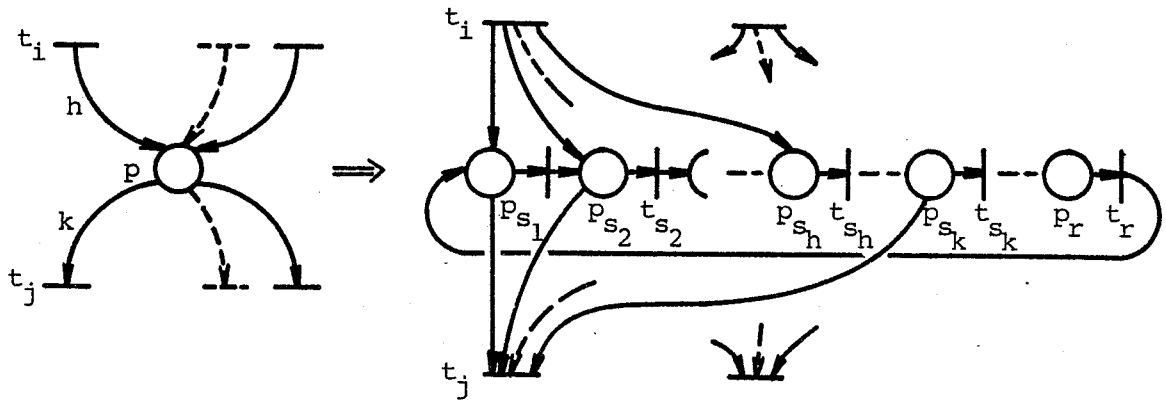
d3) si R_G est borné (resp. vivant) pour M_0 alors R_E est borné (resp. vivant) pour $M_{e_0} = g(M_0)$.

d4) si R_G est persistant pour M_0 alors R_E est persistant pour $M_{e_0} = g(M_0)$ en tenant compte toutefois de la priorité des transitions t_λ.

d5) $L_V(R_E, M_{e_0}) = L_V(R_G, M_0)$.

Remarque :

Une solution très voisine a été proposée par HACK dans [HAC.75] qui consiste à substituer à la place p un "anneau" comportant r places p_{s_1}, p_{s_2}, ... p_{s_r} et r transitions t_{s_1}, t_{s_2}, ... t_{s_r} étiquetées avec λ_V (figure 8). Tout arc (t_i,p) de poids h est remplacé par h arcs connectant t_i à p_{s_1}, p_{s_2}, ... p_{s_h}. De même, tout arc (p,t_j) de poids k est remplacé par k arcs connectant les places p_{s_1}, p_{s_2}, ... p_{s_k} à t_j, aucune priorité n'étant définie entre les transitions.



$r = \max_{i,j} \{W_i, W_j\}$ avec $W_i = W(t_i,p) \mid t_i \in {}^\bullet p$ et $W_j = W(p,t_j) \mid t_j \in p^\bullet$

Figure 8.

Le réseau obtenu réalise effectivement le réseau initial. Cependant, il n'est pas donné d'algorithme qui gère la répartition des marques dans l'anneau. Quand une opération de mise à feu d'une transition d'entrée amène une marque dans une place p_s, cette marque peut boucler indéfiniment sans quitter l'anneau. Dans ce sens, le réseau obtenu n'est pas simulable.

Proposition 3 :

Etant donné un RdPP R_p (considéré comme étant étiqueté de façon triviale), on peut toujours construire un RdPPE étiqueté R_E qui le réalise.

Démonstration :

Soit $R_p = ((R,T,\alpha,\beta), 0)$. On considère le réseau $R_e = (P_e, T_e, \alpha_e, \beta_e)$ construit à partir de R_p tel que :

r_1) $\forall\ p \in P \Rightarrow p \in P_e$

r_2) $\forall\ t \in T \Rightarrow t \in T_e$

r_3) $\forall\ (p,t) \in \alpha \Rightarrow (p,t) \in \alpha_e$

r_4) $\forall\ (t,p) \in \beta \Rightarrow (t,p) \in \beta_e$

r_5) pour chaque transition t_i on introduit trois places b_i, $\overline{b}_i$ et b_{1i} et trois transitions $t_{\lambda_{i1}}$, $t_{\lambda_{i2}}$ et $t_{\lambda_{i3}}$ connectées de la façon suivante :

- $(b_i, t_{\lambda_{i1}}) \in \alpha_e$, $(t_{\lambda_{i1}}, \overline{b}_i) \in \beta_e$, $(\overline{b}_i, t_{\lambda_{i2}}) \in \alpha_e$,

  $(b_{1i}, t_{\lambda_{i2}}) \in \alpha_e$, $(t_{\lambda_{i2}}, b_i) \in \beta_e$, $(b_{1i}, t_{\lambda_{i3}}) \in \alpha_e$,

  $(b_i, t_{\lambda_{i3}}) \in \alpha_e$, $(t_{\lambda_{i3}}, b_i) \in \beta_e$, $(t_i, b_{1i}) \in \beta_e$

- $\forall\ p \in {}^\bullet t_i \Rightarrow (p, t_{\lambda_{i1}}) \in \alpha_e$ et $(t_{\lambda i1}, p) \in \beta_e$

r_6) $\forall\ t_j \in T$, $t_j \neq t_i$ alors :

$(t_i, t_j) \in 0 \Rightarrow (b_i, t_j) \in \alpha_e$ et $(t_j, b_i) \in \beta_e$

$(t_i, t_j) \notin 0 \Rightarrow (t_j, b_{1i}) \in \beta_e$

Cette construction est illustrée par la figure 9.

Soit $R_E = ((R_e, 0_e), V_e, f_e)$ le RdPPE étiqueté défini tel que :

- $(t_i, t_j) \in 0_e$, $t_i \neq t_j \iff t_i \in T_e - T$ et $t_j \in T$

- $V_e = T$

- $\forall\ t \in T_e$, $f_e(t) = t$ si $t \in T$ et λ_V sinon.

Soient M l'ensemble des marquages de R_p et M_e l'ensemble des marquages de R_E. On définit l'application $g : M \to M_e$ telle que si $g(M) = M_e$ alors :

- $\forall\ p \in P$, $M_e(p) = M(p)$

- $\forall b_i \in P_e - P$, $M_e(b_i) = \begin{cases} 0 \text{ si la transition } t_i \text{ est validée par } M \text{ dans } R_P \\ 1 \text{ sinon} \end{cases}$
- $\forall \overline{b}_i \in P_e - P$, $M_e(\overline{b}_i) = 1$ si $M_e(b_i) = 0$ et 0 sinon
- $\forall b_{li} \in P_e - P$, $M_e(b_{li}) = 0$

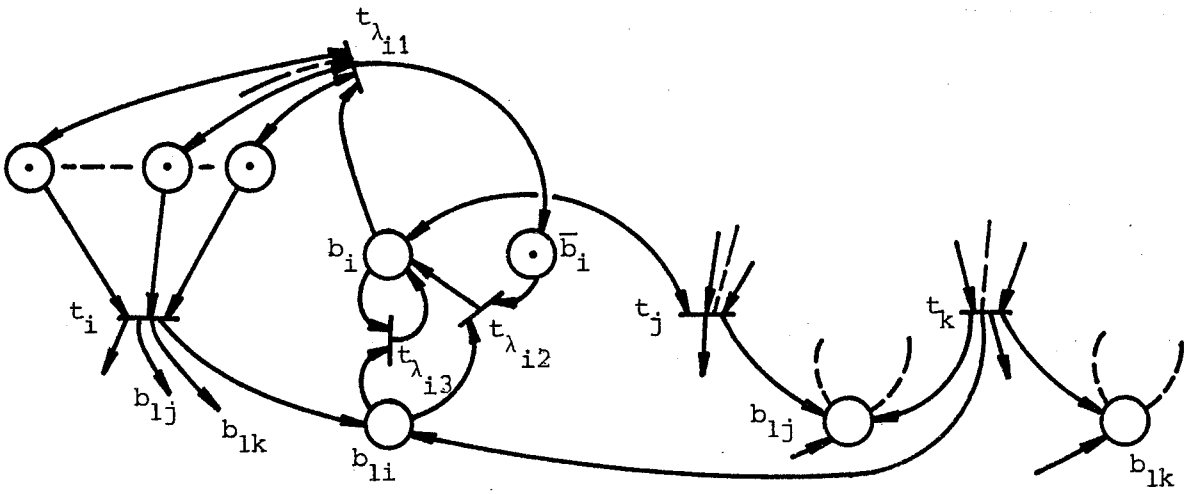
On peut vérifier facilement les résultats suivants :

d1) $\forall t \in T$, si $MAF_t(M_0)$ est définie dans R_P alors $MAF_t(g(M_0))$ est définie dans R_E.

d2) si $MAF_t(M_i) = M_j$ dans R_P alors $MAF_t(g(M_i)) = M'_{e_j}$ dans R_E tel qu'il existe une séquence de simulation unique (à une permutation près) $\sigma = t_{\lambda_1} t_{\lambda_2} \ldots t_{\lambda_n}$, $\sigma \in (T_e - T)^*$, telle que :
- $M'_{e_j} \xrightarrow{\sigma} M_{e_j} = g(M_j)$
- aucune transition $t_{\lambda_j} \in T_e - T$ n'est validée par M_{e_j}.

d3) si R_P est borné (resp. vivant) pour M_0 alors R_E est borné (resp. vivant) pour $M_{e_0} = g(M_0)$

d4) $L_V(R_E, M_{e_0}) = L_V(R_P, M_0)$.



t_i est plus prioritaire que t_j,
t_i et t_k sont non comparables.

Figure 9.

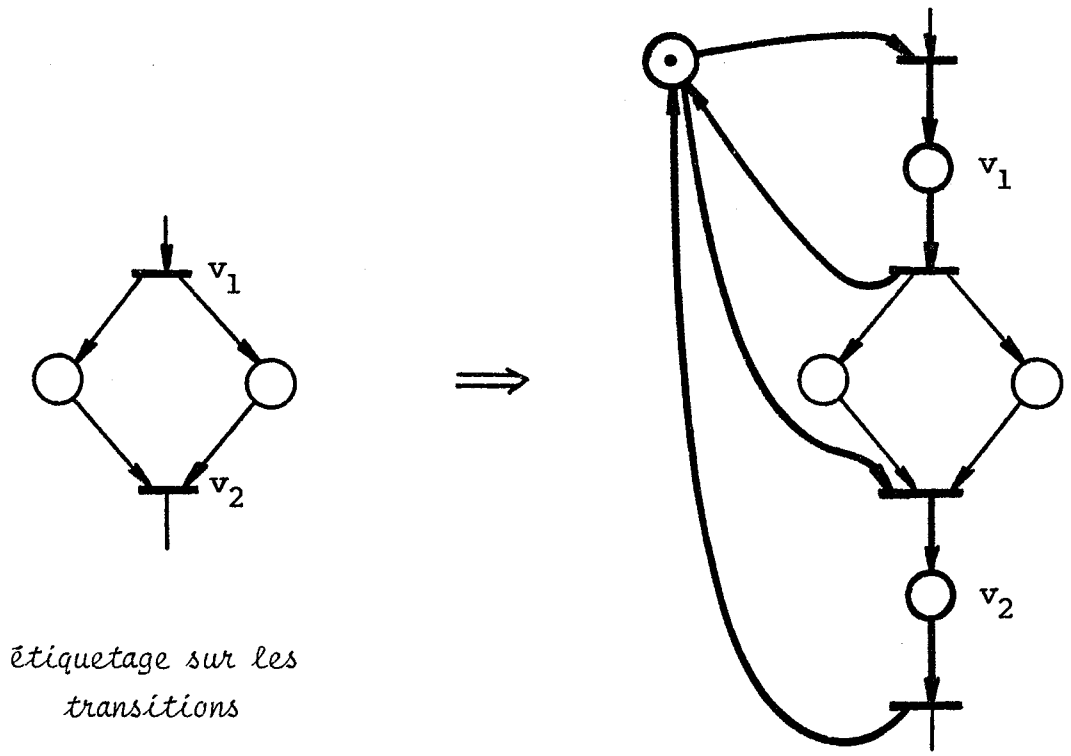
Remarques

1. Nous avons démontré les trois propositions en partant à chaque fois d'un RdP R considéré comme étant étiqueté de façon triviale. Ces démonstrations restent valables quel que soit l'étiquetage associé à R, étant donné que les transformations permettant d'obtenir R_E à partir de R ne présupposent aucune hypothèse particulière sur cet étiquetage. Si $R_0 = (R,V,f)$ est un RdP étiqueté construit sur R, l'étiquetage du réseau R_E qui réalise R_0 serait tel que $V_e = V$ et $f_e(t) = (f(t)$ si t est une transition de R et λ_V sinon) avec λ_V = élément vide de V^*.

2. Aussi, pour la commodité des démonstrations, nous avons utilisé un étiquetage sur les transitions. Les trois propositions auraient pu être énoncées en considérant un étiquetage sur les places. Il suffit de voir sur les figures 10 et 11 comment on peut passer d'un étiquetage sur les transitions à un étiquetage sur les places et inversement :

   . Quand l'étiquetage est associé aux transitions, le symbole associé à une transition t est généré chaque fois que l'opération MAF_t est effectuée. Le langage généré est un sous-ensemble de V^*.

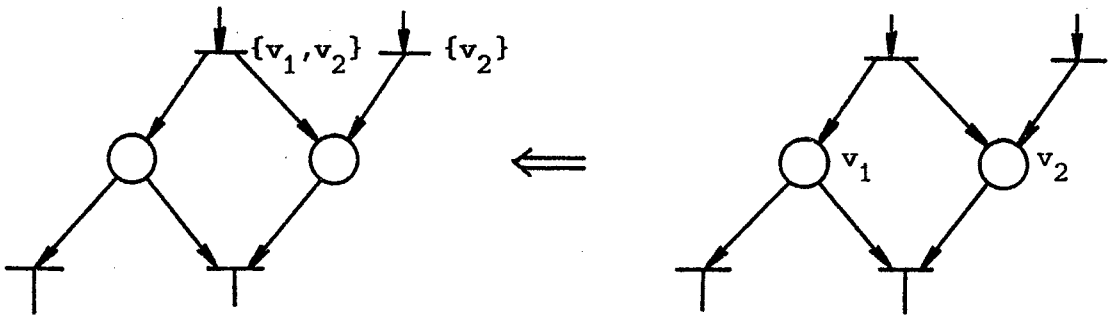
   . Quand l'étiquetage est associé aux places, chaque exécution d'une opération MAF_t génère le symbole de $P(V)$ constitué par l'ensemble des symboles associés aux places de sortie de t. Le langage généré est un sous-ensemble de $(P(V))^*$.



étiquetage sur les transitions

étiquetage sur les places

Figure 10.



étiquetage sur les transitions

étiquetage sur les places

Figure 11.

CHAPITRE V

RÉSEAUX DE PETRI NON-AUTONOMES

I - INTRODUCTION 1

II - RESEAUX DE PETRI SYNCHRONISES (RdPS) 2

. Définition 2
. Représentation 2
. Opérations sur les marquages 2
. Règles de fonctionnement d'un RdPS 5

III - PROPRIETES LIEES AUX MARQUAGES D'UN RdPS 6

. Propriété de borné 7
. Propriété de vivacité 8
. Propriété de persistance 9

IV - RESEAUX DE PETRI ETIQUETES SYNCHRONISES (RdPES) 9

. Définition 9
. Fonctionnement 10
. Relation de réalisation 10
. Relation d'équivalence 10
. Propositions 10

V - RESEAUX DE PETRI TEMPORISES SYNCHRONISES (RdPTS) 12

. Définition 12
. Représentation 12
. Opérations sur les marquages 13
. Règles de fonctionnement d'un RdPTS 14

VI - RESEAUX DE PETRI INTERPRETES (RdPI) 17

. Définition 17
. Représentation 17
. Opérations sur les marquages 17
. Simulation du fonctionnement d'un RdPI 18
. Propriétés liées aux marquages d'un RdPI 21
. Utilisation des RdPI pour la description des systèmes discrets. 21

I - INTRODUCTION

Dans les extensions successives que nous allons introduire pour définir le fonctionnement des RdP non-autonomes à partir de celui des RdP autonomes, deux notions importantes vont intervenir : la notion d'événement et la notion de temps. Ces deux notions sont nécessaires, d'une part, pour décrire le comportement d'un système vis-à-vis de son environnement externe et, d'autre part, pour pouvoir effectuer des évaluations dynamiques sur ce comportement.

Du point de vue pratique, un événement est le nom que l'on donne à un "fait". Par exemple, on peut s'intéresser à l'événement de changement de valeur d'une variable du système, ou à l'événement du passage de la valeur d'une variable d'un domaine de valeurs à un autre, ou encore à l'événement de réactualisation de la valeur d'une variable (la réactualisation n'implique pas toujours un changement de valeur de la variable),...

On dit qu'il y a occurrence d'un événement chaque fois que l'événement se produit. On se réfère généralement à un "repère de temps" pour associer à chaque occurrence une date appelée instant de l'occurrence. Le plus souvent, le repère de temps est confondu avec l'ensemble des réels $\mathbb{R}$ et l'instant d'une occurrence est confondu avec un élément de $\mathbb{R}$. On observe "rationnellement" la règle d'associer aux occurrences successives d'événements des instants croissants. La notion du temps induit ainsi sur l'ensemble des occurrences d'événements une relation de préordre total "$\leq$" et une distance "d" : Le préordre est défini par l'ordre des instants d'occurrence, et la distance est définie par la largeur des intervalles entre instants d'occurrence.

Pour un système donné, un événement peut être interne ou externe. Un événement est dit interne s'il résulte de l'activation d'un opérateur interne au système, il est dit externe sinon. Le système est dit autonome si son comportement n'est sensible à aucune occurrence d'événement externe ; autrement dit, s'il n'existe aucune relation qui lie les occurrences d'événements externes aux occurrences d'événements internes. Le système est dit non-autonome sinon.

Un système de commande en temps réel est, d'une façon générale, un système non-autonome qui réagit aux occurrences d'événements externes. L'objet de sa description par un RdP non-autonome est précisément de

décrire la relation entre les occurrences des événements externes et les occurrences des événements internes qui doivent en résulter. A ces fins, nous allons définir successivement les RdP Synchronisés (RdPS), les RdP Temporisés Synchronisés (RdPTS) et les RdP Interprétés (RdPI).

Dans un RdPS, on cherche à décrire les évolutions possibles des états du système si l'on ne tient compte que de l'ordre des occurrences des événements externes. Un état du système est représenté par un marquage du RdPS. Dans un RdPTS, on introduit, en plus, le facteur temps. Enfin, dans un RdPI, on superpose au RdPTS une interprétation qui permet de décrire une partie opérative agissant sur un ensemble de variables. On peut alors modéliser de façon complète le comportement d'un système temps réel complexe pouvant comporter, par exemple, des traitements numériques.

II - RESEAUX DE PETRI SYNCHRONISES (RdPS) [MOA.76] [MOA.78]

Définition :

Un RdPS R_S est défini par la donnée :

- d'un RdP $R = (P, T, \alpha, \beta)$,
- d'un ensemble E d'événements externes contenant un élement e appelé "événement toujours présent"
- et d'une application μ de T dans E.

Représentation :

On représente un RdPS par le RdP associé en portant sur chaque transition t_j le renseignement $<\mu(t_j)>$ précisant l'événement associé à t_j (figure 12).

Opérations sur les marquages :

- Une transition t de R_S est dite réceptive au sous-ensemble d'événements $X \subseteq E$ pour un marquage M ssi t est validée par M et $\mu(t) \in X$.

- Soit $T_{X,M} = \{t_1, t_2, \ldots t_r\}$ l'ensemble des transitions réceptives à X pour M. On appelle séquence de simulation complète par rapport à X pour le marquage M toute séquence de simulation σ_c à partir de M qui vérifie les propriétés suivantes :

a) les transitions de σ_c sont exclusivement des transitions de $T_{X,M}$

b) toute transition de $T_{X,M}$ n'apparaît qu'une fois au plus dans σ_c

c) toute séquence σ'_c obtenue en permutant les transitions de σ_c est aussi une séquence de simulation à partir de M

d) σ_c est maximale ; c'est-à-dire qu'il n'existe pas d'autres séquences plus longues qui contiennent toutes les transitions de σ_c et qui vérifient les propriétés a, b et c

- On appelle tir sur occurrence de X appliqué à M l'exécution à partir de M d'une séquence de simulation complète σ_c. On note $M \xrightarrow{X/\sigma_c} M'$. Le marquage M' est dit marquage atteint par suite du tir sur occurrence de X appliqué à M selon σ_c.

Dans le cas où aucune transition de R_S n'est réceptive à X pour M, on convient que la seule séquence de simulation complète par rapport à X pour le marquage M est la séquence de simulation vide λ_T. On note $M \xrightarrow{X/\lambda_T} M$. Le tir sur occurrence de X appliqué à M est alors dit non-effectif.

- Un marquage M de R_S est dit stable si aucune transition de R_S n'est réceptive à $\{e\}$ pour M.

- On appelle tir itéré sur occurrence de X appliqué à un marquage stable M une séquence de tirs composée d'un tir sur occurrence de X appliqué à M, suivi d'une itération de tirs sur occurrence de $\{e\}$ jusqu'à atteindre un marquage stable M'.

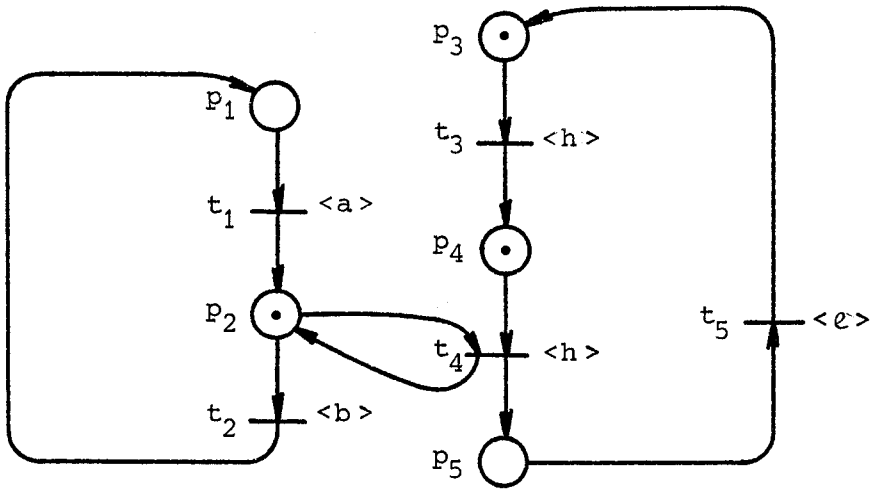
Si cette itération est finie, on note $M \xrightarrow{X//\sigma} M'$ où σ est la concaténation des séquences de simulation complètes successivement utilisées. Le marquage M' est dit marquage atteint par suite du tir itéré sur occurrence de X appliqué à M selon σ.

La notation $M \xrightarrow{X//} M'$ fait abstraction des séquences de simulation complètes successivement utilisées.

Par extension, on dira qu'un marquage stable M' est atteignable par suite d'une séquence de tirs itérés sur occurrence de ξ appliquée à un marquage stable M, où $\xi = X_1 X_2 \ldots X_q$ est une séquence de parties non vides de E, s'il existe une suite de marquages stables $M_0 = M, M_1, M_2, \ldots M_q = M'$ tels que :

$\forall i = 1,2,..,q,\ M_{i-1} \xrightarrow{X_i//} M_i$. On note $M \xrightarrow{\xi//} M'$.

- Un RdPS est dit totalement synchronisé si l'événement e n'est associé à aucune de ses transitions.



$$M = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 1 \\ 0 \end{bmatrix} \quad , \quad M'_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 2 \\ 0 \end{bmatrix} \quad , \quad M'_2 = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}$$

- $T_{\{b,\ h\},M} = \{t_2,\ t_3,\ t_4\}$

- *Séquences de simulation complètes par rapport à* {b, h} *pour le marquage* M :

  $\sigma_{c1} = t_2 t_3$ *(à une permutation près),* $M \xrightarrow{\{b,\ h\}\ /\ \sigma_{c1}} M'_1$

  $\sigma_{c2} = t_3 t_4$ *(à une permutation près),* $M \xrightarrow{\{b,\ h\}\ /\ \sigma_{c2}} M'_2$

  Le marquage M'_2 *n'est pas stable,* $M \xrightarrow{\{b,\ h\}\ //} M''_2 = M$

- $T_{\{a\},M} = \emptyset$, *le tir sur occurrence de* {a} *appliqué à* M *est non effectif.*

Figure 12. Exemple d'un RdPS

Règles de fonctionnement d'un RdPS :

Remarque :

Dans un RdP autonome, une opération MAF_t peut toujours être appliquée au marquage présent M pourvu que ce marquage valide t. Toutes les séquences de simulation sont autorisées ; autrement dit, il n'existe aucune relation particulière entre les opérations de mise à feu. Dans un RdPS, l'exécution de ces opérations est synchronisée sur les occurrences des événements externes associés aux transitions. Plus précisément, le RdPS se comporte comme une machine qui évolue en fonction d'une séquence de parties de E engendrée par l'environnement extérieur, chaque élément de la séquence modélisant l'occurrence simultanée des événements qui le composent.

Cette synchronisation crée un lien entre les opérations de mise à feu et, comme nous le verrons, modifie les propriétés liées aux marquages du réseau.

En pratique, on étudie le comportement d'un RdPS à partir d'un marquage initial M_0 stable et pour un environnement représenté par un langage $L \subseteq (P(E) - \emptyset)^*$ ayant la propriété de préfixe [PUL.79], c'est-à-dire tel que : $\forall\ \xi X \in L$, on a $\xi \in L$. Le langage L est généralement construit sur un vocabulaire $\mathcal{E}$ strictement inclu dans $P(E) - \emptyset$.

Dans la suite, nous supposerons que L et $\mathcal{E}$ vérifient toujours les propriétés suivantes :

1°) $\{e\} \in \mathcal{E}$

2°) $\forall\ X \in \mathcal{E}$, $e \in X$

3°) $\{e\} \in L$

4°) $\forall\ \xi \in L$, $\{e\}^* \xi \{e\}^* \in L$

5°) $\forall\ \xi X \in L$, $\xi \in L$.

(les propriétés 4 et 5 impliquent les propriétés 1 et 3).

- On dira que deux événements x et y sont compatibles dans L ssi il existe $X \in \mathcal{E}$ contenant x et y.

On distinguera deux environnements particuliers :

- L'environnement le moins contraignant que l'on notera par LL. Il s'agit de l'environnement constitué par l'union de tous les environnements

possibles du RdPS :

$LL = (P'(E))^*$ où $P'(E) = \{X \in P(E) \mid e \in X\}$

- L'environnement le plus strict, que l'on notera par LS, dans lequel les occurrences d'événements différents de e sont toujours disjointes :

$LS = (P''(E))^*$ où $P''(E) = \{x,e\}_{x \in E}$

Les règles de fonctionnement du RdPS sont les suivantes :

r_1) Une opération MAF_t ne peut être effectuée que dans le cadre d'un tir sur occurrence d'un sous-ensemble $X \in \mathcal{E}$ contenant $\mu(t)$.

r_2) Pour toute occurrence d'un sous-ensemble X générée par l'environnement extérieur, on effectue un (et un seul) tir itéré sur occurrence de ce sous-ensemble à partir du marquage courant (ce tir itéré peut s'avérer non-effectif).

r_3) Les occurrences de sous-ensembles d'événements sont traitées une par une dans l'ordre où elles se présentent.

On dira que le RdPS est prompt pour le marquage M_0 tenant compte de L si : $\forall$ M tel que $\exists\ \xi \in L$ et $M_0 \xrightarrow{\xi//} M$, et $\forall\ X \in \mathcal{E}$ tel que $\xi X \in L$, le tir itéré sur occurrence de X appliqué à M conduit à un marquage stable au bout d'un nombre fini de tirs sur occurrence de $\{e\}$. On dira que le RdPS est k-prompt pour M_0 si ce nombre de tirs sur occurrence de $\{e\}$ est toujours inférieur ou égal à k.

Il s'ensuit de la règle r_3 que le fonctionnement d'un RdPS R_S n'est complètement défini pour un marquage M_0 et un environnement L que si R_S est prompt pour M_0 tenant compte de L.

III - PROPRIETES LIEES AUX MARQUAGES D'UN RdPS

Soient : . $R_S = (R, E, \mu)$ un RdPS,

. L un environnement construit sur un vocabulaire $\mathcal{E} \subseteq P(E) - \emptyset$,

. M_0 un marquage stable de R_S pour lequel R_S est prompt tenant compte de L,

. p et t respectivement une place et une transition de R_S.

On note par $\overrightarrow{M_0}$ l'ensemble des marquages stables atteignables à partir de M_0 tenant compte de L :

$$\overrightarrow{M_0} = \{M \mid \exists\ \xi \in L \text{ et } M_0 \xrightarrow{\xi//} M\}$$

On note par $\tilde{M}_0$ l'ensemble des marquages M tels que : $\exists\ \xi \in L$ et M est un marquage stable ou instable atteignable lors d'une séquence de tirs itérés sur occurrence de ξ appliquée à M_0.

On note par (R_S, L) le RdPS R_S auquel on associe l'environnement L.

Propriété de borné :

- On dit que la place p est bornée pour M_0 dans (R_S, L) ssi $\exists\ k \in \mathbb{N}$ tel que : $\forall\ M \in \tilde{M}_0$, on a $M(p) \leq k$. On appelle alors borne de la place p le plus petit entier k vérifiant cette propriété.

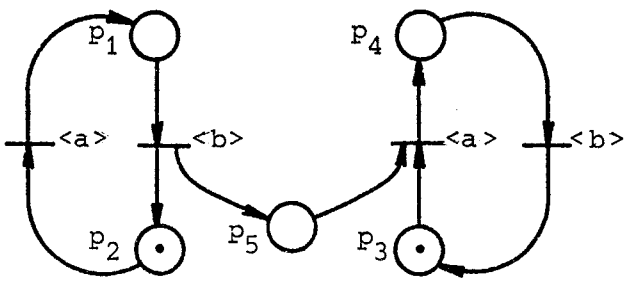
- On dit que (R_S, L) est borné pour M_0 ssi toutes les places de R_S sont bornées pour M_0.

- On dit que (R_S, L) est sauf pour M_0 ssi toutes les places de R_S ont une borne inférieure ou égale à 1 pour M_0.

Proposition 4 :

La condition que R soit borné pour un marquage M_0 (en tant que RdP autonome) est suffisante mais non nécessaire pour que (R_S, L) soit borné pour M_0.

La preuve que la condition est suffisante est triviale puisque $\tilde{M}_0 \subseteq \vec{M}_0$. Le contre-exemple donné par la figure 13 montre que cette condition n'est pas nécessaire [MOA.78].



$$M_0 = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}$$

R n'est pas borné pour M_0 alors que (R_S, L) est borné pour M_0, $\forall\ L \subseteq (P(E)-\emptyset)^*$, $E = \{a, b, e\}$

Figure 13.

Propriété de vivacité :

- On dit que la transition t est vivante pour M_0 dans (R_S, L) ssi : $\forall M \in \vec{M}_0$, il existe une séquence $\xi \in L$ et une séquence de simulation σ telles que $M \xrightarrow{\xi//\sigma} M'$ et $t \in \sigma$.

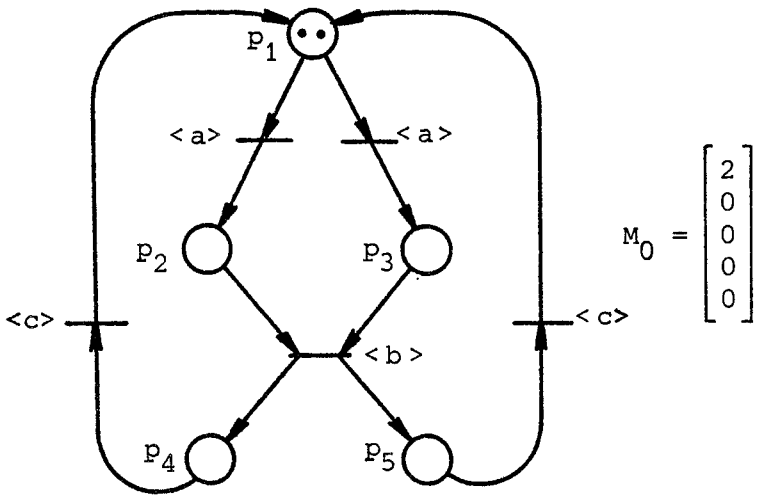
- On dit que (R_S, L) est vivant pour M_0 ssi toutes les transitions de R_S sont vivantes pour M_0.

Proposition 5 :

La condition que R soit vivant pour un marquage M_0 (en tant que RdP autonome) n'est ni nécessaire ni suffisante pour que (R_S, L) soit vivant pour M_0.

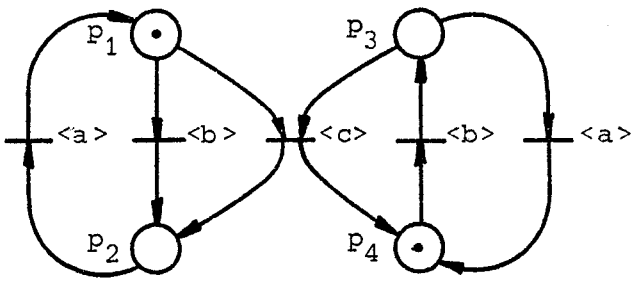
La preuve est donnée par les deux contre-exemples en figure 14 et figure 15.

Nous démontrerons dans le chapitre suivant que si un RdP ordinaire simple R est vivant pour un marquage M_0 alors : quel que soit le RdPS R_S totalement synchronisé construit à partir de R, (R_S, LL) est vivant pour M_0.



R est non vivant pour M_0 alors que (R_S, L) est vivant pour M_0 ,
$\forall L \subseteq (P(E)-\emptyset)^*$, $E = \{a, b, c, e\}$

Figure 14.



$$M_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

R est vivant pour M_0 alors que
(R_S, L) n'est pas vivant pour M_0,
$\forall\ L \subseteq (P(E)-\emptyset)^*$, $E = \{a, b, c, e\}$

Figure 15.

Propriété de persistance :

- On dit que (R_S, L) est persistant pour M_0 ssi : $\forall\ \xi \in L$ tel que $M_0 \xrightarrow{\xi//} M$, et $\forall\ X \in \mathcal{E}$ tel que $\xi X \in L$, chacun des tirs intervenant lors du tir itéré sur occurrence de X appliqué à M ne peut être effectué que selon une séquence de simulation complète σ_c unique (à une permutation près).

La condition que R soit persistant pour un marquage M_0 est suffisante mais non nécessaire pour que (R_S, L) soit persistant pour M_0. En effet, il suffit que les événements associés aux transitions de sortie de chaque place de R_S soient incompatibles dans L.

Remarque :

De la même manière que l'on vient de définir un RdPS à partir d'un RdP ordinaire autonome, on peut définir un RdPS à partir d'un RdPZ ou d'un RdPG.

IV - RESEAUX DE PETRI ETIQUETES SYNCHRONISES (RdPES)

Définition :

Un RdPES est un triplet $A = (R_S, V, f)$ où :

- R_S est un RdPS (R, E, μ)
- V est un vocabulaire

et . f est une application de T dans $P(V)$, où T est l'ensemble des transitions de R et $P(V)$ est l'ensemble des parties de V.

Fonctionnement :

Soient L un environnement construit sur un vocabulaire $\mathcal{E} \subseteq P(E) - \emptyset$, et M_0 un marquage stable de R_S pour lequel R_S est prompt tenant compte de L. On note par X un élément de $\mathcal{E}$ et par M un élément de $\vec{M}_0$.

Le fonctionnement de (A,L) à partir de M_0 suit les mêmes règles que (R_S, L) à partir de M_0. Au cours de ce fonctionnement, tout tir itéré sur occurrence de X appliqué à un marquage M selon $\sigma = t_1\ t_2\ \ldots\ t_r$ génère le symbole de sortie $w = \bigcup_{1 \le j \le r} f(t_j)$. Un tir non-effectif génère le symbole $\emptyset$ (ensemble vide).

Relation de réalisation :

Soient $A_1 = (R_{S_1}, V, f_1)$ et $A_2 = (R_{S_2}, V, f_2)$ deux RdPES avec $R_{S_1} = (R_1, E_1, \mu_1)$ et $R_{S_2} = (R_2, E_2, \mu_2)$, et soient L_1 et L_2 deux environnements respectivement de R_{S_1} et R_{S_2}.

On dira que (A_1,L_1) réalise (A_2,L_2) ssi $L_1 \supseteq L_2$ et si pour tout marquage M_2 de A_2 il existe un marquage M_1 de A_1 tel que : $\forall\ \xi \in L_2$, les ensembles des séquences possibles de symboles de sortie générées par A_1 et A_2 par suite de la séquence de tirs itérés ξ appliquée respectivement à partir de M_1 et M_2 sont les mêmes.

Relation d'équivalence :

On dira que (A_1,L_1) et (A_2,L_2) sont équivalents ssi chacun des deux doublets réalise l'autre.

Proposition 6 :

Etant donné un RdPES A_Z construit à partir d'un RdPZ totalement synchronisé comportant k transitions, on peut toujours construire un RdPES A, à partir d'un RdP ordinaire synchronisé k-prompt, tel que : pour tout environnement L de A_Z, (A,L) réalise (A_Z,L).

Démonstration :

Soit $A_Z = (R_{ZS}, V, f)$ où $R_{ZS} = (R_z, E, \mu)$ est un RdPZ totalement synchronisé. On considère le RdP étiqueté $R_{ze} = (R_z, P(V), f)$. D'après la

proposition 1 (chap. IV, § III), on peut toujours construire un RdPPE étiqueté $R_E = (R_e, P(V), f_e)$ qui réalise R_{ze}. Cette construction crée pour chaque place p de R_z, ayant un arc inhibiteur en sortie, une nouvelle place $\overline{p}$ et une nouvelle transition $t_{\lambda p}$ prioritaire étiquetée par λ(mot vide de $(P(V))^*$). Soit alors le RdPES $A = (R_s, V, f_s)$ défini tel que :

- R_s est le RdPS construit à partir de R_e en associant à chaque transition t_i qui est transition de R_{ZS} l'événement externe $\mu(t_i)$ et à chaque transition $t_{\lambda p}$ l'événement e.

- pour toute transition t de R_e, $f_s(t) = f_e(t)$ si $f_e(t) \neq \lambda$ et $f_s(t) = \emptyset$ sinon.

En fait, dans le RdPPE R_E on donne une priorité aux transitions $t_{\lambda p}$ afin d'obtenir que lors de l'exécution de toute opération MAF_t, $t \neq t_{\lambda p}$, toute place $\overline{p}$ "complémentaire" d'une place $p \in {}^\bullet t$ contient une marque ssi p n'en contient pas. On obtient ceci dans le fonctionnement synchronisé en associant aux transitions $t_{\lambda p}$ l'événement "toujours présent" e, tant que aucune transition $t \neq t_{\lambda p}$ n'est associée à l'événement e. De la façon dont est construit R_E à partir de R_{ze}, et donc A à partir de A_Z, on peut facilement vérifier les propriétés suivantes :

- Quel que soit M un marquage stable de A et quel que soit X un sous-ensemble d'événements de E, l'ensemble $T_{X,M}$ ne peut pas comporter deux transitions t_i et t_j telles que $t_i \in p^\bullet$ et $t_j \in \overline{p}^\bullet$. D'un autre côté, si k est le nombre de transitions de A_Z alors le cardinal de $T_{X,M}$ est nécessairement inférieur ou égal à k. Par conséquent, le tir sur occurrence de X appliqué à M ne peut amener dans une même place $\overline{p}$ qu'au maximum k marques. Il suffit de faire suivre ce tir d'une itération de k tirs sur occurrence de $\{e\}$ pour atteindre sûrement un marquage stable.

- Si une transition $t \neq t_{\lambda p}$ est réceptive à un sous-ensemble d'événements X pour un marquage M de A_Z, alors t est réceptive à X pour le marquage $M_1 = g(M)$ de A. Aussi, toute séquence de tirs sur occurrence de ξ appliquée à M pour A_Z telle que $M \overset{\xi//}{\Rightarrow} M'$ correspond à une séquence de tirs itérés sur occurrence de ξ appliquée à M_1 pour A telle que $M_1 \overset{\xi//}{\longrightarrow} M_1' = g(M')$, et les séquences de symboles de sortie générées sont les mêmes. (Toute transition étiquetée par λ dans R_E est étiquetée par $\emptyset$ dans A).

Le même schéma de démonstration peut être reconduit pour la propo-

sition suivante :

Proposition 7 :

Etant donné un RdPES A_G construit à partir d'un RdPG totalement synchronisé, on peut toujours construire un RdPES A, à partir d'un RdP ordinaire synchronisé k-prompt, tel que :

- k est le plus grand poids associé aux arcs de A_G
- quel que soit l'environnement L pour lequel les événements associés aux transitions de sortie d'une même place de A_G sont incompatibles, (A,L) réalise (A_G,L).

(Pour la démonstration, se référer à la proposition 2, chap. IV, § III).

V - RESEAUX DE PETRI TEMPORISES SYNCHRONISES (RdPTS) [RAM.73] [MOA.76]

Le facteur temps est introduit dans le fonctionnement d'un RdPS pour imposer une temporisation au passage des marques dans les places. Dès lors, il nous faut tenir compte également du facteur temps dans le comportement de l'environnement qui génère les occurrences des événements externes. Nous utiliserons comme "repère de temps" l'ensemble $\mathbb{R}^+$ des réels positifs ou nuls.

Un environnement pour un RdPTS sera défini par un ensemble d'événements externes E contenant l'événement toujours présent e et tel que : à tout événement $x \in E$, $x \neq e$, est associée une fonction croissante τ_x de $\mathbb{N} - \{0\} \rightarrow \mathbb{R}^+ - \{0\}$; $\tau_x(i)$ étant l'instant d'occurrence de la i-ème occurrence de x à partir de l'instant initial "0". L'ensemble des événements survenant à l'instant τ est égal à :

$$\{x \in E - e \mid \exists\, i \in \mathbb{N} - \{0\} \text{ et } \tau_x(i) = \tau\} \cup \{e\}$$

Définition :

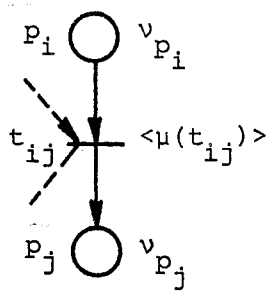
Un RdPTS est défini par la donnée :

- d'un RdPS $R_s = (R,E,\mu)$
- et d'un ensemble $\nu = \{\nu_p,\ p \in P\}$ d'applications croissantes de $\mathbb{R}^+$ dans $\mathbb{R}^+$, où P est l'ensemble des places de R.

Représentation :

On représente un RdPTS par le RdPS associé en spécifiant à côté de

chaque place p l'application ν_p.



Représentation d'un RdPTS

Opérations sur les marquages :

Une marque dans un RdPTS a deux états possibles : disponible et indisponible. Ainsi, à tout instant τ, le marquage M du réseau est la somme de deux marquages M_d et M_i où M_d est le marquage constitué des marques disponibles de M et M_i est le marquage constitué des marques indisponibles de M. (Sur la représentation graphique, on peut indiquer l'état disponible (resp. indisponible) d'une marque en la représentant par le symbole "*" (resp. "+")).

Une transition t est validée par M dans le RdPTS ssi elle est validée par M_d dans le RdPS associé. Une opération MAF_t appliquée à M à l'instant τ enlève aux places d'entrée de t des marques disponibles et dépose aux places de sortie p des marques disponibles si $\nu_p(\tau) = \tau$ et des marques indisponibles si $\nu_p(\tau) > \tau$. Ces marques indisponibles restent dans cet état durant l'intervalle $]\tau, \nu_p(\tau)[$ puis redeviennent disponibles.

On dira que $\sigma = t_1\ t_2 \ldots t_n$ est une séquence de simulation instantanée à partir de M à l'instant τ si la suite des opérations MAF_{t_1}, MAF_{t_2}, ..., MAF_{t_n} peut être appliquée à partir de M à l'instant τ.

Les opérations de tir sur occurrence d'événements à un instant τ et de tir itéré sur occurrence d'événements à un instant τ s'effectuent de la même manière que dans un RdPS en utilisant pour chaque tir une séquence de simulation complète instantanée à l'instant τ. (On peut aisément vérifier que toute séquence σ'_c obtenue en permutant les transitions d'une séquence de simulation complète σ_c instantanée à

l'instant τ est aussi une séquence de simulation complète instantanée à l'instant τ).

Règles de fonctionnement d'un RdPTS :

r_1) A l'instant initial, le marquage stable M_0 associé au réseau ne peut comporter que des marques disponibles.

r_2) Comme pour un RdPS, une opération MAF_t ne peut être effectuée que dans le cadre d'un tir sur occurrence d'un ensemble d'événements X contenant $\mu(t)$.

Toute occurrence simultanée d'événements externes déclenche instantanément l'exécution d'un (et un seul) tir itéré sur occurrence de cet ensemble d'événements. (Ce tir itéré peut s'avérer non-effectif).

r_3) Chaque fois que le marquage du réseau devient instable par suite du passage d'une (ou de plusieurs) marque(s) de l'état indisponible à l'état disponible, une séquence de tirs sur occurrence de $\{e\}$ doit être effectuée jusqu'à stabilisation et ce préalablement à toute prise en compte de nouvelles occurrences d'événements externes.

r_4) D'une façon générale, une occurrence d'événements externes ne peut être prise en compte que si le marquage du réseau est stable. Il s'ensuit que le fonctionnement d'un RdPTS n'est complètement défini que pour les marquages initiaux pour lesquels il est prompt.

Remarques :

1. Le fonctionnement d'un RdPTS pour lequel les applications ν_p sont des identités sur $\mathbb{R}^+$ est le même que celui du RdPS à partir duquel il est défini (les applications τ_x n'interviennent dans l'environnement que pour ordonner les occurrences des événements externes).

2. On peut définir un Réseau de PETRI Temporisé (RdPT) comme étant un RdPTS tel que l'ensemble E se réduit à l'événement toujours présent e.

Il faut souligner que d'autres définitions des RdPT ont été proposées dans [RAM.73] [SIF.77] et [SIF.79] qui n'adoptent pas explicitement la notion de tir sur occurrence d'événements et, par suite, admettent un ensemble de comportements plus large que celui autorisé par la définition d'un RdPT à partir d'un RdPTS.

A titre d'exemple, considérons le cas de la situation représentée

par la figure 16 et supposons que, à l'instant τ, deux marques deviennent simultanément disponibles dans la place p_1. Dans le fonctionnement d'un RdPT tel qu'on vient de le définir à partir de celui d'un RdPTS, un tir sur occurrence de $\{e\}$ doit être effectué conduisant à l'exécution des opérations MAF_{t_1} et MAF_{t_2}. Selon les définitions données dans [RAM.73] et [SIF.77], rien n'empêche que l'opération MAF_{t_1} soit appliquée deux fois successives sans que l'opération MAF_{t_2} ne soit effectuée.

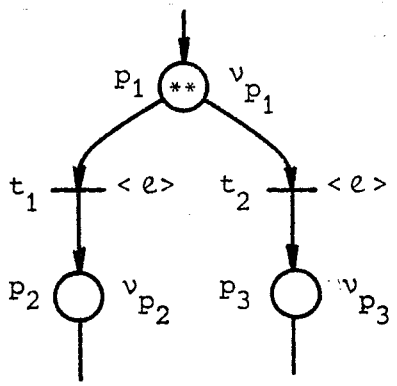


Figure 16.

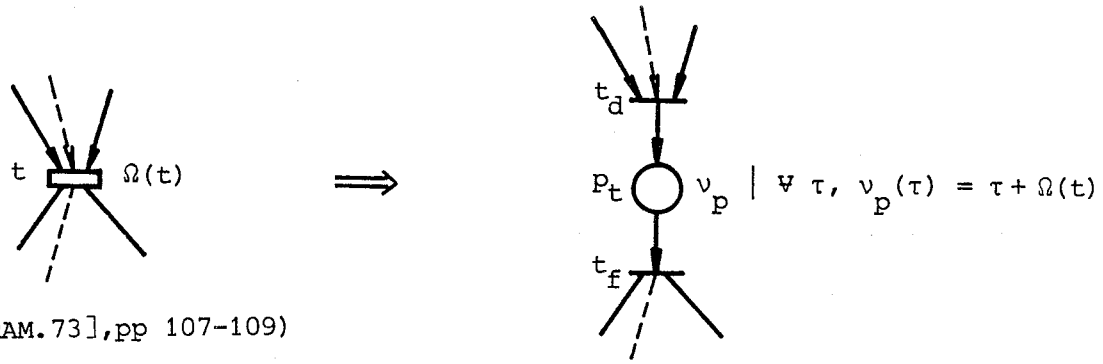
3. La définition des RdPT dans [RAM.73] diffère encore de celle proposée ici par le fait que les temps d'indisponibilité des marques sont associés aux transitions plutôt qu'aux places et sont constants.

Les figures 17 et 18 montrent que l'on peut toujours transformer une représentation où les temps d'indisponibilité sont associés aux transitions à une représentation où les temps d'indisponibilité sont associés aux places et inversement :

- Dans la transformation illustrée par la figure 17, toute transition t est éclatée en deux transitions t_d et t_f entre lesquelles vient s'intercaler une place p_t. Toutes les places d'entrée de t deviennent des places d'entrée de t_d, et toutes les places de sortie de t deviennent des places de sortie de t_f. La temporisation $\Omega(t)$ est associée à la place p_t.

- Dans la transformation inverse illustrée par la figure 18, toute place p est éclatée en deux places p_e et p_s entre lesquelles vient s'intercaler une transition t_p. Toutes les transitions d'entrée de p

deviennent des transitions d'entrée de p_e, et toutes les transitions de sortie de p deviennent des transitions de sortie de p_s. La temporisation associée à la place p devient une temporisation associée à la transition t_p ; toutes les transitions du réseau de départ gardent une temporisation nulle. A l'instant initial, les marques qui devraient être déposées dans p sont déposées dans p_s.



(cf. [RAM.73],pp 107-109)

Figure 17.

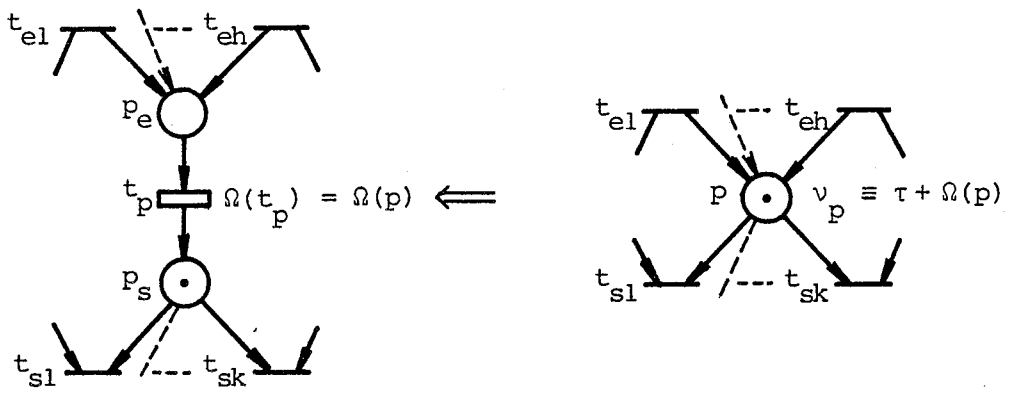


Figure 18.

Des constatations analogues à celles qui ont été faites concernant les propriétés liées aux marquages d'un RdPS (cf. § II) peuvent être faites pour le cas des RdPT [GHO.77].

VI - RESEAUX DE PETRI INTERPRETES (RdPI) [MOA.76]

Définition :

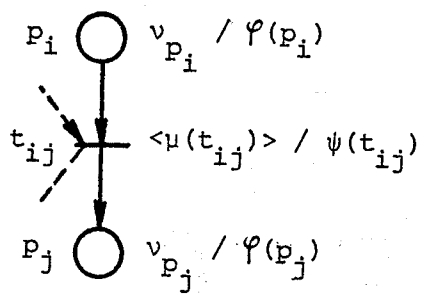
Un RdPI est défini par la donnée :

- d'un sous-système opératif (V,OP,C) tel que :
  - $V = \{v_1, v_2, \ldots, v_m\}$ est un ensemble fini de variables prenant leurs valeurs respectivement dans les domaines $D_1, D_2, \ldots, D_m$;
  - $OP = \{op_1, op_2, \ldots op_n\}$ est un ensemble fini d'opérateurs définis comme des applications internes de $D_1 \times D_2 \times \ldots \times D_m$;
  - $C = \{c_1, c_2, \ldots c_r\}$ est un ensemble de conditions (prédicats) sur les variables de V ;
- d'un RdPTS (R, E, μ, ν) ;
- d'une application $\varphi : P \rightarrow OP$, P étant l'ensemble des places de R ;

et - d'une application $\psi : T \rightarrow C$, T étant l'ensemble des transitions de R.

Représentation :

On représente un RdPI par le RdPTS associé en spécifiant à côté de chaque place p_i l'opérateur $\varphi(p_i)$ et à côté de chaque transition t_j la condition $\psi(t_j)$. (On ne s'intéresse pas à associer une représentation graphique au sous-système opératif).



Représentation d'un RdPI

Opérations sur les marquages :

Les opérations sur les marquages d'un RdPI sont celles définies pour les RdPTS avec les deux particularités suivantes :

- Une transition t n'est considérée comme étant réceptive à un ensemble d'événements X à l'instant τ que si : 1°) t est validée par le marquage du RdPTS à cet instant ; 2°) le prédicat $\psi(t)$ qui lui est associé est vérifié par les valeurs présentes des variables du sous-système opératif et 3°) $\mu(t) \in X$.

- Chaque fois qu'une marque dans une place p passe de l'état indisponible à l'état disponible, on effectue la transformation définie par l'opérateur $\varphi(p)$. Si plusieurs marques dans le réseau passent simultanément de l'état indisponible à l'état disponible, les transformations qu'elles impliquent sont effectuées dans un ordre quelconque.

Simulation du fonctionnement d'un RdPI :

Les règles de fonctionnement d'un RdPI, hormis les deux particularités que l'on vient de souligner, sont les mêmes que celles du RdPTS à partir duquel il est défini. Nous allons préciser ces règles en décrivant un algorithme de simulation du fonctionnement d'un RdPI. Cet algorithme comporte deux phases :

. Une phase A d'initialisation exécutée une fois au début de la simulation ;

. Une phase B (bouclée sur elle même) qui traite les échéances associées à un même instant.

A - Phase d'initialisation :

L'initialisation de la simulation à l'instant "0" doit préciser le marquage M_0 associé au réseau et, éventuellement, les valeurs des variables du sous-système opératif. D'une façon générale, l'initialisation des variables du sous-système opératif peut faire partie des opérations associées à certaines places du réseau.

Un échéancier de simulation est établi dans lequel sont notés, dans l'ordre, les instants associés aux occurrences des événements externes.

La simulation se poursuit en B en considérant le premier instant noté dans l'échéancier.

B - Phase de traitement des échéances associées à un même instant :

D'une façon générale au cours de la simulation, on peut trouver, associées à un même instant τ de l'échéancier, l'échéance du passage d'une (ou de plusieurs) marque(s) de l'état indisponible à l'état disponible et/ou l'échéance d'une occurrence simultanée d'événements externes.

On effectue, dans l'ordre, les traitements suivants :

1. Pour chaque marque qui doit passer de l'état indisponible à

l'état disponible dans une place p, on effectue sur les valeurs courantes des variables du sous-système opératif la transformation définie par l'opérateur $\varphi(p)$. Ensuite, on fait passer la marque à l'état disponible.

2. Ayant effectué le passage à l'état disponible de toutes les marques concernées, on réévalue, à partir des nouvelles valeurs des variables du sous-système opératif, l'ensemble des prédicats associés aux transitions. Si le nouveau marquage du réseau n'est pas stable, on effectue une itération de tirs sur occurrence de $\{e\}$ à l'instant τ jusqu'à stabilisation. Au cours de cette itération de tirs, chaque fois qu'une marque indisponible est déposée dans une place p, l'échéance $\nu_p(\tau)$ du passage à l'état disponible est notée dans l'échéancier.

3. Ce n'est qu'alors que l'on considère l'ensemble X des événements externes ayant une occurrence à l'instant τ. Un événement externe peut correspondre à un changement de valeur d'une variable v_i du sous-système opératif, ce changement étant opéré directement par l'environnement extérieur (voir remarque 2 ci-dessous). Si de tels événements apparaîssent dans X, on commence par effectuer les changements de valeur des variables respectives du sous-système opératif puis on réévalue en conséquence l'ensemble des prédicats associés aux transitions.

4. Enfin, on effectue, à partir du marquage courant M, un tir itéré sur occurrence de X à l'instant τ. Au cours de ce tir, chaque fois qu'une marque indisponible est déposée dans une place p, l'échéance $\nu_p(\tau)$ du passage à l'état disponible est notée dans l'échéancier.

La simulation reprend de nouveau en B en considérant le prochain instant de l'échéancier.

Remarques :

1. Il est évident que le fonctionnement d'un RdPI n'est simulable que pour les marquages initiaux M_0 pour lesquels il est prompt.

2. Les occurrences des événements de E traduisent les interactions du fonctionnement du RdPI avec son environnement externe. On peut considérer les événements de E comme étant tout à fait indépendants des variables du sous-système opératif. Mais on peut accepter aussi que l'environnement externe puisse agir directement sur certaines de ces variables (interactions par l'intermédiaire des variables d'entrée/sortie faisant

partie du sous-système opératif) et considérer comme événement externe le fait qu'une telle action soit effectuée.

Exemple : Considérons la situation représentée par la figure 19 et supposons que, à l'instant initial, les variables logiques x et y valent respectivement 0 et 1 et le marquage du réseau est tel que chacune des places p_5 et p_6 contient une marque disponible. Supposons aussi que les variables x et y sont des variables d'entrée qui peuvent être manipulées par l'environnement externe. On peut noter par $\uparrow x$(resp. $\downarrow x$) l'événement du passage de la variable x de la valeur 0 à la valeur 1 (resp. de la valeur 1 à la valeur 0).

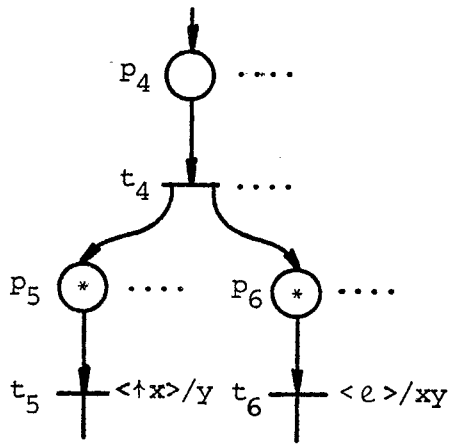


Figure 19.

Dès l'occurrence de l'événement $\uparrow x$, le prédicat xy associé à la transition t_6 devient vrai et le tir sur occurrence de $\{\uparrow x, e\}$ doit conduire à l'exécution des opérations MAF_{t_5} et MAF_{t_6}

La différence de fonctionnement entre les transitions t_5 et t_6 est la suivante : L'opération MAF_{t_5} n'est effectuée qu'à l'instant d'une occurrence de l'événement $\uparrow x$, sous la condition que, à cet instant, la transition t_5 se trouve déjà validée et le prédicat y vérifié. Par contre, l'opération MAF_{t_6} peut être effectuée soit à l'instant d'une occurrence de l'événement $\uparrow x$ si à cet instant la transition t_6 est validée et la variable y vaut 1 ; soit à l'instant d'une occurrence de l'événement $\uparrow y$ si à cet instant la transition t_6 est validée et la variable x vaut 1 ; soit enfin lors d'un tir sur occurrence de $\{e\}$ si la

transition t_6 vient à être validée pendant que le prédicat xy est vrai.

Propriétés liées aux marquages d'un RdPI :

Soient R_I un RdPI construit sur le RdPTS (R,E,μ,ν), M_0 un marquage initial stable, et L un environnement défini par les applications τ_x, $x \in E - e$.

- De la même manière que nous l'avons fait pour un RdPS, on peut définir pour le doublet (R_I,L) les propriétés de borné, de vivacité et de persistance pour le marquage M_0.

- On dira que (R_I,L) est <u>déterminé pour le marquage M_0</u> ssi : à tout instant τ où deux places p_i et p_j contiennent chacune une ou plusieurs marques indisponibles, l'ensemble des variables susceptibles d'être modifiées par $\varphi(p_i)$ et l'ensemble des variables susceptibles d'être modifiées par $\varphi(p_j)$ sont disjoints.

- On dira que (R_I,L) est <u>déterministe pour le marquage M_0</u> ssi : l'exécution de chaque phase B dans la simulation de (R_I,L) a un résultat unique en ce qui concerne le marquage atteint et l'état des variables du sous-système opératif.

Le fait que (R_I,L) soit persistant et déterminé pour le marquage M_0 est suffisant mais non nécessaire pour que (R_I,L) soit déterministe pour M_0.

Utilisation des RdPI pour la description des systèmes discrets :

On peut toujours décrire la partie opérative d'un système discret à l'aide d'un sous-système opératif {V,OP,C} tel que :

V = ensemble des variables représentant l'état de la partie opérative du système ;

OP = ensemble des opérateurs sur ces variables auquel on adjoint l'opérateur "identité" ;

et C = ensemble des tests effectués sur les variables V, auquel on adjoint le test "toujours vrai".

Le système global peut être décrit par un RdPI construit à partir de ce sous-système opératif, d'un RdPTS (R,E,μ,ν) et des applications $\varphi : P \to OP$ et $\psi : T \to C$ définies de la façon suivante :

. Le RdP R décrit le schéma de contrôle de l'algorithme de fonctionnement du système. Les applications φ et ψ définissent respecti-

vement les transformations associées aux places et les prédicats associés aux transitions.

. Les applications μ et ν traduisent le comportement dynamique que doit avoir le système dans ses interactions avec l'environnement extérieur.

Nous ne nous attarderons pas plus longtemps sur le principe de cette utilisation qui est, à peu de chose près, le même que celui que nous avons décrit pour le grafcet dans le chapitre II. Soulignons seulement que le RdPI est défini ici pour servir d'outil de conception alors que le grafcet a été défini pour servir d'outil de spécification de cahiers des charges. Nous étudierons de façon détaillée, dans le chapitre VII, les différences qui distinguent ces deux modèles, et nous situerons leurs intérêts respectifs vis-à-vis du problème global de la spécification et de la conception d'un système de commande en temps réel.

- § -

CHAPITRE VI

ANALYSE ET VÉRIFICATION DES SYSTÈMES DÉCRITS PAR DES RdP

I - PROPRIETES DE BON FONCTIONNEMENT DES SYSTEMES A EVOLUTIONS PARALLELES

. *Propriété de borné* 1
. *Propriétés de vivacité* 1
. *Propriétés relatives au partage de ressources* 1
. *Propriétés d'interdépendance entre événements internes* 2
. *Propriétés dynamiques* 3

II - METHODES D'ANALYSE ET DE VERIFICATION DES SYSTEMES DECRITS PAR DES RdP 4

II - 1. METHODES D'ANALYSE STATIQUE 5
II - 1.1. Méthodes par énumération des marquages 5
II - 1.2. Méthodes d'analyse structurelle 8
II - 1.3. Méthodes d'analyse par transformation 12

II - 2. VERIFICATION PAR SIMULATION 13

III - IMPLANTATION DE REDONDANCE 14

I - PROPRIETES DE BON FONCTIONNEMENT D'UN SYSTEME A EVOLUTIONS PARALLELES

Un des principaux intérêts de l'utilisation des RdP pour la description d'un système à évolutions parallèles est de permettre la vérification d'un certain nombre de propriétés nécessaires au bon fonctionnement de celui-ci. Ces propriétés sont, dans une large mesure, indépendantes des fonctions particulières que le système réalise, de même qu'elles sont indépendantes du modèle utilisé pour les décrire.

Nous allons rappeler quelques unes de ces propriétés et montrer comment on peut les exprimer à l'aide des RdP et des notions et concepts qu'ils introduisent.

Nous parlerons d'état d'un système et d'événement interne à un système. Les états d'un système décrit par un RdP R muni d'un marquage initial M_0 correspondent aux marquages atteignables à partir de M_0. Un événement interne au système correspond à un changement d'état résultant de l'exécution d'une opération de mise à feu de transition.

a) *Propriété de borné* : Cette propriété exprime le fait que les variables du système prennent leurs valeurs dans des domaines finis. La vérification de cette propriété en ce qui concerne les variables d'état représentées par le nombre des marques contenues dans chaque place du réseau, se ramène à vérifier que le RdP R est borné pour le marquage M_0 (chap. IV, § I - 3.3.).

b) *Propriétés de vivacité* : Ces propriétés expriment le fait qu'au cours du fonctionnement du système, tout événement interne peut se reproduire. Ces propriétés ont été étudiées sur les RdP en considérant différents types de vivacité [HAC.72] [KEL.74] [BYR.75] [LAU.T.75] [LIE.76] et [PET.77], on peut en recenser au moins sept.

Le type de vivacité à vérifier dépend évidemment de considérations relatives à ce qui est défini comme un "bon" fonctionnement du système étudié. En pratique, le type de vivacité auquel on s'intéresse le plus fréquemment est celui de la vivacité "forte" (chap. IV, § I - 3.3.) qui implique que tout événement interne est susceptible de se reproduire à partir de tout état successeur de l'état initial.

c) *Propriétés relatives au partage des ressources* : La notion de *conflit* est couramment utilisée pour exprimer le fait qu'une ressource partielle-

ment partageable peut être sollicitée par deux ou plusieurs utilisateurs simultanément. La mise en évidence de ces situations et leur résolution constituent l'un des problèmes les plus cruciaux dans l'étude et l'analyse des systèmes à évolutions parallèles.

Cette tâche est considérablement facilitée quand ces systèmes sont décrits par des RdP. En effet, la topologie même du RdP décrivant le système permet de se rendre compte des conflits possibles : un conflit est représenté par une place ayant deux ou plusieurs transitions de sortie. Démontrer la possibilité d'un conflit effectif (situation de non-persistance ; chap. IV, § I - 3.3.) à partir d'un marquage initial M_0 revient à prouver qu'il existe un marquage M successeur de M_0 qui valide deux ou plusieurs de ces transitions et tel que les mises à feu successives de ces transitions à partir de M ne soit pas possible dans n'importe quel ordre.

D'une façon générale, les problèmes de partage de ressources et d'exclusions mutuelles se traduisent par des contraintes de non-activation simultanée d'actions du système. La vérification de ces contraintes sur le RdP se ramène à prouver que les places associées aux actions en question (ces places pouvant se trouver réparties de façon quelconque dans le réseau) ne sont jamais marquées simultanément par aucun marquage successeur de M_0. Dans certains cas, la vérification de la propriété de "déterminé" pour un RdPI se ramène à la vérification d'une propriété de ce type sur le RdP associé.

d) Propriétés d'interdépendance entre événements internes : la vivacité d'un système garantit seulement que tous les événements internes peuvent se reproduire à partir de tout marquage successeur du marquage initial. Néanmoins, un système peut être vivant et admettre une séquence infinie de marquages pour lesquels une de ses transitions n'est jamais validée. Autrement dit, le fait que le système soit vivant n'oblige pas tout événement interne à avoir une occurrence certaine au bout d'un nombre fini de pas dans toute séquence de simulation effectuée à partir d'un marquage successeur du marquage initial. De telles situations traduisent ce qu'on a l'habitude d'appeler par situations de *coalition* ou de *famine* dans les systèmes [COU.77] [DIJ.71] [GHO.77].

Deux concepts intéressants ont été étudiés dans [LAU.74] [PET.75] et

[SIF.79] pour l'analyse de propriétés de ce type :

. L'avance d'un événement interne t_i par rapport à un autre événement interne t_j pour un marquage initial M_0, notée $AV(t_i,t_j;M_0)$, est définie de la façon suivante : si l'on note par $\eta(t_i,\sigma)$ le nombre d'occurrences de l'événement t_i dans la séquence σ, alors $AV(t_i,t_j;M_0)$ est égale au maximum (éventuellement infini) de la différence $\eta(t_i,\sigma) - \eta(t_j,\sigma)$ pour toute séquence de simulation σ à partir de M_0, y compris la séquence vide λ_T.

. La distance synchronique de t_i et t_j pour un marquage initial M_0, notée $DS(t_i,t_j;M_0)$, est définie comme la somme des avances respectives des deux événements, l'un par rapport à l'autre :
$DS(t_i,t_j;M_0) = AV(t_i,t_j;M_0) + AV(t_j,t_i;M_0)$.

Remarque :

Si l'on rajoute au réseau une place fictive p_{ji} ayant t_j comme transition d'entrée et t_i comme transition de sortie, la valeur $AV(t_i,t_j;M_0)$ correspond au nombre minimum de marques qu'il faut déposer initialement dans p_{ji} pour que le comportement du réseau pour le marquage M_0 ne soit pas modifié par l'adjonction de p_{ji}. La valeur $DS(t_i,t_j;M_0)$ est la somme des marques qu'il faut déposer initialement dans les places p_{ji} et p_{ij}.

L'évaluation de ces deux grandeurs [SIF.79] permet d'étudier l'interdépendance entre les événements internes du système. Le fait, par exemple, que $AV(t_i,t_j;M_0) > 0$ et $AV(t_j,t_i;M_0) = 0$ signifie que l'événement t_i "précède" toujours l'événement t_j pour toute évolution à partir de M_0. De même, le fait que $AV(t_i,t_j;M_0)$ soit infinie signifie que l'événement t_i est indépendant de l'événement t_j.

On peut vérifier qu'une condition suffisante pour qu'un système soit sans coalition est qu'il soit vivant pour le marquage initial M_0 et que, pour tout couple d'événements (t_i,t_j), $DS(t_i,t_j;M_0)$ est finie.

e) *Propriétés dynamiques* : Ce sont les propriétés qui déterminent les limites d'utilisation d'un système (temps de réponse, débits ...) à partir des caractéristiques dynamiques de ses composantes (durée des actions, densités de probabilité des transitions ...). Des propriétés de ce type ont été étudiées à l'aide des RdPT. Sur ce modèle, les temps de réponse correspondent à des retards de propagation des marques et les

débits à des fréquences de mises à feu de transitions [RAM.73] [SIF.77] [FLO.78] [SIF.79].

II - METHODES D'ANALYSE ET DE VERIFICATION DES SYSTEMES DECRITS PAR DES RdP

Il s'agit donc de vérifier des propriétés de bon fonctionnement d'un système en étudiant des propriétés du RdP qui le décrit. Un fonctionnement du système se traduit par une séquence de mises à feu de transitions dans le RdP. Le nombre des différents fonctionnements possibles peut être assez grand, voire infini.

L'analyse statique du RdP consiste à en extraire un ensemble d'informations qui sont significatives pour la vérification des propriétés recherchées. Dans certains cas, ces informations suffisent pour prouver formellement que ces propriétés sont nécessairement vérifiées par un fonctionnement donné (ou par tout fonctionnement possible) du RdP. Dans d'autres cas, ces informations sont utilisées pour enrichir les moyens d'une vérification par simulation qui s'avère la seule possible. Une vérification s'appuyant entièrement sur des résultats d'analyse statique a l'avantage de conduire à des démonstrations des propriétés, indépendamment de la longueur des fonctionnements considérés. La simulation, par contre, ne permet en général qu'une vérification partielle relative aux séquences de fonctionnement pour lesquelles le RdP est testé.

Les méthodes d'analyse statique existantes ont surtout été étudiées pour le cas des RdP autonomes pour lesquels les évolutions des marquages ne sont soumises à aucune contrainte particulière. Les extensions introduites dans les RdP non-autonomes (synchronisation, temporisation et interprétation) induisent des contraintes sur les évolutions possibles des marquages et modifient, par conséquent, les hypothèses d'application de ces méthodes. Nous avons déjà montré par des exemples (chap. V, § III) que la condition qu'un RdP R soit vivant pour un marquage M_0 n'est ni nécessaire ni suffisante pour qu'un RdPS (et a fortiori un RdPI) construit à partir de R soit vivant pour M_0. De même, nous avons montré que la condition que R soit borné pour M_0 n'est pas nécessaire pour qu'un RdPS construit à partir de R le soit.

Nous allons commencer par rappeler les principes de ces méthodes et montrer sous quelles conditions elles peuvent être utilisées pour le cas des RdP non-autonomes.

II - 1. METHODES D'ANALYSE STATIQUE

II - 1.1. Méthodes par "énumération" des marquages [KAR.69] [HAC.75]

Ces méthodes s'appliquent aux RdP ordinaires et aux RdP généralisés. Le principe, dû à KARP et MILLER [KAR.69], consiste à construire l'arbre des marquages du réseau : La racine de l'arbre correspond au marquage initial M_0. Pour chaque transition t_i validée par M_0, on crée un nouveau noeud représentant le marquage $M_i = MAF_{t_i}(M_0)$ et on joint M_0 au marquage M_i par un arc portant l'étiquette t_i. On poursuit la construction à partir de ces nouveaux noeuds ... .

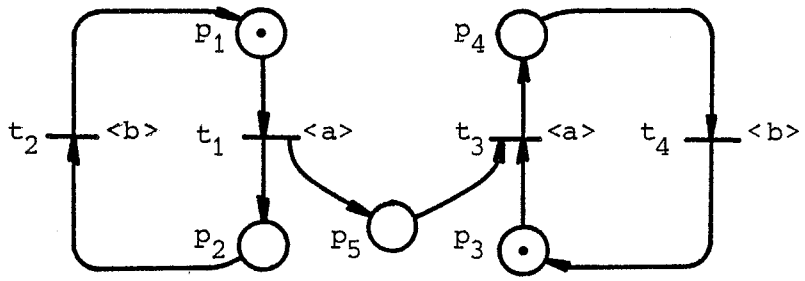
Au fur et à mesure de cette construction, si l'on rencontre un marquage M_s ayant un prédécesseur M_r tel que $M_s > M_r$ alors on remplace les composantes de M_s qui sont strictement supérieures à celles de M_r par le symbole ω qui représente une valeur aussi grande que l'on veut. Le fait qu'une composante prend la valeur ω signifie que la place correspondante est non bornée. La construction est arrêtée lorsque toutes les feuilles de l'arbre sont soit des marquages ne validant aucune transition, ce qui suffit à prouver que le réseau est non vivant, soit des marquages ayant un prédécesseur qui leur est égal.

Il est démontré que cette construction pour le cas des RdP ordinaires et des RdP généralisés se termine et, donc, que la propriété de borné est décidable pour ces classes de réseaux. Un simple examen de l'arbre permet de vérifier les propriétés de mutuelle exclusion et quelques types de vivacité faible [KEL.74]. La vivacité forte est décidable si le réseau est borné.

De telles méthodes ne peuvent en fait être envisagées en pratique que pour des réseaux de taille relativement réduite, étant donné le caractère fortement combinatoire de l'algorithme de construction de l'arbre -de complexité exponentielle en temps et en espace mémoire [LIP.76].

Une extension de ces méthodes aux RdPS serait de construire l'arbre des marquages en considérant, au lieu des opérations de mise à feu prises isolément, des opérations de tirs itérés sur occurrence d'événements. Cependant, dans ce cas, le critère "il existe un marquage M_s ayant un prédecesseur M_r tel que $M_s > M''_r$ ne permet pas de conclure que le

RdPS est non borné. Il suffit de vérifier ceci sur l'exemple donné en figure 20. [MOA.78]



$$M_0 = \begin{bmatrix}1\\0\\1\\0\\0\end{bmatrix} \quad \{a\}\{b\}// \quad \nrightarrow \quad M_2 = \begin{bmatrix}1\\0\\1\\0\\1\end{bmatrix} > M_0$$

; bien que le réseau soit borné pour M_0

Figure 20.

Ce critère ne peut pas, non plus, s'appliquer aux RdPZ. Il est d'ailleurs démontré dans [HAC.75] que la propriété de borné est indécidable pour cette classe de réseaux et pour toutes les extensions autonomes ayant une puissance équivalente à celle d'une machine de TURING. On peut utiliser ce résultat pour démontrer que la propriété de borné est également indécidable pour la classe des RdPS.

Théorème 1 :

La propriété de borné pour un marquage M est indécidable pour la classe des RdPS.

Démonstration :

Il suffit de montrer que quels que soient le RdPZ R_Z et M_z un marquage de R_Z, on peut trouver un RdPS R_S muni d'un marquage M_s et d'un environnement L tels que : décider que R_Z est borné pour M_z revient à décider que (R_S, L) est borné pour M_s.

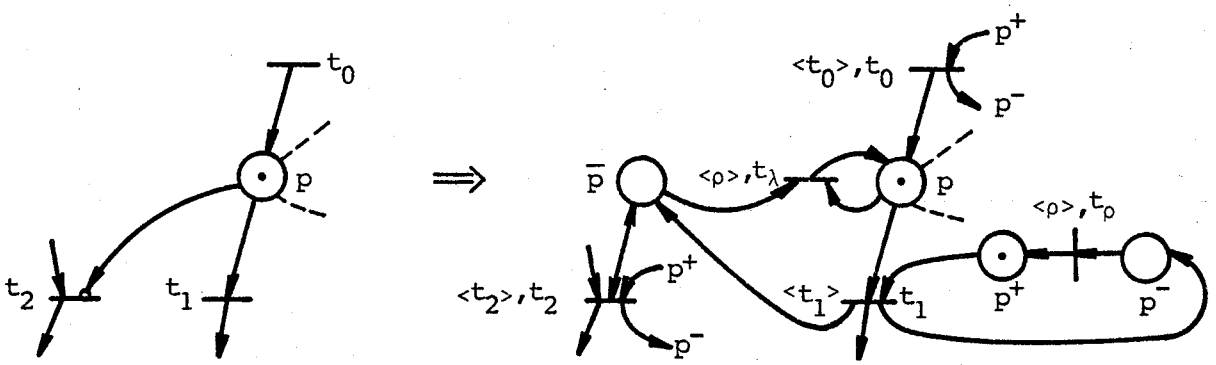
Soit $R_Z = (R, I)$ où $R = (P, T, \alpha, \beta)$. On effectue la construction du RdPS R_S en trois étapes : (figure 21).

1) On construit le RdPPE étiqueté $R_E = (R_e, V, f)$ avec $R_e = (P_e, T_e, \alpha_e, \beta_e)$ qui réalise R_Z, en procédant de la même manière que pour la démonstration de la proposition 1 (cf. chap. IV, § III).

2) On rajoute à R_E deux places p^+ et p^- et une transition t_ρ (étiquetée par λ_V) et on étend les relations d'incidence α_e et β_e de la façon suivante :

- $\forall\, t \in T : (p^+, t) \in \alpha_e$ et $(t, p^-) \in \beta_e$
- $(p^-, t_\rho) \in \alpha_e$ et $(t_\rho, p^+) \in \beta_e$

3) Le réseau R_S est obtenu à partir de R_E en associant à toute transition t_j de R_E non étiquetée par λ_V un événement externe distinct que l'on notera également par t_j, et à toutes les transitions étiquetées par λ_V un même événement externe ρ.



(a) dans R_Z (b) dans R_S

Figure 21.

Soient M_Z l'ensemble des marquages de R_Z et M_S l'ensemble des marquages de R_S. On définit l'application $g : M_Z \to M_S$ telle que si $g(M_{oz}) = M_{os}$ alors :

- $\forall$ p place de R_Z, $M_{os}(p) = M_{oz}(p)$,
- $\forall\, \bar{p}$ place "complémentaire" à p dans R_S, $M_{os}(\bar{p}) = 1$ si $M_{os}(p) = 0$ et 0 sinon,
- $M_{os}(p^+) = 1$ et $M_{os}(p^-) = 0$.

Il est aisé de vérifier que si M_z est un marquage de R_Z et M_s est un marquage de R_S tel que $M_s = g(M_z)$ alors toute séquence de simulation

$\sigma = t_1\ t_2 \ldots t_r$ à partir de M_z correspond à une séquence de tirs effectifs $\xi = \{t_1,e\}\{\rho,e\}\{t_2,e\}\ldots\{t_r,e\}\{\rho,e\}$ à partir de M_s et inversement. De plus, si $M_z \xrightarrow{\sigma} M'_z$ alors $M_s \xrightarrow{\xi//} M'_s = g(M'_z)$. Par conséquent, décider que R_Z est borné pour M_z revient à décider que (R_S, LL) est borné pour $M_s = g(M_z)$, d'où la démonstration du théorème.

Remarque :

Le résultat démontré est en fait plus fort que celui énoncé dans le théorème, puisqu'on montre que la propriété de borné est indécidable pour la sous-classe des RdPS totalement synchronisés.

II - 1.2. Méthodes d'analyse structurelle

Ces méthodes ont la particularité de s'appuyer sur l'examen de la structure des réseaux, indépendamment du marquage initial. Le principe consiste à déterminer dans le réseau des sous-ensembles de noeuds ou des sous-réseaux dont l'existence aide à la vérification des propriétés du réseau global.

Verrous et Trappes [HAC.72] [BES.75]

Un verrou est un ensemble de places du réseau telles que l'ensemble de leurs transitions d'entrée est contenu dans l'ensemble de leurs transitions de sortie. Une propriété caractéristique d'un verrou est que si toutes ses places sont vides (sans marques) pour un marquage donné alors elles le resteront pour tout marquage successeur.

La connaissance de l'ensemble des verrous minimaux permet de mettre en évidence un éventuel blocage partiel ou total : si, en partant du marquage initial, on trouve une séquence de mises à feu qui vide les places d'un verrou alors le réseau est non vivant.

Une trappe est un ensemble de places du réseau telles que l'ensemble de leurs transitions de sortie est contenu dans l'ensemble de leurs transitions d'entrée. Une propriété caractéristique d'une trappe est la suivante : si l'une de ses places est marquée par un marquage donné, alors elle aura toujours au moins une place marquée pour tout marquage successeur.

Une condition nécessaire et suffisante pour qu'un RdP libre choix soit vivant pour un marquage M_0 est que tout verrou contienne une

trappe ayant une place marquée par M_0 (théorème de COMMONER [HAC.72]).

Ce théorème appliqué aux graphes de transitions fortement connexes peut être énoncé comme suit : "une condition nécessaire et suffisante pour qu'un graphe de transitions fortement connexe soit vivant pour un marquage M_0 est que tout circuit élémentaire du graphe admette une place marquée par M_0". En effet, dans ce cas, tout circuit élémentaire est à la fois verrou minimal et trappe minimale.

Ces résultats restent vrais en passant du fonctionnement autonome à un fonctionnement synchronisé, temporisé ou non, sous l'environnement *LL*. Nous allons démontrer précisément que si un RdP R ordinaire simple est vivant pour un marquage M_0 alors : quel que soit le RdPS R_S totalement synchronisé construit à partir de R, (R_S, LL) est vivant pour M_0.

Lemme 1 :

Soit $R_S = (R, E, \mu)$ un RdPS totalement synchronisé construit à partir d'un RdP R ordinaire simple, et soit M_0 un marquage de R_S. Les propositions suivantes sont équivalentes :

P1 : t est non vivante pour M_0 dans (R_S, LL)

P2 : $\exists\ M \in \overrightarrow{M_0}$ et $p \in {}^\bullet t$ tels que $\forall\ M' \in \overrightarrow{M}$, $M'(p) = 0$.

Démonstration :

P1 se déduit de P2 de façon triviale puisque s'il existe une place p d'entrée de t qui est vide (ne contient pas de marques) pour tout marquage $M' \in \overrightarrow{M}$, cela veut dire que t est non vivante pour M et a fortiori pour M_0. (le réseau étant totalement synchronisé, on a $\overrightarrow{M} = \tilde{M} \subseteq \tilde{M}_0 = \overrightarrow{M_0}$).

Supposons que P2 n'est pas vérifiée. Cela se traduit par la proposition :

$\overline{P2}$: $\forall\ M \in \overrightarrow{M_0}$ et $\forall\ p \in {}^\bullet t$, $\exists\ M' \in \overrightarrow{M}$ tel que $M'(p) > 0$.

Soit ${}^\bullet t = \{p_1, p_2, \ldots, p_r\}$ l'ensemble des places d'entrée de t où p_r est la place d'entrée partagée éventuelle, et soit M un marquage de R, $M \in \overrightarrow{M_0}$. D'après $\overline{P2}$, et du fait que R est simple, on peut former de façon récurrente une suite de marquages $M = M_1, M_2, \ldots, M_r, M_{r+1}$ telle que :

$\forall\ i = 1,2,\ldots r$ et $\forall\ j = 1,2,\ldots i$, $M_{i+1} \in \overrightarrow{M_i}$ et $M_{i+1}(p_j) > 0$.

On peut donc trouver $M_{r+1} \in \overrightarrow{M}$ qui valide t. Ce raisonnement pouvant être appliqué pour tout marquage $M \in \overrightarrow{M_0}$, il en résulte que t est vivante pour M_0, donc $\overline{P1}$.

Théorème 2 :

Soit R = (P,T,α,β) un RdP ordinaire simple, et soit M_0 un marquage de R. Si R est vivant pour M_0 alors : quel que soit le RdPS R_S totalement synchronisé construit à partir de R, (R_S,*LL*) est vivant pour M_0.

Démonstration :

Supposons qu'il existe une transition t_0 de R_S qui soit non vivante pour M_0. D'après le lemme 1, il existe alors au moins une place $p_1 \in {}^\bullet t_0$ et un marquage $M_1 \in \overrightarrow{M_0}$ tels que : $\forall\, M \in \overrightarrow{M_1}$, $M(p_1) = 0$. Notons par P_1 l'ensemble constitué de la place p_1. A partir de (P_1,M_1), on peut définir de façon récurrente la suite (P_1,M_1) (P_2,M_2) ... (P_i,M_i) vérifiant :

$\forall\, i$, $P_i \subseteq P$ et $M_i \in \overrightarrow{M_0}$, et $\forall\, p_r \in P_i$, $\forall\, M \in \overrightarrow{M_i}$, $M(p_r) = 0$.

(P_{i+1},M_{i+1}) est défini à partir de (P_i,M_i) de la façon suivante : soit T_i l'ensemble des transitions d'entrée des places de P_i qui ne sont pas transitions de sortie de places de P_i. Si $T_i = \emptyset$ alors $P_{i+1} = P_i$ et $M_{i+1} = M_i$. Sinon, soit t_i une transition de T_i. Par définition de T_i, t_i est non vivante pour M_i ; il existe donc $p_{i+1} \in {}^\bullet t_i$, $p_{i+1} \notin P_i$, et $M' \in \overrightarrow{M_i}$ tels que : $\forall\, M \in \overrightarrow{M'}$, $M(P_{i+1}) = 0$. On prendra alors $P_{i+1} = P_i \cup \{p_{i+1}\}$ et $M_{i+1} = M'$. Le nombre de places du réseau étant fini, il existe un rang h tel que : $\forall\, k < h$ on a $P_{k+1} \neq P_k$ et $\forall\, \ell > h$ on a $P_\ell = P_h$. P_h est un verrou et M_h est un marquage de $\overrightarrow{M_0}$, donc de $\vec{M}_0$, tel que : $\forall\, p \in P_h$, $M_h(p) = 0$; ce qui implique que R est non vivant pour M_0.

Remarque :

Les démonstrations du lemme 1 et du théorème 2 peuvent être faites pour tout environnement *L* tel que : $\forall\, t \in T$ et $\forall\, \xi \in L$, $\exists\, \xi' \in (P(E))^*$ et $X \in P(E)$ tels que $\xi\xi' X \in L$ et $\mu(t) \in X$.

Composantes consistantes et Composantes conservatives [LAU.74] [RAM.73] [SIF.78]

Soit R un RdP ordinaire ou généralisé ayant un nombre fini de places et de transitions. Si l'on note les places par $p_1,p_2,\ldots p_n$ et les transitions par $t_1,t_2,\ldots t_m$, on peut associer à R une matrice $C = [c_{ij}]$ appelée matrice d'incidence définie telle que :

$$c_{ij} = W(t_j,p_i) - W(p_i,t_j)$$

(dans les RdP ordinaires, les poids des arcs définis par les relations α

et β valent tous 1).

Si M_0 est un marquage initial de R et M un marquage successeur atteint par application d'une séquence de mises à feu σ à partir de M_0, ($M_0 \xrightarrow{\sigma} M$), on a la relation $M = M_0 + CX$ où X est le vecteur dont la i-ème composante (i = 1,2,... m) est égale au nombre d'occurrences de t_i dans σ.

Soit X_c une solution à composantes entières non négatives de l'équation $CX = 0$. Le sous-réseau défini à partir des transitions correspondant à des composantes non nulles de X_c et des places qui leur sont adjacentes (d'entrée et de sortie) constitue ce qu'on appelle une composante consistante. De telles solutions correspondent à des séquences de mises à feu cycliques pour un marquage initial du réseau. Inversement, toute évolution cyclique est engendrée par une composante consistante.

Il est démontré dans [RAM.73] que si R n'est pas lui-même une composante consistante alors : pour tout marquage initial M_0, le réseau est ou non borné ou non vivant. Ce résultat reste évidemment valable quel que soit l'interprétation que l'on superpose au réseau.

Soit Y_c^t un vecteur ligne, à composantes non négatives, solution de l'équation $Y^tC = 0$ et soit P_1 l'ensemble des places du réseau correspondant aux composantes non nulles de Y_c^t. Le sous-réseau défini par les places de p_1 et les transitions qui leur sont adjacentes constitue ce qu'on appelle une composante conservative. En multipliant à gauche l'équation $M = M_0 + CX$ par le vecteur Y_c^t on obtient la relation $Y_c^t M = Y_c^t M_0$; ce qui signifie que la somme des nombres des marques des places de P_1 pondérée par les composantes correspondantes de Y_c^t est constante pour toute évolution à partir du marquage initial M_0.

On peut remarquer que, du fait que les composantes de Y_c^t sont non négatives, les places de P_1 sont obligatoirement bornées pour tout marquage initial. Si R est lui-même une composante conservative, c'est-à-dire tel qu'il existe Y_c^t solution de $Y^tC = 0$ ayant toutes ses composantes positives, alors il est borné pour tout marquage.

Les travaux de [MEM.77] et [SIF.78] suggèrent des méthodes pour le calcul des composantes conservatives. Une mise en oeuvre en est faite dans le programme OGIVE [BER.79] [CHE.79]. On trouve dans [SIF.79] d'autres résultats concernant les composantes conservatives ainsi que

les sous-réseaux définis par les solutions positives de $Y^tC < 0$ et $Y^tC > 0$.

Ces résultats étant démontrés pour tout fonctionnement possible de R, sont directement applicables à tout RdPI construit à partir de R.

II - 1.3. Méthodes d'analyse par transformation [BER.76] [BER.78] [SZL.77] [SIF.2.77]

On note par (R,M_0) le réseau R à analyser muni d'un marquage initial M_0. Les méthodes d'analyse par transformation consistent à transformer le doublet (R,M_0) en un doublet (R',M'_0) tel que :

1) Si (R',M'_0) satisfait la propriété PR à vérifier, alors (R,M_0) la satisfait nécessairement.
2) La propriété PR est plus facilement vérifiable sur (R',M'_0) que sur (R,M_0).

Les méthodes développées dans [BER.76] [BER.78] [SZL.77] consistent à réduire la complexité du réseau initial (en nombre de places et de transitions) en lui appliquant progressivement des règles de transformation locale qui préservent les propriétés de borné et de vivant. Ces méthodes ont l'avantage d'être d'utilisation simple et directe. Cependant, leur efficacité dépend de la structure du réseau initial : on peut trouver des RdP qui ne sont réductibles par aucune de ces règles de transformation.

La méthode proposée dans [SIF.78] permet d'obtenir, par homomorphisme à partir de la matrice d'incidence du réseau initial R, les matrices d'incidence de réseaux R' qui réalisent R. Ces réseaux ne sont pas nécessairement de taille plus réduite, mais leur structure rend plus aisée la vérification des propriétés de borné et de vivant.

Les principes de ces méthodes, étudiés pour le cas des RdP autonomes, peuvent difficilement être généralisés aux RdP non-autonomes. Néanmoins, ces méthodes, comme toutes les autres méthodes d'analyse statique que nous venons de rappeler, sont utiles quand elles permettent de prouver que le RdP R étudié est borné pour un marquage M_0 ; cette condition étant suffisante pour que tout RdP non-autonome construit à partir de R soit borné pour M_0.

D'une façon générale, la vérification par analyse statique d'un système décrit par un RdPI ne peut couvrir que la propriété de borné, des propriétés d'exclusions mutuelles et des propriétés de distances synchroniques. Pour ces propriétés, les méthodes que nous venons de rappeler permettent d'obtenir, selon le cas, des conditions nécessaires et/ou suffisantes de vérification.

L'application systématique de ces méthodes, a posteriori de la description, peut s'avérer trop lourde pour des réseaux dépassant quelques dizaines de places et de transitions. Néanmoins, on peut s'imposer, dès la description, des restrictions sur la structure du réseau de façon soit à faciliter l'analyse ultérieure, soit à garantir, par avance, la vérification des propriétés recherchées [LAU.E.75] [VAL.76].

La vérification de la propriété de vivacité est plus difficile, étant très liée à l'interprétation superposée au réseau. Des solutions pourraient, éventuellement, être trouvées dans la combinaison des méthodes d'analyse applicables aux RdP autonomes et des résultats obtenus par ailleurs en analyse sémantique des programmes ; cette voie est encore peu explorée à l'heure actuelle [COU.78] [SIF.79] [COU.80]. Dans beaucoup de cas, le recours à la simulation reste inévitable.

II - 2. VERIFICATION PAR SIMULATION

La vérification par la simulation permet de prendre en compte tout ce qui est jugé significatif dans le comportement fonctionnel et dynamique du système décrit. En revanche, comme nous l'avons dit plus haut, les résultats que l'on obtient ne peuvent être que partiels. A moins que l'interprétation du RdP qui décrit le système soit relativement simple et que le système soit destiné à avoir un fonctionnement périodique, auquel cas la simulation peut être faite de façon exhaustive.

On peut envisager au moins deux manières de vérifier les propriétés recherchées du RdPI :

- Pour un contexte de fonctionnement donné, (ou pour un ensemble de contextes jugés représentatifs de l'ensemble des fonctionnements possibles du système), on simule le RdPI et on surveille directement la vérification de ces propriétés au cours de la simulation : On peut vérifier que le nombre de marques dans chaque place reste toujours inférieur

ou égal à une borne donnée. On peut relever les fréquences relatives des mises à feu des transitions, détecter des situations de non-persistance ou de blocage ... . La signification du diagnostic croît avec la longueur et l'exhaustivité de la simulation.

On peut améliorer cette vérification en effectuant une analyse préalable du réseau afin de caractériser les situations de violation de ces propriétés. On peut déterminer, par exemple, les sous-ensembles du réseau correspondant à des verrous ou à des conflits et vérifier les propriétés de ces sous-ensembles à chaque pas de la simulation. Le diagnostic que l'on peut obtenir est alors plus significatif. En particulier, des situations de blocage partiel qui n'auraient pas pu être détectées par une simple observation de la simulation pourront ainsi être mises en évidence.

- L'autre manière consiste à examiner de façon semi-exhaustive les possibilités de violation de ces propriétés : Partant du marquage initial, et sans soumettre la simulation à un contexte de fonctionnement particulier du système décrit par le RdP, on vérifie qu'aucun des marquages successeurs possibles ne peut conduire à la violation de ces propriétés. La simulation est poursuivie ensuite en choisissant un marquage successeur au hasard.

III - IMPLANTATION DE REDONDANCE

La recherche d'une réalisation à fonctionnement sûr d'un système amène, au-delà de la vérification de la bonne conception, à introduire des mécanismes de détection et de recouvrement d'erreurs. Ces mécanismes permettent surtout de se prémunir contre des fonctionnements dangereux, et parfois imprévisibles, de pannes du matériel pouvant se produire en cours de fonctionnement. Il est, par conséquent, intéressant de pouvoir les étudier au niveau d'une description fonctionnelle du système, indépendamment de sa réalisation. Des travaux dans ce sens ont été effectués pour les systèmes décrits à l'aide de RdP. Une erreur est traduite par la "génération" ou la "disparition" de marques dans le réseau, ou encore par la modification de sa structure (par exemple, une coupure d'arc). [MAR.75] [JAC.76] [MER.76] [HAN.77] [SIF.2.77].

Remarquons que les méthodes déjà développées pour les machines

séquentielles s'appliquent aux RdP mais de façon indirecte et sous certaines contraintes : si l'on peut décomposer le RdP en graphes d'états saufs ayant chacun une seule place marquée, ou si l'on peut transformer le RdP en une machine séquentielle qui le réalise. On connaît, à l'heure actuelle, deux méthodes directement applicables aux RdP :

- La méthode développée dans [SIF.2.77] qui consiste à transformer le réseau de départ R muni d'un marquage initial M_0 en un réseau équivalent R' muni d'un marquage M'_0 tel que l'ensemble $\vec{M}'_0$ des marquages successeurs ait une distance de HAMMING désirée d. Il devient alors possible de détecter toutes les erreurs de multiplicité inférieure ou égale à d-1 ou corriger des erreurs de multiplicité inférieure ou égale à $\lfloor (d-1)/2 \rfloor$ (résultats classiques sur les codes, [PET.72]). Cette transformation est systématique et s'effectue à partir de la matrice d'incidence représentant le réseau.

Une erreur de multiplicité m correspond à la modification d'un marquage correct M'_1 en un marquage M''_1 tel que M'_1 et M''_1 diffèrent exactement en m composantes. Une erreur de multiplicité 1 -dite erreur simple- traduit donc la génération ou la disparition imprévisible de marques dans une place du réseau.

- La deuxième méthode proposée dans [MAR.75] s'applique uniquement aux RdP saufs et consiste à associer à chaque place du réseau un poids (nombre entier positif) de telle manière que la somme des poids des places marquées reste constante pour tout marquage successeur du marquage initial M_0. Si cette association n'est pas possible directement sur le réseau de départ, un arrangement peut toujours être trouvé moyennant l'adjonction de places fictives.

Les deux méthodes se ramènent, en fait, à vérifier des relations invariantes du type $Y^tM = Y^tM_0$ dues à l'existence de composantes conservatives. Les poids associés aux places, dans la deuxième méthode, ne sont pas autre chose que les composantes d'une solution Y_c^t de $Y^tC = 0$. Si le RdP n'est pas lui-même une composante conservative, ou si l'exploitation des relations invariantes du RdP initial ne permet pas d'atteindre le degré de tolérance aux erreurs souhaité, des places fictives sont rajoutées de manière à créer de nouvelles composantes conservatives. Notons que

ces méthodes sont applicables indépendamment du mode de fonctionnement autonome ou non-autonome du RdP.

L'intérêt de cette redondance est fonction, bien sûr, des réponses que l'on peut donner aux deux questions suivantes :

1°) Dans quelle mesure les erreurs détectées par la disparition ou la génération de marques, ou par la modification de la structure du réseau, couvrent-elles l'ensemble des erreurs possibles ?

2°) Est-il toujours possible de lier les erreurs de multiplicité m, étudiées au niveau d'une description fonctionnelle d'un système, aux pannes du matériel dans une réalisation donnée ?

Dans le cas général d'un système décrit par un RdPI, les réponses à ces deux questions vont dépendre nécessairement, d'une part, de la complexité de l'interprétation associée au RdPI et, d'autre part, de la fidélité avec laquelle la structure et l'algorithme de fonctionnement du RdPI sont traduits dans la structure et l'algorithme de fonctionnement de la réalisation.

- § -

CHAPITRE VII

GRAFCET ET RdPI

I - COMPARAISON SELON LE POINT DE VUE DES POSSIBILITES DE DESCRIPTION 1

I - 1. REPRESENTATION DES SITUATIONS ET DES EVOLUTIONS ENTRE SITUATIONS 1

I - 2. PRISE EN COMPTE DES OCCURRENCES D'EVENEMENTS EXTERNES 8

I - 3. INTERPRETATION DES ACTIONS ASSOCIEES AUX ETAPES (PLACES) ET PRISE EN COMPTE DU FACTEUR TEMPS 10

I - 4. EQUIVALENCE DES MODELES GRAFCET ET RdPI SAUFS ET PERSISTANTS SOUS L'ENVIRONNEMENT LE PLUS STRICT 11

II - COMPARAISON SELON LE POINT DE VUE DES POSSIBILITES D'ANALYSE ET DE VERIFICATION 14

III - CONCLUSION 16

Comme nous l'avons dit, la définition du grafcet s'est largement inspirée de celle des RdPI. La motivation originale de la définition du grafcet est essentiellement d'apporter des possibilités de description plus directes et mieux adaptées au contexte de la spécification des automatismes que celles qui sont offertes par les RdPI. En revanche, ces facilités réduisent les possibilités d'analyse et de vérification qui font des RdPI un meilleur outil de conception.

Dans ce chapitre, nous allons résumer les points communs et les différences des deux modèles ; ce qui nous permettra, dans un premier temps, d'illustrer les avantages du grafcet par rapport aux RdPI selon le point de vue des possibilités de description. Nous comparerons ensuite les deux modèles selon le point de vue des possibilités d'analyse et de vérification.

I - COMPARAISON SELON LE POINT DE VUE DES POSSIBILITES DE DESCRIPTION

Nous distinguerons les trois critères suivants :

1. Représentation des situations (états) et des évolutions entre situations d'un système.
2. Prise en compte des occurrences d'événements externes.
3. Interprétation des actions associées aux étapes (resp. places) et prise en compte du facteur temps.

I - 1. REPRESENTATION DES SITUATIONS ET DES EVOLUTIONS ENTRE SITUATIONS

La représentation des situations d'un système et des évolutions possibles entre ces situations est au centre de l'intérêt porté aux deux modèles. On peut, par conséquent, considérer ce premier critère de comparaison comme étant le critère essentiel.

De ce point de vue, on peut souligner deux points communs et trois différences.

Points communs :

c1 - Comme pour un RdPI, un grafcet comporte deux types de noeuds qui sont les étapes et les transitions, les étapes correspondant aux places dans un RdPI. Les conventions de représentation graphique, du moins celles ordinairement utilisées (cf. chap. II), sont quasiment les mêmes.

c2 - Les évolutions des marquages dans les deux modèles sont synchronisées sur les occurrences d'événements :

a) toute occurrence d'un événement externe oblige à effectuer, instantanément, un tir itéré sur occurrence de cet événement ;

b) une nouvelle occurrence d'événement externe ne peut être prise en compte tant que le grafcet (resp. le RdPI) n'a pas atteint une situation stable.

Différences :

d1 - Dans un grafcet, les transitions franchissables considérées lors d'un tir sur occurrence d'événement sont toutes franchies, ce qui n'est pas toujours le cas dans un RdPI non persistant.

Cela tient au fait qu'une marque dans un RdPI est considérée comme un objet indivisible qui ne peut être utilisé que par une seule opération de mise à feu à la fois : si, lors d'un tir sur occurrence d'événement, une place contenant une marque se trouve en conflit entre deux transitions réceptives à cet événement alors une seule des deux transitions sera franchie. La notion de conflit n'existe pas dans le modèle grafcet : une marque sert simplement à représenter l'état d'activité d'une étape, et une étape active peut participer à la mise à feu de toutes les transitions de sortie qui sont simultanément franchissables ; dans ce sens, la marque est divisible à volonté.

d2 - Dans un RdPI, les marques s'accumulent dans les places. A tout moment, l'état d'une place est déterminé par le nombre de marques qu'elle contient. Les RdPI saufs sont conçus de façon à ce que les situations telles qu'une place contient plus d'une marque n'arrivent jamais. Dans un grafcet, une étape a deux états possibles : active (étape contenant une marque) ou inactive (étape sans marque). Mais le fait qu'une étape n'a jamais plus d'une marque ne découle pas d'une construction particulière du grafcet : par définition, les marques dans une même étape se confondent.

d3 - Enfin, dans un RdPI, les réceptivités associées aux transitions ne peuvent porter que sur les valeurs des variables du sous-système opératif et des variables d'interaction avec l'environnement externe, excluant toute considération sur l'état (en nombre de marques) des places. Dans un grafcet, les réceptivités peuvent porter tant sur les valeurs des variables internes et externes que sur l'état d'activité ou d'inactivité des étapes.

Les deux premières différences rapprochent déjà le grafcet des notions couramment manipulées par les automaticiens. En effet, dans la description du fonctionnement d'un automatisme (et notamment au niveau de la spécification du cahier des charges), la notion de conflit en partage de ressource ou la notion de comptage des événements qui conduisent à l'activation d'une étape interviennent plutôt rarement. Par contre, il est courant que l'on veuille considérer une étape comme un "niveau logique" dont seule l'information sur son état (actif ou inactif) est significative. Dès lors, les conventions de fusionnement des marques à l'entrée d'une étape et de partage d'une marque en sortie d'une étape

apparaîssent plus appropriées. Les exclusions mutuelles entre des séquences d'actions issues d'une même étape sont toujours résolues de façon déterministe par des réceptivités incompatibles associées aux transitions de sortie de l'étape.

La troisième différence apporte des simplifications intéressantes dans la représentation d'automatismes complexes, au détriment certes de la visualisation du flot de contrôle.

Exemple :

Conditionner le franchissement d'une transition par le fait qu'une place contient une marque (test à "1") ou n'en contient pas (test à "0") est lourd à spécifier avec les RdPI ; alors que cela peut être fait de façon concise et directe à l'aide des variables φ associées aux étapes dans un grafcet (figure 22).

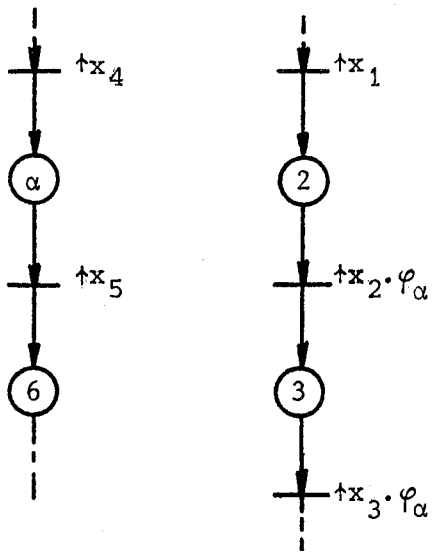
Un fait important à souligner est que la représentation des tests à "1" et des tests à "0" dans un RdPI doit toujours tenir compte de l'interprétation superposée au réseau, ce qui n'est pas le cas dans un grafcet. Ceci est illustré clairement dans les exemples donnés en figure 22. Dans l'exemple du grafcet de la figure 22-a, on veut que la transition issue de l'étape 2 (resp. de l'étape 3) ne soit considérée comme étant franchissable à l'instant d'une occurrence de l'événement $\uparrow x_2$ (resp. de l'événement $\uparrow x_3$) que si l'étape α est active à cet instant. En introduisant la variable φ_α dans les réceptivités associées à ces transitions, on exprime cette contrainte et rien d'autre, et ceci est valable quels que soient les liens qui peuvent exister entre les événements $\uparrow x_2$, $\uparrow x_3$, $\uparrow x_4$ et $\uparrow x_5$ et quelles que soient les actions associées aux étapes 2, 3 et α. La représentation RdPI de ce même fonctionnement, donnée par la figure 22-b, ne peut être utilisée de façon systématique que si :

1°) les événements $\uparrow x_2$ et $\uparrow x_5$ d'une part, et $\uparrow x_3$ et $\uparrow x_5$ d'autre part, sont incompatibles ;

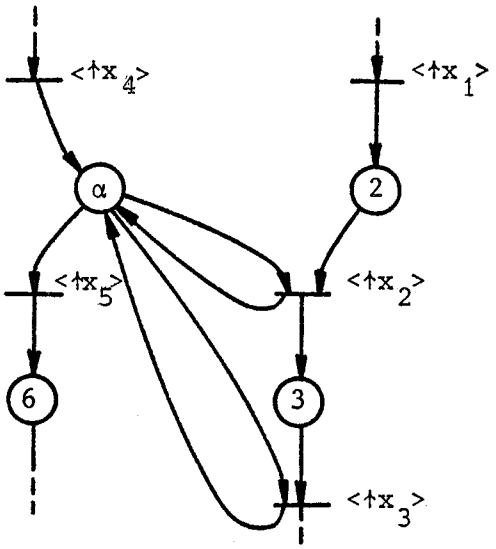
2°) les temps d'indisponibilité des marques dans la place α sont toujours nuls ;

3°) l'opérateur associé à la place α est l'opérateur identité.

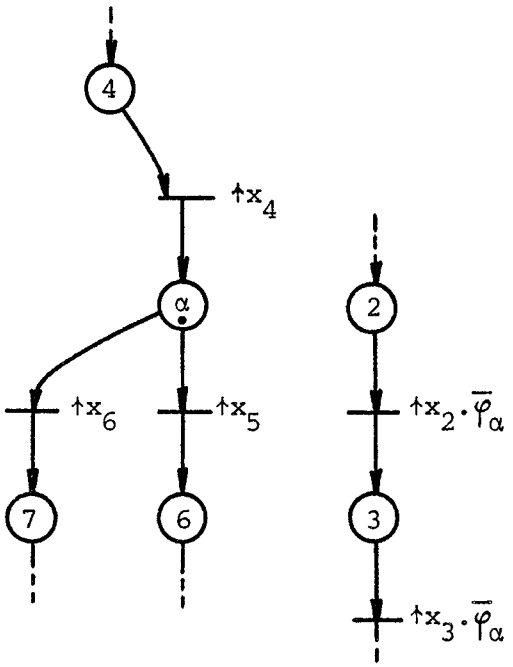
Dans le cas où les conditions 2 et/ou 3 ne sont pas vérifiées, il faut trouver une autre représentation, par exemple celle donnée par la



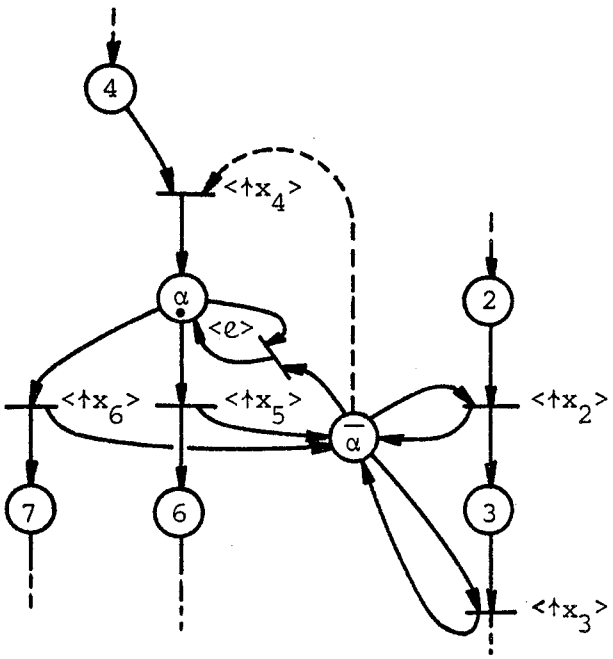
(a) Test à "1" d'une étape α dans un grafcet



(b) Test à "1" d'une place α dans un RdPI sauf. (on suppose que $\uparrow x_2$ et $\uparrow x_3$ sont incompatibles avec $\uparrow x_5$)



(c) Test à "0" d'une étape α dans un grafcet



(d) Test à "0" d'une place α dans un RdPI sauf. (on suppose que $\uparrow x_2$, $\uparrow x_3$, $\uparrow x_5$ et $\uparrow x_6$ sont différents de e)

Figure 22. Représentation du test à "1" et du test à "0" dans un grafcet et dans un RdPI sauf

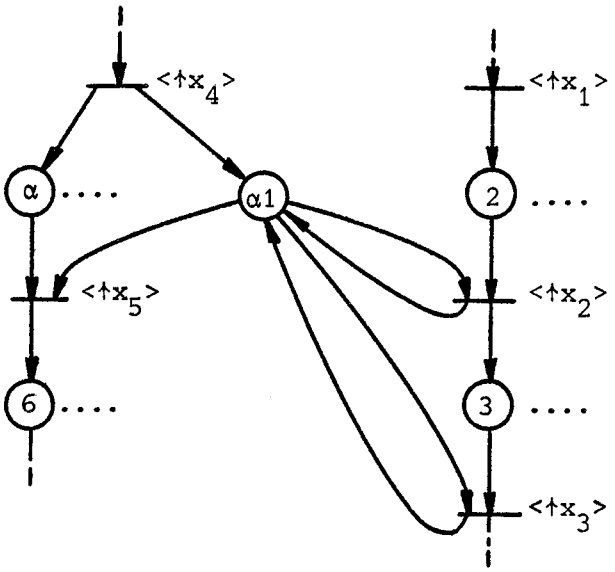
figure 23-a. Si, de plus, la condition 1 n'est pas vérifiée —par exemple, parce que $\uparrow x_2$ et $\uparrow x_5$ sont identiques et sont incompatibles avec $\uparrow x_3$— il faut encore transformer cettre représentation pour obtenir, par exemple, celle donnée par la figure 23-b.

Revenons à l'exemple de représentation du test à "0" donné dans la figure 22-d. Cette représentation suppose que les événements $\uparrow x_2$, $\uparrow x_3$, $\uparrow x_5$ et $\uparrow x_6$ sont différents de l'événement e. On aurait pu remplacer la transition à laquelle est associé l'événement e par un arc joignant la place $\overline{\alpha}$ à la transition d'entrée de la place α (arc représenté en pointillé), à condition que le fonctionnement du RdPI soit tel que les places 4 et α ne puissent jamais être marquées simultanément et que l'événement $\uparrow x_4$ soit incompatible avec les événements $\uparrow x_2$ et $\uparrow x_3$. La représentation grafcet du même test à "0", donnée par la figure 22-c, ne présuppose aucune condition ni sur le marquage ni sur les liens entre événements.

Une autre façon d'exprimer le test à "1" et le test à "0" dans un RdPI consiste à introduire des "variables de synchronisation" dans le sous-système opératif et à traduire les tests à "1" et tests à "0" par des tests sur les valeurs de ces variables de synchronisation (figure 24). Mais là encore, des précautions sont à prendre : la représentation du test à "1" dans l'exemple de la figure 24-a n'est correcte que si les événements $\uparrow x_4$ et $\uparrow x_5$ sont incompatibles ou si le marquage du réseau est tel que les places α et 6 ne peuvent jamais être marquées simultanément. Il en est de même pour la représentation du test à "0" dans l'exemple de la figure 24-b.

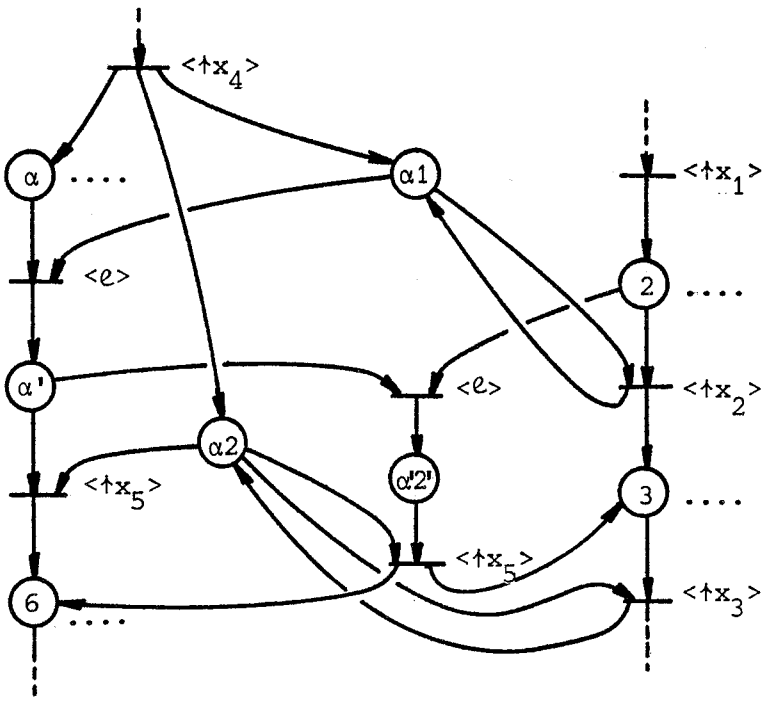
L'utilisation du test à "1" et du test à "0" a des applications fréquentes dans la représentation des spécifications du cahier des charges d'un automatisme. Elle apparaît de façon générale dans la description des contraintes de synchronisation entre des séquences d'actions. Les exemples ci-dessus montrent que la possibilité d'exprimer ces tests à l'aide des variables φ allège considérablement la représentation graphique et limite les risques d'erreur. Un autre avantage de l'utilisation des variables φ est de permettre une représentation structurée d'un problème complexe :

- Il est fréquent qu'un automatisme ait à synchroniser la commande



La place α1 qui est ajoutée est telle que :
$\nu_{\alpha 1}(\tau) \equiv \tau$
et
$\varphi(\alpha 1) \equiv$ op.identité

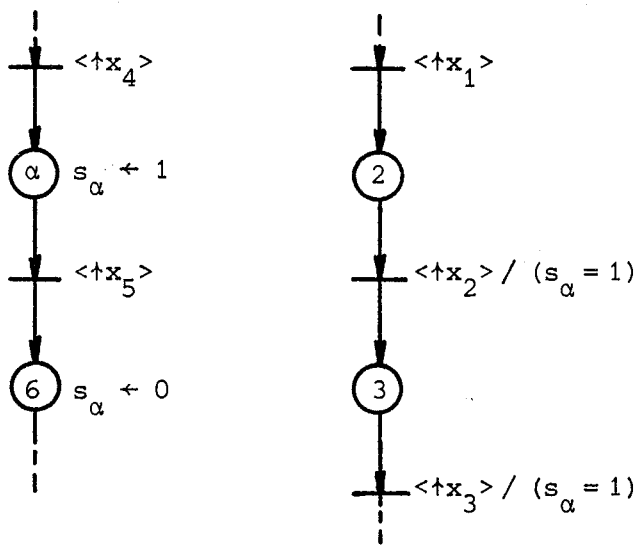
(a) Exemple de représentation du test à "1" d'une place α dans un RdPI sauf. Cas où $\nu_\alpha(\tau) \neq \tau$ et/ou $\varphi(\alpha) \neq$ op.identité et $\uparrow x_5$ est incompatible avec les événements $\uparrow x_2$ et $\uparrow x_3$



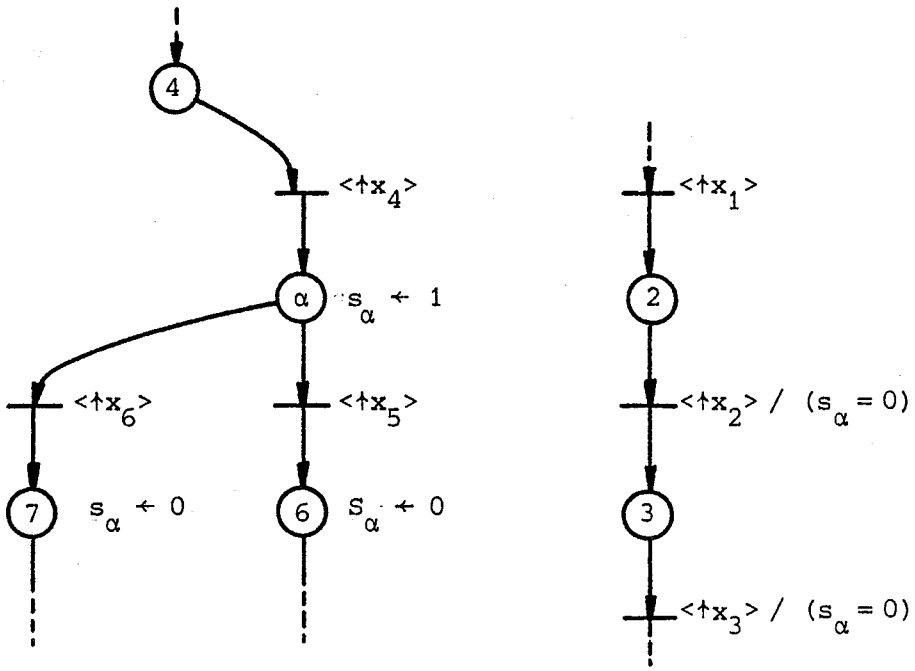
Les places α', α1, α2 et α'2' sont telles que :
$\nu_p(\tau) \equiv \tau$
et
$\varphi(p) \equiv$ op.identité

(b) Exemple de représentation du test à "1" d'une place α dans un RdPI sauf. Cas où $\nu_\alpha(\tau) \neq \tau$ et/ou $\varphi(\alpha) \neq$ op.identité et $\uparrow x_2$ et $\uparrow x_5$ sont identiques et incompatibles avec $\uparrow x_3$

Figure 23.



(a) *Test à "1" d'une place α dans un RdPI sauf. Représentation à l'aide de la variable de synchronisation* s_α *(exemple de la figure 22-a)*



(b) *Test à "0" d'une place α dans un RdPI sauf. Représentation à l'aide de la variable de synchronisation* s_α *(exemple de la figure 22-c)*

Figure 24.

de deux ou plusieurs modules fonctionnels d'un procédé. Une représentation structurée voudrait que la description des fonctions de commande de chaque module soit donnée de façon séparée. Grâce à l'utilisation des variables φ, on peut décrire de façon rigoureuse la synchronisation entre ces fonctions tout en maintenant leurs représentations séparées. (le grafcet G1, en page 101 de l'annexe, présente un exemple type de ce genre d'utilisation).

- Il est fréquent aussi qu'un automatisme ait plusieurs modes de fonctionnement possibles (manuel, automatique continu, automatique pas-à-pas ou par cycle, ...). Quand c'est le cas, il est intéressant de donner de façon séparée (et une fois pour toute) le grafcet représentant les modes de fonctionnement et les modalités de passage d'un mode à un autre, notamment si ces modalités doivent suivre un processus séquentiel. La prise en compte du mode actif dans la représentation des fonctions de commande proprement dites se ferait de façon simple à l'aide des variables φ associées aux étapes du grafcet représentant les modes de fonctionnement. Le test sur les valeurs de ces variables pourraît être utilisé dans les conditions d'exécution des actions et dans les réceptivités associées aux transitions.

- De la même manière, on peut représenter de façon séparée des automates de surveillance (cf. chap. III, § IV).

Ces trois premières différences que nous venons d'étudier sont les seules concernant la représentation des situations et des évolutions entre situations. Il apparaît clairement qu'elles sont à l'avantage du grafcet du point de vue des facilités de description.

Les autres différences qui vont suivre illustrent davantage les orientations voulues pour chacun des deux modèles : la spécification de cahiers des charges pour le grafcet, la description et l'analyse de solutions de conception pour les RdPI.

I - 2. PRISE EN COMPTE DES OCCURRENCES D'EVENEMENTS EXTERNES

d4 - Dans un grafcet, les occurrences des événements élémentaires externes sont considérées comme étant toujours disjointes. Par contre, dans un RdPI on admet que des occurrences d'événements élémentaires externes puissent être prises en compte simultanément.

On ne peut, en fait, parler de simultanéité d'occurrences d'événements que relativement à des "règles d'arbitrage" qui permettent de décider, de façon unique, quand les occurrences d'événements distincts doivent être considérées comme étant simultanées.

En ce qui concerne les événements internes à un système, que ce système soit décrit à l'aide d'un grafcet ou à l'aide d'un RdPI, les règles d'arbitrage sont les mêmes : deux occurrences d'événements internes sont considérées comme étant simultanées si ces occurrences sont engendrées lors de l'exécution d'un même tir, ou si ces occurrences sont associées à une même date d'échéance liée à l'occurrence antérieure d'autres événements (Par exemple, dans un grafcet, les événements $\uparrow T^t_\alpha$ et $\uparrow T^t_\beta$ ont une occurrence simultanée à l'instant τ si, durant l'intervalle $[\tau-t,\tau[$, les étapes α et β ont été activées pour la dernière fois lors d'un même tir effectué à l'instant $\tau-t$. De même, dans un RdPI, on peut trouver, associées à une même date, les échéances du passage de plusieurs marques de l'état indisponible à l'état disponible).

Les occurrences des événements externes échappent entièrement au contrôle du système et, de ce fait, des règles d'arbitrage les concernant ne peuvent pas être établies de façon absolue.

C'est pourquoi, dans le contexte de la spécification du cahier des charges d'un automatisme et afin de garantir une interprétation unique de ces spécifications, on convient de considérer les occurrences d'événements externes comme étant toujours disjointes. Le grafcet, défini pour servir essentiellement d'outil de spécification de cahiers des charges, s'impose cette règle.

Mais comme nous l'avons dit, le modèle RdPI se veut être surtout un outil de description et d'analyse de solutions de conception. D'une façon quasi-générale, et notamment dans les réalisations programmées, la détection des occurrences des événements externes est confiée à un "interface" conçu à ces fins (Par exemple, un sous-système de gestion d'interruptions, ou un processus de scrutation périodique des entrées ...). Pour des raisons évidentes de performances, on s'arrange pour traiter les occurrences des événements non pas une par une mais plusieurs simultanément ; l'interface en question décide au fur et à mesure les occurrences qu'il faut traiter simultanément et assigne à ces

occurrences une même date d'occurrence.

La définition de cet interface lors de la conception doit bien sûr garantir que le comportement de l'automatisme ainsi réalisé est conforme à celui déduit de l'interprétation des spécifications du cahier des charges. Cet interface est formalisé dans le modèle RdPI par la notion d'"environnement" consistant à associer à chaque événement externe une suite de dates d'occurrence.

L'"environnement le plus strict" (*LS*, voir chap. V) associe aux occurrences des événements externes différents de ε des dates distinctes. L'"environnement le moins contraignant" (*LL*) n'interdit aucune combinaison d'occurrences simultanées d'événements externes.

I - 3. INTERPRETATION DES ACTIONS ASSOCIEES AUX ETAPES (PLACES) ET PRISE EN COMPTE DU FACTEUR TEMPS

d5 - Dans un RdPI, les opérateurs associés aux places ne permettent de décrire que des actions "instantanées" et ininterruptibles. Cette interprétation correspond à celle des actions de "calcul" dans un grafcet. Le grafcet autorise, en plus, l'utilisation des concepts d'action à niveau (action effectuée tant que l'étape à laquelle elle est associée est active) et d'action impulsionnelle (action à durée limitée).

d6 - Dans un grafcet, la prise en compte du facteur temps est effectuée tant au niveau des actions impulsionnelles qu'au niveau des échéances de franchissement des transitions. Dans un RdPI, le facteur temps ne peut être pris en compte qu'à travers les durées d'indisponibilité des marques dans les places.

Il s'agit là encore de différences qui rapprochent le grafcet des besoins d'une représentation directe des spécifications du cahier des charges d'un automatisme. En revanche, une description en termes de RdPI est plus directement réalisable. Aussi, le fait que les contraintes de temps soient traduites exclusivement par des durées d'indisponibilité des marques dans les places facilite une évaluation du comportement dynamique d'un système décrit par un RdPI. Une telle évaluation est difficile à faire à partir d'une description en grafcet.

Il convient maintenant d'étudier l'impact de l'ensemble de ces différences sur la puissance théorique de description offerte par chacun des deux modèles.

I - 4. EQUIVALENCE DES MODELES GRAFCET ET RdPI SAUFS ET PERSISTANTS SOUS L'ENVIRONNEMENT LE PLUS STRICT

Proposition 8 :

Tout fonctionnement décrit par un grafcet peut avoir une représentation grafcet équivalente dans laquelle :

1°) une étape n'est activée qu'au plus une fois à chaque tir et n'est jamais réactivée tant qu'elle est active (pas de fusionnement de marques) ;

2°) une étape active ne participe qu'au franchissement d'une seule transition à la fois (pas de partage de marque) ;

3°) aucun test à "1" ou test à "0" n'est exprimé à l'aide des variables φ.

Démonstration :

Soit G1 un grafcet quelconque donnant une représentation initiale de ce fonctionnement, et soit n le nombre des étapes de ce grafcet. On peut identifier chacune de ces étapes par un numéro allant de 1 à n. De même, on peut identifier chaque situation possible du grafcet par la suite ordonnée des n numéros dans laquelle chaque numéro i apparaît sous sa forme normale (i) si pour cette situation l'étape i est active et sous la forme barrée ($\overline{i}$) sinon. (Exemple : si n = 4 alors la suite $\overline{1}$, 2, $\overline{3}$, $\overline{4}$ identifie la situation dans laquelle l'étape 2 est active et les étapes 1, 3 et 4 sont inactives). On peut avoir au maximum 2^n situations possibles différentes.

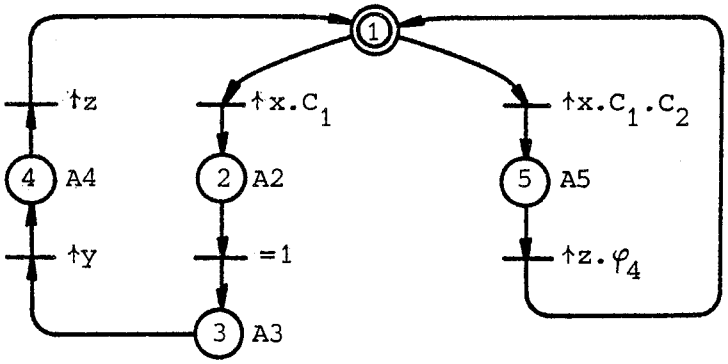
On construit progressivement la nouvelle représentation grafcet équivalente en suivant le schéma suivant : (voir exemple en figure 25)

a) à chaque situation possible dans le grafcet initial, on associe une étape dans la nouvelle représentation.

b) on peut toujours déterminer la réceptivité qui permet d'évoluer d'une situation s à une autre situation s' différente. Sur le nouveau grafcet, on représente cette évolution par une transition portant cette réceptivité et ayant un arc d'entrée issu de s et un arc de sortie vers s'.

c) à chaque étape i du grafcet initial, on fait correspondre une

Grafcet initial :

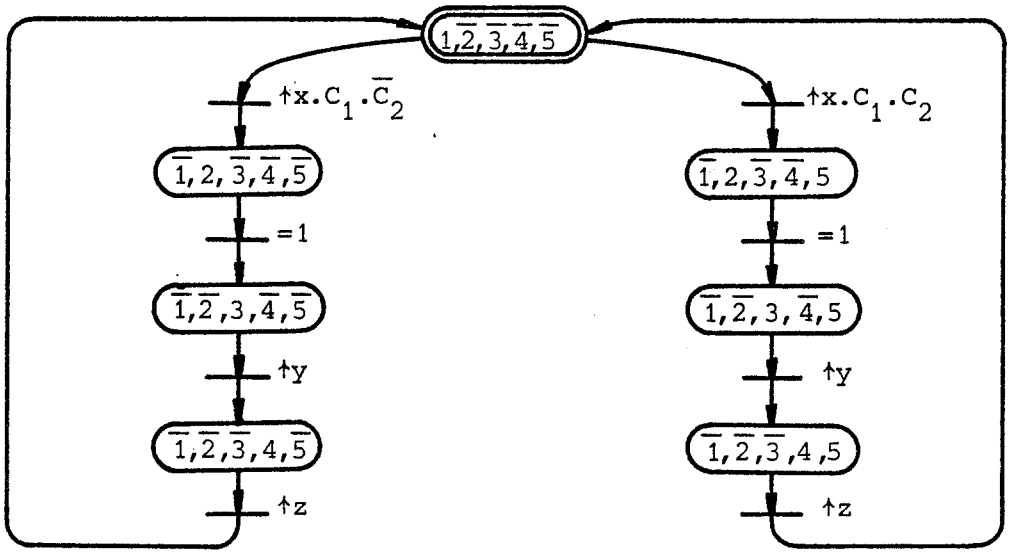


a) *Situations possibles :*

$\{1,\overline{2},\overline{3},\overline{4},\overline{5}\}$, $\{\overline{1},2,\overline{3},\overline{4},\overline{5}\}$, $\{\overline{1},\overline{2},3,\overline{4},\overline{5}\}$, $\{\overline{1},\overline{2},\overline{3},4,\overline{5}\}$, $\{\overline{1},2,\overline{3},\overline{4},5\}$,

↑ situation initiale — situations instables — $\{\overline{1},\overline{2},3,\overline{4},5\}$, $\{\overline{1},\overline{2},\overline{3},4,5\}$.

b) *Grafcet des situations :*



c) *Grafcet équivalent au grafcet initial :*

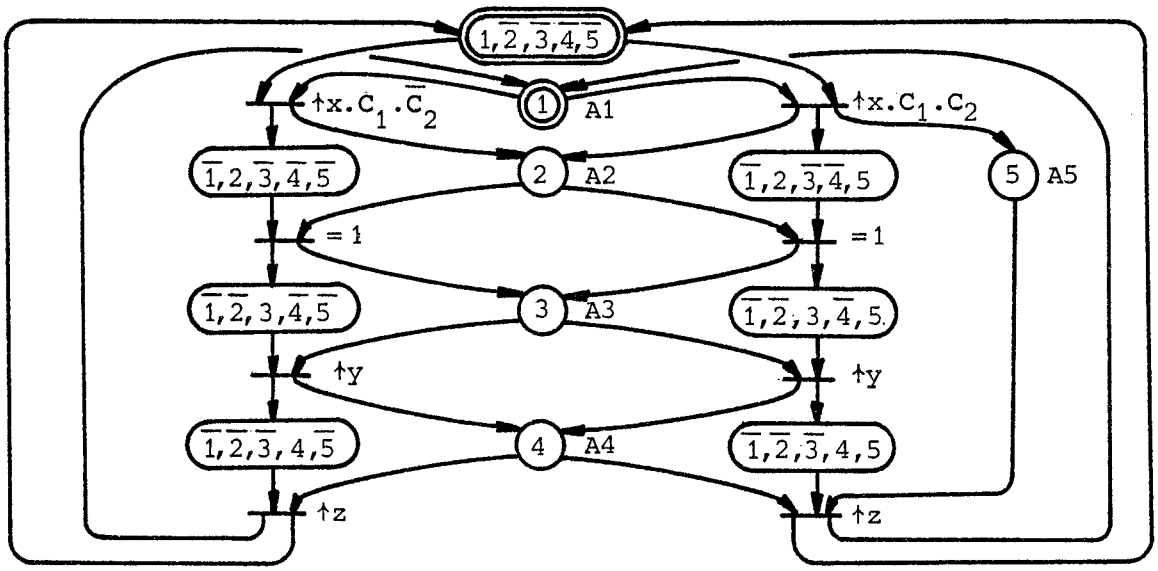


Figure 25.

étape i dans la nouvelle représentation ; on associe à cette étape la même action que celle spécifiée dans le grafcet initial. Chaque transition représentant une évolution d'une situation s à une situation s' aura :

. un arc d'entrée issu de l'étape i si i est active dans s et devient inactive dans s' ;

. un arc de sortie vers l'étape i si i est inactive dans s et devient active dans s'.

Il est aisé de vérifier que le grafcet ainsi obtenu est équivalent au grafcet initial. Dans cette nouvelle représentation, seules ont été retenues les évolutions qui changent la situation du grafcet. Les réceptivités associées aux transitions de sortie d'une même étape représentant une situation s sont, par construction, incompatibles. Par conséquent, tout tir sur occurrence d'événement se réduit au franchissement d'une seule transition et aucune étape active n'est réactivée par suite de ce franchissement.

Remarque :

Il est possible sur cette nouvelle représentation de détecter les événements de nouvelle activation d'une étape et de désactivation effective d'une étape. Une nouvelle activation d'une étape i est détectée au passage d'une situation $s = (...,\overline{i},...)$ à une situation $s' = (...,i,...)$. Inversement, une désactivation effective est détectée au passage d'une situation $s' = (...,i,...)$ à une situation $s'' = (...,\overline{i},...)$. On peut ainsi gérer des compteurs de temps T_i^t qui doivent être réinitialisés à chaque nouvelle activation de l'étape i.

Proposition 9 :

Si l'on fait abstraction de la nature des actions que l'on peut associer aux étapes/places et de la manière dont le facteur temps est pris en compte, alors le grafcet a exactement la même puissance de description que les RdPI saufs et persistants sous l'environnement le plus strict.

Démonstration :

D'après la proposition 8, on peut toujours passer d'une représentation quelconque en grafcet à une représentation grafcet équivalente

qui est sans fusionnement ni partage de marques et sans variables φ. Si l'on fait donc abstraction de la nature des actions associées aux étapes et de la manière dont le facteur temps est pris en compte, cette nouvelle représentation a exactement le même fonctionnement qu'un RdPI sauf et persistant sous l'environnement le plus strict.

Inversement, si l'on fait toujours abstraction de la manière dont le facteur temps est pris en compte, tout RdPI sauf et persistant sous l'environnement le plus strict a le même fonctionnement qu'un grafcet.

Ces deux propositions montrent que, à elles seules, les facilités apportées par le grafcet concernant la représentation des situations et des évolutions entre situations ne constituent pas (du strict point de vue de la puissance théorique de description) une différence essentielle par rapport aux RdPI.

Il est évident que l'on maîtrise mieux une description qui ne fait pas intervenir le fusionnement et le partage des marques et dans laquelle le parallélisme et la synchronisation sont exprimés de façon structurale. En revanche, une représentation qui devient trop touffue est inutilisable.

Les différences concernant l'interprétation des actions et la manière dont le facteur temps est pris en compte se justifient par les orientations différentes des deux outils.

II - COMPARAISON SELON LE POINT DE VUE DES POSSIBILITES D'ANALYSE ET DE VERIFICATION

L'état du savoir faire actuel donne de ce point de vue un avantage certain aux RdPI, même si les résultats dont on dispose pour l'analyse et la vérification des RdP non-autonomes sont encore assez faibles. Cet avantage tient principalement à l'indivisibilité et au cumul des marques dans les places, ce qui rend possible, à l'aide uniquement de la structure du réseau, d'imposer ou de reconnaître a posteriori des propriétés invariantes du système décrit par un RdPI ; chose que le grafcet ne permet pas à cause du fusionnement et du partage des marques.

L'indivisibilité et le cumul des marques dans un RdPI sont, tout d'abord, deux facteurs importants d'éclairage dans la "chasse" aux ambiguïtés et aux erreurs :

- Par exemple, dans un RdPI sauf, une place ayant deux ou plusieurs transitions de sortie représente nécessairement un OU-exclusif. Une vérification à faire systématiquement, si l'on veut que le RdPI soit persistant, consiste à s'assurer que le marquage du réseau et/ou l'interprétation qui lui est superposée sont tels que ces transitions ne sont jamais franchissables simultanément. Une telle vérification n'a pas de sens dans le grafcet du fait qu'une étape ayant plusieurs transitions de sortie peut représenter, en fonction de l'interprétation, aussi bien un OU- exclusif qu'un OU-non toujours exclusif ou un ET impératif ; une erreur de description peut ainsi passer inaperçue.

Souvent en pratique, lors de l'ébauche d'une description, on représente systématiquement une ressource non partageable par une place ayant une seule marque. L'indivisibilité de la marque garantit l'exclusion mutuelle dans l'accès à la ressource et permet de ne pas se soucier dans l'immédiat de la résolution de la non persistance en cas de conflit possible entre les transitions de sortie de cette place.

- Le cumul des marques dans les places assure que les actions associées à chaque place sont effectuées autant de fois que le fonctionnement normal du système décrit l'exige ; chaque arrivée d'une marque dans une place engage une exécution des actions associées à la place. Dans le cas où l'on se restreint à n'utiliser que des RdPI saufs, l'analyse de l'éventualité d'avoir simultanément deux marques dans une place constitue un moyen de vérification supplémentaire. Ces vérifications ne sont pas possibles dans un grafcet puisque la réactivation d'une étape déjà active est admise et est sans effet.

- D'une façon plus générale, on peut calculer les composantes consistantes et les composantes conservatives du réseau. Si le réseau n'est pas lui-même une composante consistante, c'est qu'il est ou non borné ou non vivant, et ce quel que soit l'interprétation qu'on lui associe. Les composantes conservatives permettent de vérifier des propriétés invariantes du système décrit.

La possibilité de systématiser ces vérifications lors d'une simulation du fonctionnement d'un RdPI est un atout appréciable dont la vérification par simulation d'un grafcet ne bénéficie pas.

Outre leur intérêt pour la validation de la conception, ces vérifications peuvent aussi être utilisées pour le contrôle en ligne lors de l'exploitation effective d'un système réalisé par implantation directe d'un RdPI sauf. Toute erreur pouvant résulter d'une panne du matériel ou d'un mauvais comportement de l'environnement, et qui a pour effet soit de rendre le RdPI non sauf soit de créer des situations de conflit ou de blocage, peut ainsi être détectée et traitée en temps réel. Les méthodes d'implantation de redondance que nous avons rappelées dans le chapitre précédent sont très utiles à ces fins.

III - CONCLUSION

Les facilités de représentation autorisées par le grafcet ont pour but de donner à un spécificateur d'automatisme le moyen d'exprimer le plus directement et le plus simplement possible ses intentions. Dans la mesure du possible, le spécificateur cherchera à rester proche d'une représentation du type RdPI qui met clairement en évidence le parallélisme et la synchronisation, et il ne fera recours aux facilités du grafcet que si la représentation devient illisible et peu communicable. Il faut admettre que devoir utiliser des conventions trop rigides pour exprimer des intentions qui sont encore informelles (que l'on cherche précisément à éclaircir en les mettant noir sur blanc) est une tâche difficile, et qu'à force de vouloir adapter ces intentions aux conventions de représentation, on risque finalement de les déformer et d'y introduire des erreurs.

Que la représentation obtenue soit facilement analysable est un souci secondaire pour le spécificateur. Ce qui lui importe avant tout est de pouvoir décrire le plus fidèlement possible le fonctionnement qu'il attend de l'automatisme.

Le problème de l'analyse se pose surtout au concepteur. Celui-ci a besoin de faire une analyse préliminaire des spécifications du cahier des charges afin de les comprendre. Il a ensuite besoin de faire une analyse complète des solutions de réalisation qu'il propose afin de vérifier que ces solutions sont correctement construites. Les deux difficultés majeures qui se posent au spécificateur ne se posent pas au concepteur :

- *La difficulté due à la rigidité de l'outil de description* :
le concepteur, de par son métier, possède une expérience de manipulation des outils de description que n'a pas forcément le spécificateur. Cette expérience lui apprend, au fur et à mesure, des recettes qui permettent une description structurée et claire des problèmes et de leurs solutions tout en préservant au maximum les possibilités de vérification.

- *La difficulté de rester neutre vis-à-vis de la conception* :
afin d'éviter les risques de surspécification, le spécificateur doit se garder d'utiliser des artifices de simplification qui n'apparaîssent pas "liées de façon naturelle" au problème à décrire. Le concepteur, par contre, a tout loisir de faire intervenir les recettes qui lui permettent d'optimiser la réalisation, en mettant de son côté le maximum de garanties pour une conception sûre.

Notons que la communicabilité des représentations ne se pose pas de la même manière au niveau des spécifications du cahier des charges qu'au niveau des spécifications de conception. Les spécifications d'un cahier des charges doivent être communicables entre des non spécialistes, alors qu'il suffit que les spécifications de conception soient communicables entre techniciens.

En définitive, il nous paraît tout à fait indiqué d'utiliser le grafcet pour la spécification du cahier des charges et les RdPI pour la description de la conception et pour la réalisation, dans la mesure où le premier s'adapte mieux à une représentation concise et simple et où les RdPI offrent plus de possibilités pour la vérification. Les deux modèles sont assez voisins pour que l'on puisse passer facilement de l'un à l'autre.

- § -

BIBLIOGRAPHIE (*Partie II*)

[AND.81] ANDRE C. : Systèmes à évolutions parallèles : Modélisation par réseaux de Petri à capacité et analyse par abstraction, Thèse ès-Sciences, Univ. de NICE, Février 1981.

[AGE.73] AGERWALA T., FLYNN M. : Comments on capabilities, limitations and correctness of Petri nets, First Annual Symp. on Comp. Architecture, TAMPA, Floride, 1973, pp. 81-86.

[AZE.77] AZEMA P., DIAZ M. : Checking experiments for concurrent systems, Proc. FTCS-7, p. 206, Juin 1977.

[BER.76] BERTHELOT G., ROUCAIROL G. : Reduction of Petri nets, Math. Found. of Comp. Sci., GDANSK, Pologne, Septembre 1976.

[BER.78] BERTHELOT G. : Vérification des réseaux de Petri, Thèse 3e cycle, Université de PARIS VI, Janvier 1978.

[BER.79] BERTHOMIEU B. : Analyse structurelle des réseaux de PETRI : Méthodes et outils, Thèse Doc. Ingénieur, Université Paul Sabatier, TOULOUSE, Septembre 1979.

[BES.75] BEST E., SCHMID H.A. : Systems of open paths in Petri nets, Notes in Comp. Sci., n° 32, Springer Verlag, BERLIN, 1975.

[BLA.79] BLANCHARD M. : Automatismes logiques : Grafcet ou réseaux de PETRI ?, Le Nouvel Automatisme, Mai 1979.

[BYR.75] BYRN W.H. : Sequential processes, deadlocks and semaphore primitives, TR 7-75, HARVARD University, 1975.

[CHE.79] CHEZALVIEL-PRADIN B. : Un outil graphique interactif pour la vérification des systèmes à évolution parallèle décrits par réseaux de PETRI, Thèse Doc. Ingénieur, Université Paul Sabatier, TOULOUSE, Décembre 1979.

[COU.77] COURTOIS P.J., GEORGES J. : On starvation prevention, RAIRO Informatique, Vol. 11, N° 2, 1977, pp. 127-141.

[COU.78] COUSOT P. : Méthodes itératives de construction et d'approxi-

mation de points fixes d'opérateurs monotones sur un treillis, analyse sémantique des programmes, Thèse ès-Sciences, USMG-INPG, GRENOBLE, Mars 1978.

[COU.80] COUSOT P., COUSOT R. : Semantic analysis of communicating sequential processes, ICALP, Lecture Notes in Comp. Science, Vol. 85, Juillet 1980.

[DJI.71] DIJKSTRA E.W. : Hierarchical ordering of sequential processes, Acta informatica, Vol. 1, 1971, pp. 115-138.

[FLO.78] FLORIN G., NATKIN S. : Evaluation des performances d'un protocole de communication à l'aide des réseaux de Petri et des processus stochastiques, Journées AFCET : Multiprocesseurs et multi-ordinateurs en temps réel, PARIS, Mai 1978.

[GHO.77] GHOSH S. : Some comments on timed Petri nets, Journées d'étude AFCET : Réseaux de Petri, PARIS, Mars 1977, pp. 151-163.

[GRA.77] Rapport final de la commission AFCET : Normalisation de la représentation du cahier des charges d'un automatisme logique, Août 1976. Publié intégralement dans la revue Automatique et Informatique Industrielles, numéros 61-62, Nov.-Déc. 1977.

[GRA.79] Document ADEPA : Le Grafcet, diagramme fonctionnel des automatismes séquentiels, Mai 1979.

[HAC.72] HACK M. : Analysis of production schemata by Petri nets, MAC TR.94, M.I.T., Février 1972.

[HAC.75] HACK M. : Decision problems for Petri nets and vector addition systems, MAC Techn. Memo. 59, M.I.T., Mars 1975.

[HAN.77] HAN Y.W., HEIMERDINGER W.L. : Transformation of a Petri net-like labelled graph to a directed graph for design error détection, Proc. FTCS-7, p. 205, Juin 1977.

[HOL.68] HOLT A.W. : Information system theory project, Applied Data Research inc., PRINCETON University, New Jersey, Septembre 1968.

[HOL.70] HOLT A.W., COMMONER F. : Events and Conditions, Record of the project MAC Conference on Concurrent Systems and Parallel Computation, A.C.M., NEW YORK, 1970, pp. 3-52.

[JAC.76] JACK L.A. : Functional faults, Workshop on distributed fault-tolerant systems, PORTO RICO, Novembre 1976.

[KAR.69] KARP R.M., MILLER R.E. : Parallel Program Schemata, JCSS, vol. 3, 1969, pp. 147-195.

[KEL.74] KELLER R.M. : Vector replacement systems : a formalism for modeling asynchronous systems, Techn. Rep. 117, Comp. Sci. Lab., PRINCETON University, Janvier 1974.

[KOS.73] KOSARAJU S.R. : Limitations of DIJKSTRA's semaphore primitives and Petri nets, Research Report 25, John Hopkins University, BALTIMORE, May 1973.

[LAU.74] LAUTENBACH K., SCHMID H.A. : Use of Petri nets for proving correctness of concurrent process systems, Proc. IFIP Congress 1974, North-Holland Publ. Co., 1974, pp. 187-191.

[LAU.E.75] LAUER P.E., CAMPBELL R.H. : Formal Semantics of a Class of High-level Primitives for Coordinating Concurrent Processes, Acta Informatica, Vol. 5, 1975, pp. 297-332.

[LAU.T.75] LAUTENBACH K. : Liveness in Petri nets, G.M.D. Int. Rep. ISF-75-02.1, BONN, Juillet 1975.

[LIE.76] LIEN Y.E. : Termination properties of generalized Petri nets, S.I.A.M. Journal Comp., Vol. 5, n° 2, Juin 1976, pp. 251-265.

[LIP.76] LIPTON R. : The reachability problems and the boundedness problem for Petri nets are exponential-space hard, TR-62, Dept. Comp. Sci., Yale University, NEW HAVEN, Conn., Janvier 1976.

[MAR.76] MARIN J., ANDRE C., BOERI F. : Conception de systèmes séquentiels totalement autotestables à partir des réseaux de Petri, Revue RAIRO-Automatique, Vol. 10, n° 11, Novembre 1976, pp. 23-40.

[MEM.77] MEMMI G. : Semiflows and Invariants. Applications in Petri nets theory, Journées d'étude AFCET : Réseaux de Petri, Mars 1977, PARIS, pp. 145-150.

[MER.76] MERLIN P.M., FARBER D.J. : Recoverability of communication protocols. Implications of a theoretical study, IEEE Trans. Comm., Vol. COM-24, Septembre 1976, pp. 1036-1043.

[MOA.3.76] MOALLA M. : L'approche fonctionnelle dans la vérification des systèmes informatiques. Proposition d'un ensemble de méthodologies, Thèse Docteur-Ingénieur, INPG-ENSIMAG, GRENOBLE, Décembre 1976.

[MOA.78] MOALLA M., PULOU J., SIFAKIS J. : Synchronized Petri Nets : A model for the description of non-autonomous systems, Mathematical Foundations of Computer Sciences, Ed. Springer Verlag, 1978, pp. 374-383. (Aussi : Réseaux de Petri Synchronisés, RAIRO-Automatique, Vol. 12, n° 2, 1978, pp. 103-130).

[MOA.2.78] MOALLA M., SIFAKIS J., SILVA M. : A la recherche d'une méthodologie de conception sûre des automatismes logiques basée sur l'utilisation des réseaux de Petri, Rapport de recherche IMAG, n° 138, Oct. 1978. (Texte reproduit dans la monographie d'informatique de l'AFCET : sûreté de fonctionnement des systèmes informatiques, Ed. Hommes et Techniques, 1980).

[MOA.2.79] MOALLA M. : Spécification de réalisation de la centrale de sécurité CALOR 40.11, Doc. OPTION, réf. 78-12-158.041-02, Mai 1979.

[MOA.3.79] MOALLA M. : Spécification de réalisation du programmateur de four CIT-ALCATEL, Doc. OPTION, réf. 79-11-046, Décembre 1979.

[PAT.70] PATIL S.S : Coordination of asynchronous events, PhD Thesis, Dept. Electrical Engineering, MIT, CAMBRIDGE, USA, Mai 1970.

[PET.62] PETRI C.A. : Kommunikation mit Automaten, Schriften des Rheinisch-Westfalischen Institutes für instrumentelle Mathematik an der Universitat Bonn, BONN, 1962 (thèse traduite en anglais dans Communication with automata, Tech. Rep. n° RADC-TR-65-337, Vol. 1, Rome Air Develop. Center, Griffis Air Force Base, NEW YORK, Janvier 1966.

[PET.72] PETERSON W.W, WELDON E.J. Jr : Error Correcting Codes, The M.I.T. Press, CAMBRIDGE, Mass., 1972.

[PET.75] PETRI C.A. : Interpretations of net theory, Interner Bericht 75-07, G.M.D., BONN, Juin 1975.

[PET.77] PETERSON J.L. : Petri Nets, ACM Comp. Surveys, Vol. 9, n° 3, Septembre 1977, pp. 223-251.

[PUL.79] PULOU J. : Expression de la synchronisation dans les systèmes informatiques et conception d'architectures tolérant les pannes, Thèse Doc. Ingénieur, INPG-ENSIMAG, GRENOBLE, Septembre 1979.

[RAM.73] RAMCHANDANI C. : Analysis of asynchronous concurrent systems by timed Petri nets, PhD. Thesis, M.I.T., Septembre 1973.

[SIF.77] SIFAKIS J. : Use of Petri nets for performance evaluation, Measuring, Modeling and Evaluating Computer Systems, North-Holland Publ. Co., 1977, pp. 75-93.

[SIF.2.77] SIFAKIS J. : Homomorphisms of Petri nets - Applications to the realization of fault tolerant systems, Rapport de recherche IMAG, n° 90, GRENOBLE, Octobre 1977.

[SIF.78] SIFAKIS J. : Structural properties of Petri nets, 7th Symp. on Math. Foundations of Computer Sci., ZAKOPANE, Pologne, Septembre 1978.

[SIF.79] SIFAKIS J. : Le contrôle des systèmes asynchrones : concepts, propriétés, analyse statique, Thèse ès-Sciences, USMG-INPG, GRENOBLE, Juin 1979.

[SZL.77] SZLANKO J. : Petri nets for proving some correctness properties of parallel programs, IFAC-IFIP Workshop on Real-time Programming, EINDHOVEN, Juin 1977.

[VAL.76] VALETTE R. : Sur la description, l'analyse et la validation des systèmes de commande parallèles, Thèse ès-Sciences, Université Paul Sabatier, TOULOUSE, Novembre 1976.

[VAL.78] VALETTE R. : Etude comparative de deux outils de représentation : Grafcet et réseau de Petri, Le Nouvel Automatisme, Décembre 1978.

PARTIE III

MISE EN OEUVRE ASSISTEE PAR ORDINATEUR

Cette troisième partie, constituée du chapitre VIII, sera consacrée essentiellement à la présentation d'un outil de vérification et de simulation fonctionnelle multiniveaux de systèmes décrits par des Réseaux de Petri Interprétés : le système MAS (Multilevel Assisted Design System).

Cet outil est au coeur même de la mise en oeuvre de la méthodologie de conception proposée dans l'introduction de ce mémoire.

L'avantage de disposer d'un simulateur fondé sur les RdPI est double :

- d'une part, le fait d'utiliser pour la simulation le même modèle que pour la conception facilite l'intervention de la simulation tout au long de la conception et non pas seulement au cours des étapes finales. Ces simulations permettent de comparer entre elles différentes solutions que ce soit du point de vue fonctionnel ou du point de vue du comportement dynamique (temps de réponse, taux d'activité des différents sous-ensembles (fonctions) du système, ...), comparaisons qui sont difficiles à faire par analyse statique.

- d'autre part, comme nous l'avons montré dans le chapitre VI, la simulation reste, dans le cas général, le seul recours possible pour la vérification des propriétés caractéristiques d'un RdPI, bien que cette vérification ne puisse être que partielle.

Tout cela implique que le simulateur ne se réduise pas à un simple algorithme gérant les évolutions d'un RdPI, mais qu'il soit assorti des facilités d'observation, de mesure et de vérification d'invariants sur ces évolutions.

Les principales idées qui ont conduit à la définition du système MAS datent de la fin de 1975 [MOA.75][MOA.76][LED.76][MOA.2.76][MOA.3.76]. Initialement, ce système a été fondé sur les Réseaux de Contrôle de Processus Parallèles (RCPP [SIF.74]) dont les règles de fonctionnement sont très proches de celles qui ont été définies plus tard pour le GRAFCET. Une première version du simulateur utilisant ce modèle a été réalisée par M. ZACHARIADES dans le cadre d'une thèse de 3ème cycle [ZAC.77]. Une deuxième version utilisant, cette fois-ci, les Réseaux de PETRI Interprétés (RdPI [MOA.3.76]) a été réalisée en collaboration avec la Sté SINTRA [CAP.78][CHA.79].

Ces deux versions réalisées de MAS ont visé davantage à en démontrer la faisabilité qu'à fournir de véritables outils industriels. Aussi, notre intention ici n'est pas de décrire de façon précise l'une ou l'autre des deux réalisations et les langages de programmation auxquels elles ont abouti, mais d'en rappeler les idées sources, mûries et enrichies, depuis, par une large expérimentation sur des applications concrètes. Nous décrirons et justifierons les concepts retenus dans le projet de définition de MAS. Nous aborderons ensuite le problème du passage de la description d'une solution de conception d'un système, telle qu'elle peut être validée par MAS, à une implantation effective sur mini ou microcalculateurs. Nous présenterons enfin un exemple d'application qui illustre quelques uns des apports fondamentaux de l'utilisation des RdPI dans l'étude et l'analyse de l'automatisation de procédés industriels.

CHAPITRE VIII

LE SYSTÈME M.A.S

I - CARACTERISTIQUES DE BASE 2

II - DESCRIPTION D'UN SYSTEME DANS LE LANGAGE MAS 3

II - 1. STRUCTURATION DE LA DESCRIPTION 3

II - 2. RAPPEL DE DEFINITION DU MODELE RdPI 5

II - 3. PRINCIPE DE LA SIMULATION D'UN RdPI 8

II - 4. DESCRIPTION DES MODULES-TYPES 11

II - 5. DESCRIPTION D'UNE CONFIGURATION DU SYSTEME 13

II - 6. INITIALISATION ET DEFINITION DES SEQUENCES D'ENTREES POUR LA SIMULATION 15

II - 7. REMARQUES, PRECISIONS ET EXTENSIONS 16

III - PASSAGE A UNE REALISATION EFFECTIVE D'UN SYSTEME VALIDÉ PAR MAS 24

IV - ETUDE DE CAS : APPLICATION DES RdPI A LA SPECIFICATION, L'ANALYSE ET LA CONCEPTION DE L'AUTOMATISATION D'UN ATELIER FLEXIBLE DE MASTICAGE DE CARROSSERIES AUTOMOBILES 25

IV - 1. PRESENTATION DE L'APPLICATION 25

IV - 2. PROBLEMES POSES PAR L'AUTOMATISATION 28

IV - 3. APPORTS ATTENDUS DES OUTILS CAO 30

IV - 4. UTILISATION DES RdPI 30

V - CONCLUSION 38

I - CARACTERISTIQUES DE BASE

Structuration et hiérarchisation de la démarche de conception, associées à l'utilisation des RdPI pour la description, l'analyse et la validation des solutions de conception, sont les idées clés sur lesquelles nous avons fondé l'approche proposée dans l'introduction de ce mémoire.

Au niveau du simulateur MAS, ces idées se traduisent par les caractéristiques de base suivantes :

- la modularité : A chacun des niveaux de la conception, le système étudié est vu comme un ensemble de modules qui interagissent. Il est important que la programmation pour la simulation puisse maintenir apparente cette structure en termes de modules et d'interconnexions entre modules

- la description séparée de la partie commande et de la partie opérative de chaque module : Cette séparation résulte directement de l'utilisation du modèle RdPI. La partie commande est représentée par le RdP qui décrit les situations (états) du système et les possibilités d'évolution entre situations, en prenant en compte les temporisations et les réceptivités qui conditionnent et synchronisent ces évolutions. La partie opérative comprend les variables du système et les opérateurs agissant sur ces variables. Une description qui sépare les deux parties est nettement plus lisible et se prête plus facilement à la mise au point

- la possibilité de donner des descriptions à plusieurs niveaux de détails (multiniveaux) si l'on veut que le même outil puisse être utilisé aux différents stades de la conception

- l'implantation de procédures qui permettent de vérifier, au cours de la simulation, les propriétés de bon fonctionnement de chaque module et la coordination de l'ensemble pour la réalisation des fonctions globales du système étudié

- la facilité de mise en oeuvre, qualité essentielle pour un outil d'aide à la conception.

Nous allons montrer comment ces caractéristiques sont approchées dans MAS. Comme nous l'avons annoncé, notre intention est uniquement de décrire les concepts adoptés et de donner une idée sur ce que peuvent être les primitives d'un langage de programmation permettant d'exprimer ces concepts.

La définition précise de la syntaxe et de la sémantique des primitives d'un tel langage doit faire l'objet d'une étude approfondie des solutions techniques de réalisation, de l'étendue des domaines d'application visés et des pratiques de mise en oeuvre.

II - DESCRIPTION D'UN SYSTEME DANS LE LANGAGE MAS

II - 1. STRUCTURATION DE LA DESCRIPTION

Les possibilités d'une description modulaire sont conçues dans MAS de façon à répondre à trois objectifs :

1 - A un niveau haut de description, un système complexe peut être décrit par un seul RdPI de taille importante. La transcription de ce RdPI dans le programme de simulation peut être mieux maîtrisée en la donnant par parties, chaque partie constituant un module. L'interconnexion entre modules s'effectue par superposition de places communes (figure 1).

2 - A un niveau plus détaillé, le système peut être défini explicitement comme étant réalisé à partir de l'interconnexion de modules, chaque module étant décrit par un RdPI. Dans ce cas, la transcription dans le programme de simulation doit pouvoir maintenir nettement apparente cette structure en termes de modules et de liens entre modules permettant l'échange de valeurs de variables (figure 2).

3 - Dans les deux cas, plusieurs modules identiques peuvent être utilisés pour constituer le système étudié. La description, éventuellement paramétrée, de chaque module-type doit alors pouvoir être donnée une seule fois. Le simulateur prendra en charge de répliquer ces modules-types autant de fois qu'il est nécessaire pour composer le système étudié, selon les indications qui lui sont fournies.

La structure d'un programme de description d'un système à l'entrée de MAS est alors celle illustrée par la figure 3. On distingue trois sections principales :

α) Définition des modules-types du système. La description de chaque module-type est délimitée par les mots-clés DCLMODTYPE et FINMOD.

β) Création des modules d'une configuration du système (par replication à partir des modules-types) et interconnexion de ces modules. Cette section est délimitée par les mots-clés DCLCONF et FINCONF.

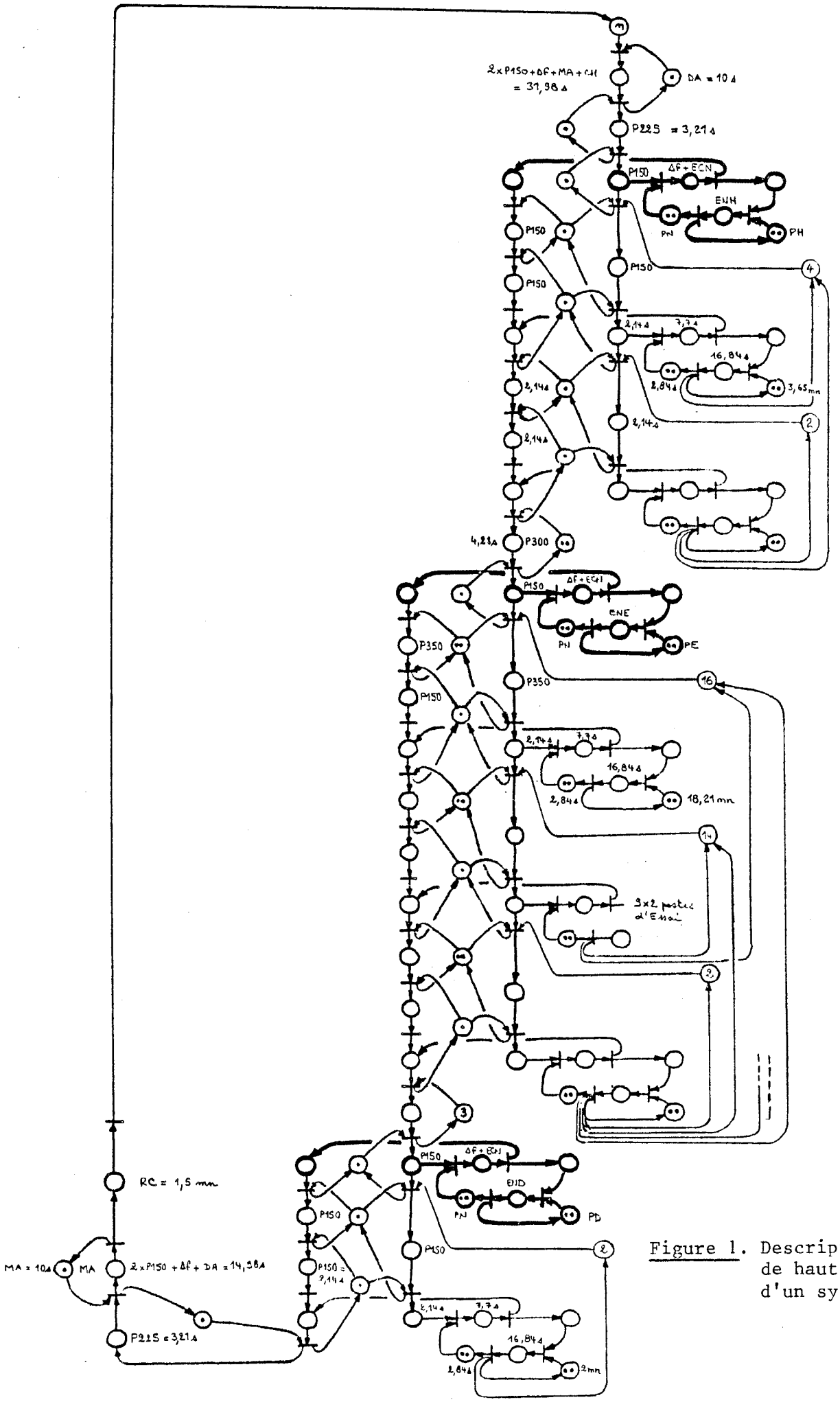


Figure 1. Description de haut niveau d'un système

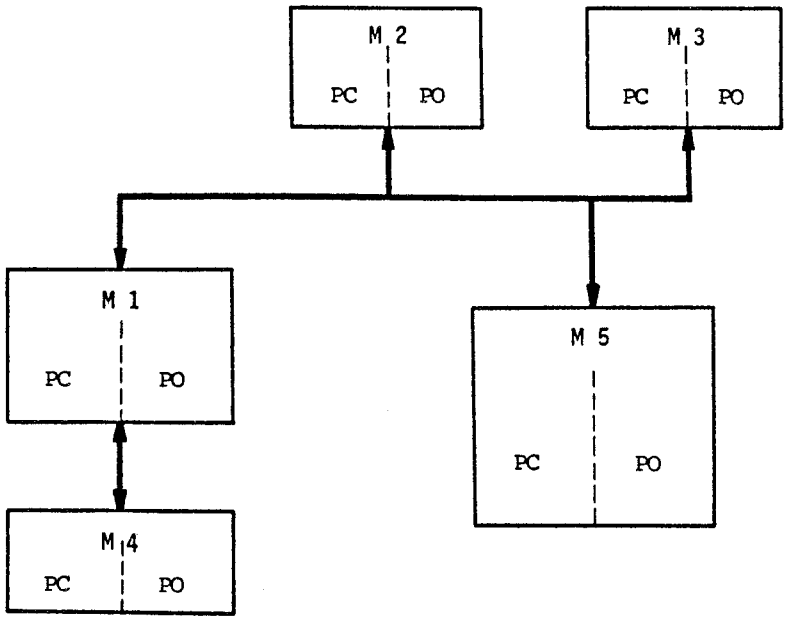


Figure 2. Description modulaire

γ) Initialisation des variables du système et spécification des séquences d'occurrences d'événements externes pour lesquelles on veut étudier son comportement. Cette section est délimitée par les mots-clés INIT et FINPROG.

En tête du programme, on peut trouver une déclaration des paramètres qui interviennent dans la définition des modules-types.

Nous allons détailler la description de chacune de ces sections. Auparavant, il nous faut rappeler la définition du modèle RdPI et introduire la manière dont le fonctionnement d'un RdPI est simulé dans MAS.

II - 2. RAPPEL DE DEFINITION DU MODELE RdPI (Chap. V, § VI - 5)

Un RdPI est défini par la donnée :

- d'un sous-système opératif (V,OP,C) tel que :
  - $V = \{v_1, v_2, .. , v_m\}$ est un ensemble fini de variables prenant leurs valeurs respectivement dans les domaines $D_1, D_2, \ldots, D_m$;

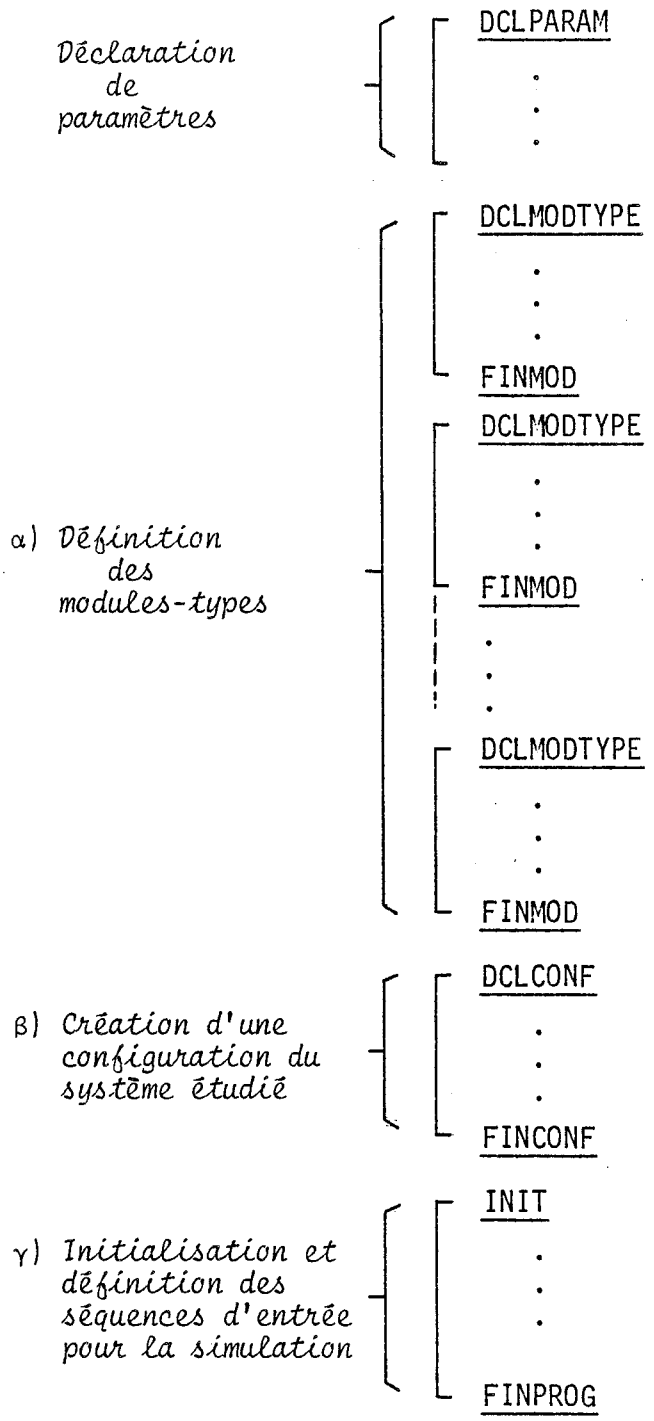


Figure 3. Structure d'un programme de description d'un système à l'entrée de MAS

. OP = $\{op_1, op_2, \ldots op_n\}$ est un ensemble fini d'opérateurs définis comme des applications internes de $D_1 \times D_2 \times \ldots \times D_m$;

. C = $\{c_1, c_2, \ldots c_r\}$ est un ensemble de conditions (prédicats) sur les variables de V ;

- d'un RdPTS (R, E, μ, ν) ;
- d'une application $\varphi : P \rightarrow OP$, P étant l'ensemble des places de R ;

et - d'une application $\psi : T \rightarrow C$, T étant l'ensemble des transitions de R.

Les conventions de représentation graphique d'un RdPI sont rappelées dans la figure 4.

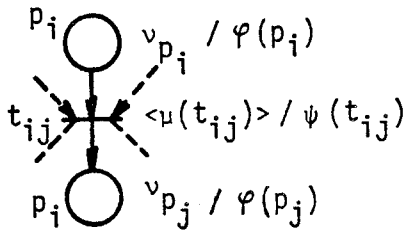


Figure 4. Représentation d'un RdPI (Rappel)

L'expérimentation de ce modèle sur des cas pratiques a révélé l'intérêt de deux extensions :

1. La possibilité d'utiliser, dans l'expression des conditions associées aux transitions, des variables représentant des places. A chaque place, on associe une variable de même nom, dont la valeur à tout instant est égale au nombre de marques à l'état disponible dans la place.

Cette extension généralise les notions de test à "0" et de test à "1" que nous avons évoquées dans le chapitre précédent. A elle seule, cette extension ne modifie en rien la puissance théorique de description du modèle (nous l'avons démontré pour le test à "0" et le test à "1" - chap. VII, § I - 4). Elle a pour but simplement d'apporter des facilités pour une description plus directe et plus claire.

Nous verrons, par la suite, sous quelles conditions l'utilisation de variables représentant les places est autorisée de façon plus générale dans MAS.

2. La possibilité d'utiliser des prédicats temporels. Un prédicat temporel se comporte comme une variable logique dont la mise à la valeur "faux" est effectuée à partir des actions (opérateurs) associées aux places du RdPI. On indique, à chaque mise à la valeur "faux", une durée limite au bout de laquelle la variable (le prédicat) doit reprendre systématiquement la valeur "vrai".

Cette extension est particulièrement intéressante pour simuler des échéances limites ("chiens de garde"), chose que le modèle initial ne permet pas de faire.

II - 3. PRINCIPE DE LA SIMULATION D'UN RdPI

La simulation d'un RdPI comporte :

- une phase A d'initialisation exécutée une fois au début de la simulation, et
- une itération sur une phase B de traitement des échéances associées aux instants d'un échéancier de simulation (Chap. V, § VI - 5).

A - Phase d'initialisation :

L'initialisation porte à la fois sur le marquage initial du réseau et sur les valeurs des variables du sous-système opératif. Par défaut, les places sont initialisées avec le marquage vide et les variables du sous-système opératif sont initialisées avec la valeur "indéfinie".

On vérifie que, compte tenu de ces valeurs de variables, le marquage initial défini est stable. Cela implique, en particulier, que l'on puisse évaluer toutes les conditions associées aux transitions qui sont validées par ce marquage et qui sont réceptives à l'événement e ; autrement dit, le fait que certaines des variables de ces conditions puissent se trouver avec la valeur indéfinie ne doit pas empêcher d'associer à ces conditions des valeurs définies. (Pour que la simulation puisse avoir un sens, la condition de franchissement d'une transition validée doit pouvoir être évaluable à tous les instants d'occurrence de l'événement associé à la transition. La détection de la non-vérification de cette règle conduit à l'abandon définitif de la simulation).

Cette phase d'initialisation s'achève par l'établissement d'un échéancier de simulation dans lequel sont notés, dans l'ordre, les instants associés aux occurrences des événements externes pour lesquelles

on veut étudier le comportement décrit par le RdPI.

La simulation se poursuit en B en considérant le premier instant noté dans l'échéancier.

B - Phase de traitement des échéances associées à un même instant :

D'une façon générale au cours de la simulation, on peut trouver, associées à un même instant τ de l'échéancier, l'échéance du passage d'une (ou de plusieurs) marque(s) de l'état indisponible à l'état disponible et/ou l'échéance d'une occurrence simultanée d'événements externes.

On effectue, dans l'ordre, les traitements suivants :

B1) Pour chaque marque qui doit passer de l'état indisponible à l'état disponible dans une place p, on effectue sur les valeurs courantes des variables du sous-système opératif la transformation définie par l'opérateur $\varphi(p)$. Ensuite, on fait passer la marque à l'état disponible.

B2) Ayant effectué le passage à l'état disponible de toutes les marques concernées, on réévalue, à partir des nouvelles valeurs des variables du sous-système opératif, l'ensemble des conditions associées aux transitions. Si le nouveau marquage du réseau n'est pas stable, on effectue une itération de tirs sur occurrence de $\{e\}$ à l'instant τ jusqu'à stabilisation. Au cours de cette itération de tirs, chaque fois qu'une marque indisponible est déposée dans une place p, l'échéance $\nu_p(\tau)$ du passage à l'état disponible est notée dans l'échéancier.

B3) Ce n'est qu'alors que l'on considère l'ensemble X des événements externes ayant une occurrence à l'instant τ. On effectue à partir du marquage présent un tir itéré sur occurrence de X à l'instant τ. Au cours de ce tir itéré, chaque fois qu'une marque indisponible est déposée dans une place p, l'échéance $\nu_p(\tau)$ du passage à l'état disponible est notée dans l'échéancier.

La simulation reprend de nouveau en B en considérant le prochain instant de l'échéancier.

L'un des points les plus importants dans la réalisation de cet algorithme concerne la simulation des événements.

Vu de l'intérieur d'un module, un événement externe, qu'il soit

engendré par l'environnement extérieur au système ou par l'un quelconque des autres modules qui lui sont connectés, correspond toujours à un changement de valeur de l'une de ses variables d'interface. La simulation de la prise en compte des occurrences d'événements s'effectue de la façon suivante :

. A chaque variable d'interface VES on associe une variable interne $VES dans laquelle on mémorise en permanence la dernière valeur affectée à la variable VES. Ainsi, à chaque nouvel examen de cette variable, on peut détecter, en comparant VES et $VES, si un changement de valeur a eu lieu.

. D'un autre côté, à chaque événement différent de e qui est utilisé dans les réceptivités du RdPI décrivant le module, on associe une variable logique du type EVENEMENT. Ces variables sont déclarées explicitement comme telles en tête de la description de la partie de commande du module. On spécifie également pour chacun de ces événements l'expression qui permet de le calculer à partir des valeurs des variables VES et $VES
Exemple : si la variable VES est du type BINAIRE et si l'événement EVPVES représente le passage de cette variable de la valeur "0" à la valeur "1", on écrira :

EVPVES := ¬ $VES.VES

Si la variable VES est une variable du type REEL et si l'événement EV20VES représente le passage de cette variable d'une valeur inférieure à 20 à une valeur supérieure ou égale à 20, on écrira :

EV20VES := ($VES < 20)(VES >= 20)

On peut remarquer que, procédant de ce principe, il est possible de définir à partir d'événements élémentaires toutes sortes d'événements composés.

. Du point de vue de la simulation, les réceptivités associées aux transitions se transforment ainsi en des conditions (les événements étant remplacés dans l'expression des réceptivités par des variables logiques). L'évaluation des variables du type EVENEMENT est effectuée au début de la sous-phase B3 de chaque phase B de la simulation. Ces variables sont remises à zéro systématiquement dès la fin du premier tir effectué dans la sous-phase B3.

II - 4. DESCRIPTION DES MODULES-TYPES

Le schéma de la figure 5 donne un exemple de description d'un module-type. Cette description comporte :

α1) La déclaration des variables d'interface (entrées/sorties) du module, annoncée par le mot-clé VARES.

En fonction du type de lien établi, les échanges entre modules peuvent se faire :

. par communication de valeurs de variables, ces variables pouvant être du type ENTIER, REEL, BINAIRE, ..., simples ou vectorielles ;

. par communication de marques dans des places communes, les variables d'interface correspondantes sont alors déclarées du type PLACE.

α2) La déclaration des variables internes au module, annoncée par le mot-clé VARINT. Ces variables peuvent également être du type ENTIER, REEL, BINAIRE, ..., simples ou matricielles. Elles peuvent éventuellement être initialisées.

Les variables représentant des prédicats temporels sont déclarées sous le type COMPT. Afin de les distinguer visuellement des autres variables, on peut décider que le premier caractère de leur identification soit toujours le caractère "!".

α3) La description de la partie opérative : les opérateurs associés aux places du RdP représentant la partie commande sont décrits dans la partie opérative sous la forme de procédures agissant tant sur les variables internes au module que sur ses variables d'interface. L'exécution de la transformation définie par un opérateur se traduit ainsi par l'exécution non-interruptible de la procédure qui le décrit.

La description de chaque procédure est délimitée par les mots-clés DCLPROC et FIN. Dans la description du corps de la procédure on peut utiliser, en lecture seulement, une variable HORLOGE qui indique en permanence la valeur courante du temps simulé. Lors de l'exécution de la procédure, la valeur de la variable HORLOGE correspond à l'instant où la procédure a été activée ; cette valeur est maintenue constante le long de l'exécution de la procédure. La primitive INIT permet d'initialiser des prédicats temporels. Dans l'exemple donné, INIT (!CT1/20) commande l'initialisation du prédicat temporel !CT1 pour une temporisation de

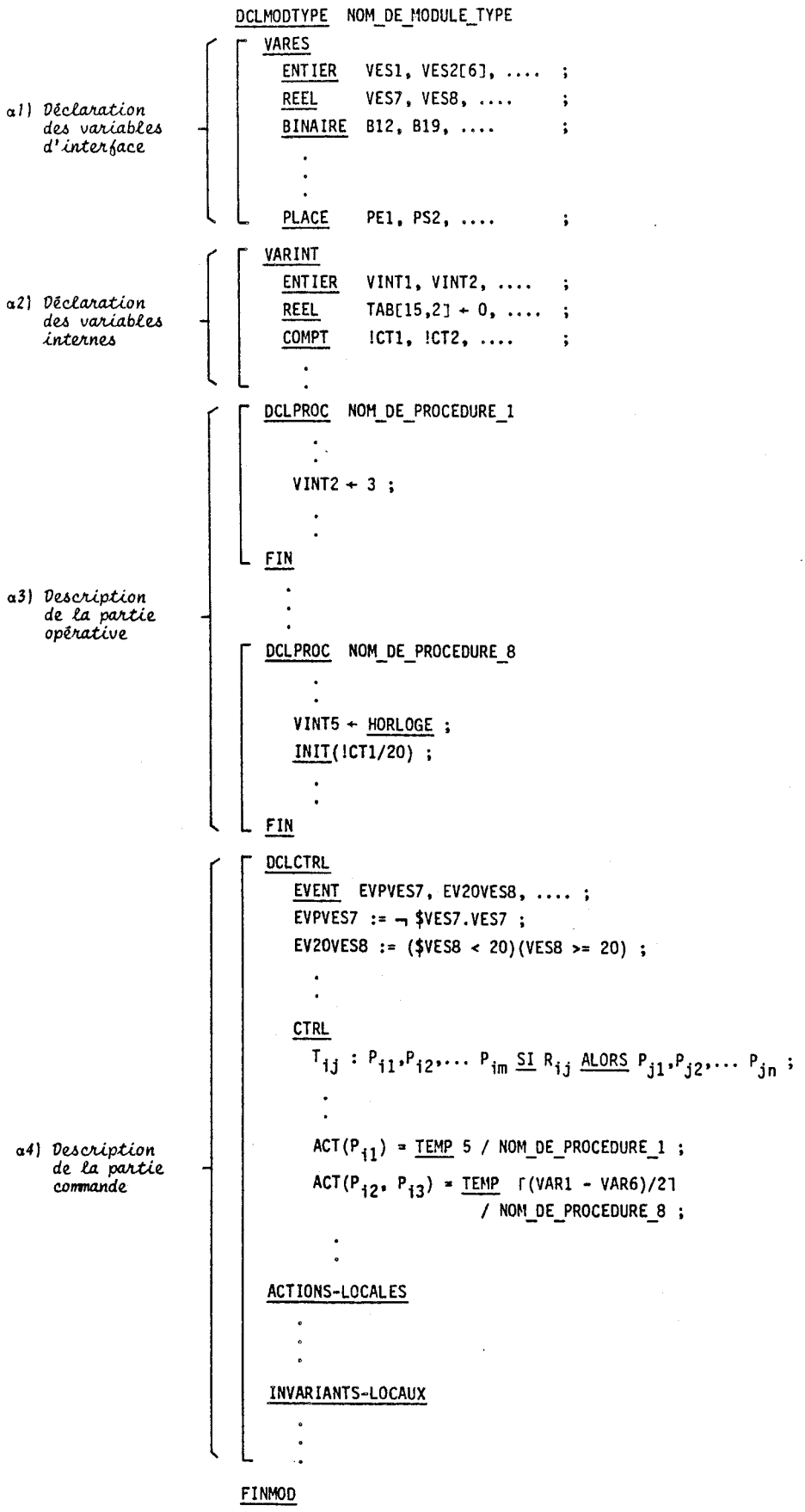


Figure 5. Exemple de description d'un module-type

20 unités en temps simulé, en comptant à partir de la valeur indiquée par HORLOGE.

Nous reviendrons, dans la suite, sur le choix du langage que l'on peut utiliser de façon plus générale pour la description des corps de procédures.

α4) La description de la partie commande, annoncée par le mot-clé DCLCTRL :

- En tête de cette description, on trouve la déclaration des variables du type EVENEMENT (mot-clé EVENT) et la spécification des expressions qui permettent d'évaluer ces variables.

- Le mot-clé CTRL annonce la description de la partie commande proprement dite, c'est-à-dire la structure du RdPI représentant les états du module ainsi que les temporisations et les réceptivités qui régissent les évolutions entre ces états. Les instructions utilisées sont de deux types :

. l'instruction

$$T_{ij} : P_{i1},P_{i2},\cdots P_{im} \ \underline{SI}\ R_{ij}\ \underline{ALORS}\ P_{j1},P_{j2},\cdots P_{jn}$$

spécifie que le RdPI comprend la transition T_{ij} à laquelle est associée la réceptivité R_{ij} et dont les places d'entrée sont $P_{i1},P_{i2},\cdots P_{im}$ et les places de sortie $P_{j1},P_{j2},\cdots P_{jn}$.

. l'instruction

$$\underline{ACT}(P_i) = \underline{TEMP}\ \nu_{P_i} \ /\ NOM\_DE\_PROCEDURE$$

spécifie qu'à la place P_i sont associés la fonction ν_{P_i} qui détermine les temporisations du passage des marques dans cette place (la fonction ν_{P_i} peut être constante ou définie à l'aide d'une expression quelconque à valeurs dans l'ensemble des réels) et l'opérateur $\varphi(P_i)$ décrit par la procédure NOM_DE_PROCEDURE.

La description de la partie commande comprend d'autres éléments (ACTIONS_LOCALES et INVARIANTS_LOCAUX) que nous détaillerons par la suite.

II - 5. DESCRIPTION D'UNE CONFIGURATION DU SYSTEME

Il s'agit, dans la deuxième section du programme délimitée par les mots-clés DCLCONF et FINCONF, de fournir au simulateur les indications qui

lui permettent de constituer la configuration du système étudié. Un exemple est donné dans la figure 6.

```
DCLCONF
  CREE
  MODULE1, MODULE2 := MODELE ;
     .
     .
     .
  LIE
  MODULE1.VES11 = MODULE2.VES21 ;
  MODULE1(VES12, VES13) = MODULE3(VES33, VES35) ;
     .
     .
     .
  ACTIONS-GLOBALES
     .
     .
     .
  INVARIANTS-GLOBAUX
     .
     .
     .
FINCONF
```

Figure 6. Description de la configuration du système étudié

On y distingue 4 sous-sections :

β1) Dans la première sous-section, annoncée par le mot-clé CREE, on spécifie les modules à créer à partir des modules-types prédéfinis. Dans l'exemple donné, deux modules MODULE1 et MODULE2 sont créés à partir du module-type MODELE.

β2) Dans la deuxième sous-section, annoncée par le mot-clé LIE, on établit les liens entre les variables d'interface de ces modules. Du point de vue de la simulation, la connexion de deux modules MODULE1 et MODULE2 par l'intermédiaire des variables VES11 du module MODULE1 et VES21 du module MODULE2 consiste simplement à confondre ces variables. Pour désigner ces variables, on peut utiliser la notation "génitive" (exemple :

MODULE1.VES11 désigne la variable VES11 du module MODULE1) ou la notation "parenthésée" (exemple : MODULE1(VES12, VES13) désigne les variables VES12 et VES13 du module MODULE1). Il est bien évident que les variables d'interface respectives qui lient deux modules doivent être de types compatibles : places avec places, variables binaires avec variables binaires, etc.

β3 et β4) Les deux dernières sous-sections concernent respectivement la description d'actions globales au niveau du système et la description d'invariants globaux que la simulation doit contrôler en permanence. Elles font partie des extensions dont nous justifierons l'intérêt par la suite.

II - 6. INITIALISATION ET DEFINITION DES SEQUENCES D'ENTREES POUR LA SIMULATION

Cette troisième et dernière section du programme, délimitée par les mots-clés INIT et FINPROG, sert à spécifier les valeurs initiales de certaines variables du système (d'autres initialisations peuvent être spécifiées dans la description même des modules), puis à décrire les séquences d'occurrences d'événements externes pour lesquelles on veut effectuer la simulation. Un exemple est donné en figure 7 :

- Le mot-clé INIT annonce une série d'initialisations pouvant porter aussi bien sur des variables d'interface que sur des variables internes, quels qu'en soit le type, y compris les variables du type PLACE (définition du marquage initial).

- Le mot-clé DONNEES annonce la description de la séquence des entrées. Chaque élément de la séquence est présenté sous la forme suivante :

τ_i : MODULE1.VAR11 := Valeur11, MODULE2(P21,VAR26) := (Valeur21, valeur26), ... ;

ce qui signifie qu'à l'instant τ_i, la variable VAR11 du module MODULE1 et les variables P21 et VAR26 du module MODULE2 ... prendront respectivement les valeurs Valeur11, Valeur21 et Valeur26 ... . En principe, ces variables doivent avoir été déclarées en tant que variables d'interface. Cependant, il s'avère utile d'autoriser l'accès direct à des variables internes, éventuellement même du type PLACE, afin de pouvoir simuler des effets de pannes.

On peut spécifier des entrées périodiques. Exemple :

τ_k!Δ : MODULE1.VAR5 := Valeur5 ;

ceci signifie que la variable VAR5 du module MODULE1 sera initialisée à la valeur Valeur5 à tous les instants $\tau_k + m\Delta$, pour m : 0,1,2,...

L'échéance τ_s placée devant la directive STOP indique l'instant où la simulation doit être arrêtée.

```
INIT
   MODULE1.VES1 := Valeur1 ;
   MODULE2(VES21, PE2, VINT20) := (Valeur3, Valeur4, Valeur5) ;
      .
      .
      .
      .
DONNEES
   τi : MODULE1.VAR11 := Valeur11 ,
        MODULE2(P21,VAR26) := (Valeur21, Valeur26), .....  ;
   τj : ......       ;
        .
        .
        .
        .
   τk!Δ  : MODULE1.VAR5 := Valeur5 ;
        .
        .
        .
   τs STOP
        .
        .
        .
FIN
```

Figure 7. Initialisation et définition des séquences d'entrée pour la simulation

II - 7. REMARQUES, PRECISIONS ET EXTENSIONS

1. Choix du langage pour la description des corps de procédures dans les parties opératives des modules :

Ce choix est important puisqu'il détermine directement les possibilités de description multiniveaux.

Pour un module donné, le passage d'une description d'un niveau de détail à une description d'un niveau plus fin peut s'effectuer en

détaillant les états de la partie commande et/ou en détaillant la description des opérateurs associés aux places représentant ces états. Si le langage de description de la partie commande reste le même (les mêmes primitives permettent de décrire la structure d'un RdP quelle que soit sa taille), le langage de description des opérateurs doit pouvoir s'adapter au niveau où l'on veut se situer, et ce en offrant une gamme assez large de primitives qui permettent de donner la description de ces opérateurs de la façon la plus directe et la plus concise possible.

Le langage APL[IVE.72] est certainement l'un des meilleurs exemples que l'on puisse citer, étant donnée la grande richesse de ses primitives. C'est d'ailleurs un sous-ensemble du langage APL qui a été utilisé dans la première réalisation de MAS [ZAC.77].

La description d'un système n'a pas besoin d'être homogène quant à son niveau de détail. Le langage choisi doit permettre que les descriptions des modules du système, ainsi que les descriptions des opérateurs d'un même module, puissent être données à des niveaux de détail différents.

2. Description des parties commande :

2.1. Nous avons suggéré deux types d'instructions pour la transcription d'un RdPI :

[1] T_{ij} : $P_{i1}, P_{i2}, \ldots P_{im}$ SI R_{ij} ALORS $P_{j1}, P_{j2}, \ldots P_{jn}$;

[2] ACT(P_i) : TEMP ν_{P_i} / NOM_DE_PROCEDURE ;

L'instruction [1] se simplifie dans le cas où l'ensemble des places d'entrée de T_{ij} (resp. l'ensemble des places de sortie) est vide. On peut utiliser une notation spécifique pour indiquer quand l'un ou l'autre de ces deux ensembles est vide. On peut aussi convenir que l'absence d'identificateurs entre T_{ij} : et SI (resp. entre ALORS et le délimiteur ";") signifie que l'ensemble des places d'entrée (resp. l'ensemble des places de sortie) est vide.

La même remarque peut être faite concernant le champ NOM_DE_PROCEDURE de l'instruction [2] dans le cas où l'opérateur associé à la place est l'opérateur identité (action nulle).

2.2. Rappelons que pour qu'une marque arrivant dans une place P_i puisse conduire à l'activation de l'opérateur associé à cette place, il

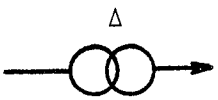
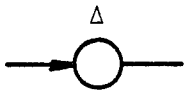
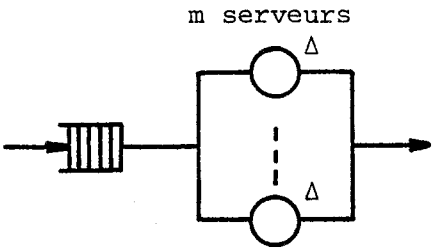
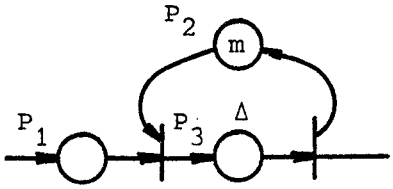
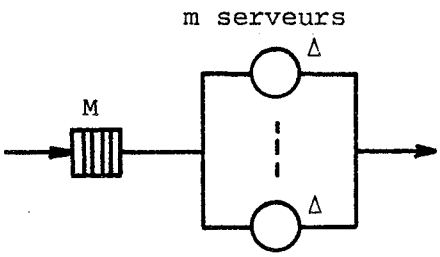
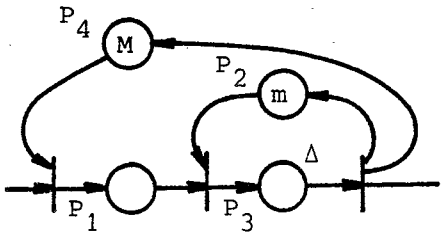
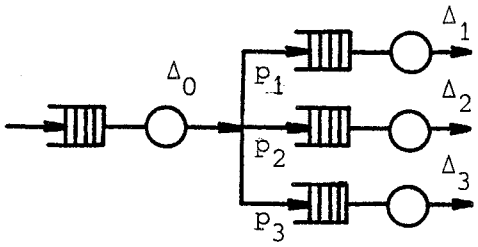
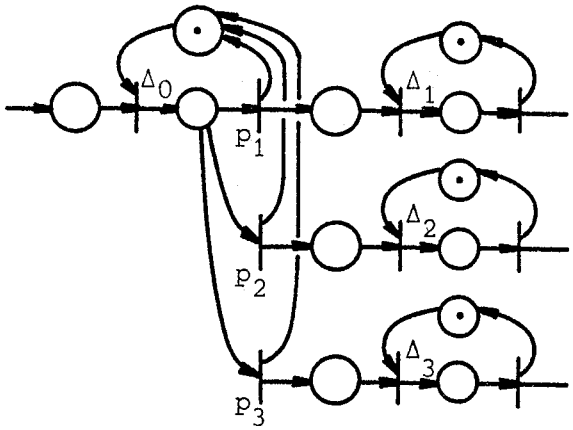
Description dans un modèle à files d'attente	*Description équivalente dans un RdPI*
	
[1] *Station à serveur infini (retard pur)*	*Une place temporisée avec Δ suffit à représenter un retard pur*
	
[2] *Station à m serveurs de capacité infinie, durée de service aléatoire selon une distribution Δ*	*La place P_1 représente la file d'attente. La place P_2 initialisée avec m marques représente les serveurs à l'état au repos. La place P_3, temporisée avec Δ, représente les serveurs à l'état actif*
	
[3] *Idem à [2] avec capacité limitée (pas plus de M clients en attente ou en cours de service dans la station*	*La place P_4 initialisée avec M marques représente les cases vides dans la file d'attente*

Figure 8. Passage d'un modèle à files d'attente au modèle RdPI (première partie)



4 *Aiguillage probabiliste en fin de service*



Le cheminement des clients d'une station à l'autre (routage déterministe ou probabiliste) dans le modèle à files d'attente s'exprime à l'aide de l'interprétation dans le modèle RdPI. Il en est de même de la discipline de service dans le cas où l'on associe aux clients des attributs (heures d'arrivée, priorités, ...).

Figure 8. Passage d'un modèle à files d'attente au modèle RdPI (suite et fin)

faut que cette marque reste indisponible dans la place pendant une durée non nulle.

La fonction ν_{p_i} qui détermine cette durée peut être une constante ou définie à l'aide d'une expression quelconque à valeurs dans l'ensemble des réels. Une extension intéressante serait d'autoriser l'emploi de fonctions de distribution du genre de celles que l'on utilise couramment dans les modèles probabilistes à files d'attente.

Moyennant cette extension, toute description utilisant les concepts classiques des modèles à files d'attente peut être simulée dans MAS. La description MAS équivalente s'obtient systématiquement en remplaçant chaque noeud dans le modèle à files d'attente par la structure RdPI correspondante (voir figure 8).

Par rapport au modèle à files d'attente, les RdPI offrent une plus grande aptitude à décrire, de façon graphique et donc visuelle, les problèmes de synchronisation (synchronisation sur ressources, sous-réseaux à capacité limitée, etc.).

2.3 La description de la partie commande de chaque module peut comprendre, sous le mot-clé ACTIONS_LOCALES, la spécification d'un ensemble de "tâches", à condition que ces tâches n'affectent d'aucune manière le fonctionnement décrit par le RdPI. Ces tâches seront effectuées une fois à la fin de la phase A d'initialisation de la simulation et à la fin de chaque phase B.

Cela permet la programmation de certaines actions d'"espionnage" du comportement du module (par exemple, pour effectuer des mesures). On a aussi, par ce biais, la possibilité de décrire de façon classique le "combinatoire de sortie" d'un module d'un automatisme séquentiel ; le calcul décrivant ce combinatoire de sortie peut alors se référer aux valeurs des variables représentant les places.

2.4. La description des parties commande peut comprendre également la spécification d'invariants que la simulation prendra en charge de contrôler en permanence. On peut distinguer essentiellement deux types d'invariants :

- des invariants intrinsèques qui sont liés uniquement aux propriétés caractéristiques du modèle. Par exemple, l'utilisateur peut demander que des propriétés telles que la persistance, la propriété de sauf ou la propriété de déterminé soient contrôlées en permanence au cours de la simulation ;
- des invariants fonctionnels qui n'ont de signification que pour le fonctionnement qui est décrit. Un invariant fonctionnel peut être, par exemple, le fait qu'une variable donnée ne puisse jamais prendre une valeur supérieure à ... ou le fait que deux places quelconques du RdPI ne puissent jamais contenir simultanément des marques.

Les variables du type PLACE peuvent être utilisées pour exprimer ces invariants. La détection de la non-vérification d'un invariant conduit à la suspension de la simulation.

Remarquons ici que, contrairement au cas des invariants fonctionnels, le contrôle des invariants intrinsèques n'est possible que si la structuration interne des données ainsi que les algorithmes du simulateur ont été conçus en conséquence.

Il n'est évidemment pas réaliste d'envisager, par la simulation, la vérification de toutes les propriétés caractéristiques d'un RdPI. Par exemple, pour ce qui concerne la vérification de la propriété "borné", la simulation ne peut que vérifier qu'aucun des marquages des places ne dépassera une borne m (m = 1 pour la propriété "sauf"). Tout au plus, la simulation peut noter les marquages minimum et maximum qui seront atteints par chaque place, ce qui peut être exploité aussi bien pour la vérification de la propriété borné que pour avoir une estimation réelle de la "distance synchronique" entre deux transitions, moyennant l'injection de places fictives dans le réseau (voir remarque dans chap. VI, § I - d).

Cela n'a pas de sens non plus d'envisager, par la simulation, la vérification de la propriété de "vivacité". La simulation peut détecter certaines situations de blocage total (en particulier, le cas où le réseau atteint un marquage qui ne valide aucune transition). La simulation peut également détecter certaines situations de blocage partiel à condition que soient spécifiés au simulateur les indices qui permettent cette détection. Par exemple, on peut effectuer au préalable une analyse du réseau qui permet de déterminer les verrous ; on spécifie ensuite au simulateur les invariants concernant ces verrous.

Enfin, la vérification de la propriété de "déterministe" d'un RdPI, bien que techniquement possible, conduirait à des algorithmes de simulation trop complexes et à des temps de simulation exorbitants. Par contre, la vérification des propriétés de "déterminé" et de "persistance" est beaucoup plus aisée à réaliser.

3. Actions globales et invariants globaux :

Les actions locales et invariants locaux que l'on peut introduire dans la description de la partie commande d'un module ne peuvent se référer qu'à des valeurs de variables propres au module. La spécification d'actions globales et d'invariants globaux pouvant se référer à l'ensemble des variables du système peut être faite dans la section du programme réservée à la description de la configuration du système (section DCLCONF).

4. Description de l'environnement externe :

4.1. Le comportement de l'environnement externe au système peut être décrit de deux façons selon que l'on s'intéresse uniquement à engendrer des occurrences d'événements externes (fonctionnement du système en boucle ouverte) ou que l'on veuille simuler une interaction à double sens entre l'environnement externe et le système (fonctionnement en boucle fermée) :

- Dans le premier cas, la séquence des occurrences des événements externes est décrite dans la section INIT (sous-section DONNEES) du programme de simulation.

- Dans le deuxième cas, une partie du comportement de l'environnement externe peut être décrite par un ou plusieurs modules (de la même manière que sont décrits les modules faisant réellement partie du système). On peut ainsi simuler, dans un même programme, à la fois le comportement d'un système de commande et celui de l'environnement avec lequel il interagit.

4.2. En principe, les interactions aussi bien des modules entre eux que des modules avec l'environnement externe doivent s'effectuer exclusivement par l'intermédiaire de variables d'interface communes. Exceptionnellement, on peut autoriser, à partir de la sous-section DONNEES de la section INIT, l'accès direct à des variables internes aux modules. Comme nous l'avons dit, cela permet de simuler des effets de pannes. Dans ce même but, on peut introduire des primitives qui permettent le "verrouillage" de certaines des variables à des valeurs définies.

4.3. La primitive STOP ne doit pas nécessairement terminer la description de la séquence des occurrences d'événements externes dans la sous-section DONNEES. Elle peut être placée en tout point de cette séquence et constitue, de ce fait, une facilité pour la mise au point du programme de simulation. Rappelons que cette primitive sert à spécifier l'instant où la simulation doit être arrêtée. Pour une simulation donnée, seules les occurrences d'événements dont la spécification apparaît en amont de la primitive STOP sont pris en compte.

5. Précisions sur l'algorithme de simulation :

5.1. Du point de vue de la simulation, l'ensemble des RdPI décrivant les modules du système constitue un seul RdPI auquel on applique l'algorithme de simulation dont le principe a été rappelé dans le paragraphe II - 2.

Tenant compte des précisions que nous venons d'apporter, cet algorithme peut être résumé comme suit :

A - Phase d'initialisation :

A10 - Initialisation du marquage et des variables de chaque module ;

A11 - vérification de la stabilité de ce marquage initial ;

A12 - contrôle des invariants locaux et globaux ;

A13 - exécution des actions locales et globales ;

A15 - établissement d'un échéancier de simulation.

B - Phase de traitement des échéances associées à un même instant de l'échéancier

B10 - Validation du passage de marques de l'état indisponible à l'état disponible et exécution des procédures associées aux places dans lesquelles ces changements d'état de marques sont effectués ;

B20 - exécution d'une itération de tirs sur occurrence de $\{e\}$, avec contrôle des invariants locaux et globaux à la fin de chaque tir, jusqu'à ce qu'un marquage stable soit atteint ;

B30 - prise en compte des occurrences d'événements externes associées à cet instant dans l'échéancier : les changements de valeurs de variables traduisant ces occurrences d'événements sont effectués que ce soit sur les variables d'entrée des modules ou sur les variables internes aux modules ;

B31 - Grâce aux anciennes valeurs des entrées de chaque module (valeurs mémorisées dans les variables $VES), le simulateur évalue les expressions définissant les variables du type EVENEMENT, réactualise les valeurs des variables $VES, effectue un premier tir sur occurrence de $\{e\}$ à la suite de quoi il remet à zéro toutes les variables du type EVENEMENT, poursuit enfin une itération de tirs sur occurrence de $\{e\}$ jusqu'à ce qu'un marquage stable soit atteint. A la fin de chaque tir, les invariants locaux et globaux sont contrôlés ;

B32 - nouvelle exécution des actions locales et globales.

5.2. Tout l'aspect opérationnel lié à l'exploitation de MAS (récupération des résultats de la simulation, introduction de points d'arrêt, modes d'exploitation (par lot ou en conversationnel), messages d'erreurs, ...) reste évidemment à étudier.

III - PASSAGE A UNE REALISATION EFFECTIVE D'UN SYSTEME VALIDÉ PAR MAS

Nous avons proposé dans les chapitres précédents d'utiliser pour la réalisation le même modèle que celui de la conception, à savoir les RdPI.

Dans le cas des réalisations programmées utilisant un matériel informatique classique (micro ou minicalculateur), une solution consiste à développer un "interpréteur" capable d'accepter en entrée une description du même genre que celle que peut accepter le simulateur MAS. La même structure interne des données ainsi que les mêmes algorithmes de simulation utilisés dans MAS peuvent être repris pour l'interpréteur, en supprimant bien sûr toutes les procédures qui auront été introduites dans MAS à des fins de contrôle et de vérification et qui s'avèreront peu utiles au niveau de la réalisation. Dans ce même objectif, on peut envisager que le langage d'implantation de l'interpréteur puisse être le même que celui du simulateur (un langage standard de haut niveau, facilement transportable d'un matériel informatique à un autre). Les langages de transcription des RdPI à l'entrée de l'interpréteur et à l'entrée du simulateur, pour ce qui concerne notamment la programmation de la structure des RdP dans les parties commande et la programmation des procédures dans les parties opératives, pourront ainsi être quasiment les mêmes. La section INIT dans le programme de simulation se transformerait dans le programme d'entrée de l'interpréteur en un ensemble de tâches prenant en charge de gérer des dispositifs périphériques d'entrée/sortie pour réaliser les interactions du système avec son environnement externe.

Il va de soi que des précautions doivent être prises concernant la réalisation des échanges entre les différents modules du système dans le cas où l'implantation de ces modules doit être répartie entre divers noeuds d'un réseau de calculateurs. Comme nous l'avons dit, la simulation considère l'ensemble des RdPI décrivant les modules comme étant un seul RdPI. Si une synchronisation par un véritable protocole d'échange n'est pas établie

entre les différents modules, il est évident que le fonctionnement que l'on obtient en simulant tout le système sur un seul processeur peut ne pas être le même que celui où chaque module est simulé par un processeur séparé.

La solution de l'interpréteur a l'avantage de réduire les risques d'introduction d'erreurs lors du passage de la conception à la réalisation. Cependant, elle nécessite, de façon générale, un volume de mémoire plus important que ne nécessiterait une programmation moins systématique qui chercherait à optimiser l'espace mémoire utilisé. Aussi, ce genre de solution peut ne s'avérer économiquement viable que dans la mesure où le même support matériel de réalisation ainsi que le même interpréteur sont conçus de façon à servir à l'implantation en petites séries d'applications diverses.

Dans le cas des applications de moyennes et grandes séries pour lesquelles la conception d'une architecture matérielle spécifique se justifie et où l'optimisation de l'espace mémoire peut apparaître comme étant une contrainte principale, il est tout à fait indiqué d'utiliser un langage intermédiaire de haut niveau (éventuellement, un langage libre imaginé pour les besoins spécifiques de l'application) qui permette dans un premier temps de traduire de façon algorithmique les descriptions obtenues en fin de conception et facilite par la suite un passage systématique au codage final [MOA.2.79] [MOA.3.79].

IV - ETUDE DE CAS : APPLICATION DES RdPI A LA SPECIFICATION, L'ANALYSE ET LA CONCEPTION DE L'AUTOMATISATION D'UN ATELIER FLEXIBLE DE MASTICAGE DE CARROSSERIES AUTOMOBILES [MOA.80][CAV.81].

L'exemple que nous allons étudier illustre clairement, sur un cas concret, l'intérêt de l'utilisation d'un simulateur tel que MAS. Cet exemple vise aussi à montrer quelques uns des apports méthodologiques fondamentaux de l'utilisation des RdPI pour la spécification, l'analyse et la conception de systèmes de commande complexes.

IV - 1. PRESENTATION DE L'APPLICATION

Dans la gamme des opérations de peinture des carrosseries automobiles,

l'application des mastics d'étanchéité s'insère entre le dépôt anti-corrosion effectué par trempée sous cataphorèse et la pulvérisation des peintures de préparation avant laques. Ces mastics servent à étancher les joints entre diverses pièces de tôlerie. L'application de ces mastics est réalisée manuellement à l'aide de pistolets d'injection.

Traditionnellement, ces opérations étaient exécutées sur convoyeur à défilement continu, les opérateurs étant répartis le long de ce convoyeur. Cette organisation de la production, largement éprouvée dans les fabrications de série, présente des inconvénients parmi lesquels on peut citer :

- Une parcellisation excessive du travail, d'où une répétitivité accrue.
- Une répartition des tâches élémentaires très difficile à gérer en cas de fluctuation des effectifs disponibles, de modification de cadence ou de diversification des modèles simultanément fabriqués.
- Un défilement continu du produit qui rend difficiles les travaux délicats, d'où des problèmes de qualité.

Pour éviter ces inconvénients, un atelier de masticage à postes autonomes a été étudié et implanté.

La desserte des postes en carrosseries à mastiquer est organisée à partir de modules de manutention appelés "tables à rouleaux".

Une table à rouleaux est constituée par un ensemble de rouleaux entrainés par un moteur électrique. Ces rouleaux assurent l'acheminement des carrosseries posées sur des patins de roulement. Chaque table à rouleaux a une longueur suffisante pour recevoir une seule carrosserie.

Une table à rouleaux peut elle-même être fixée sur un mécanisme de pivotement ou de translation permettant ainsi des changements d'orientation et des aiguillages.

L'atelier implanté doit être capable d'assurer le masticage de 4 types de carrosseries à des cadences moyennes de 70 véhicules à l'heure. Son schéma synoptique est donné dans la figure 9. Il comporte 12 postes de travail (P) répartis en deux groupes de 6. Chaque poste est aménagé pour que 4 opérateurs puissent y travailler.

En amont de chaque poste de travail, un banc de transfert à deux

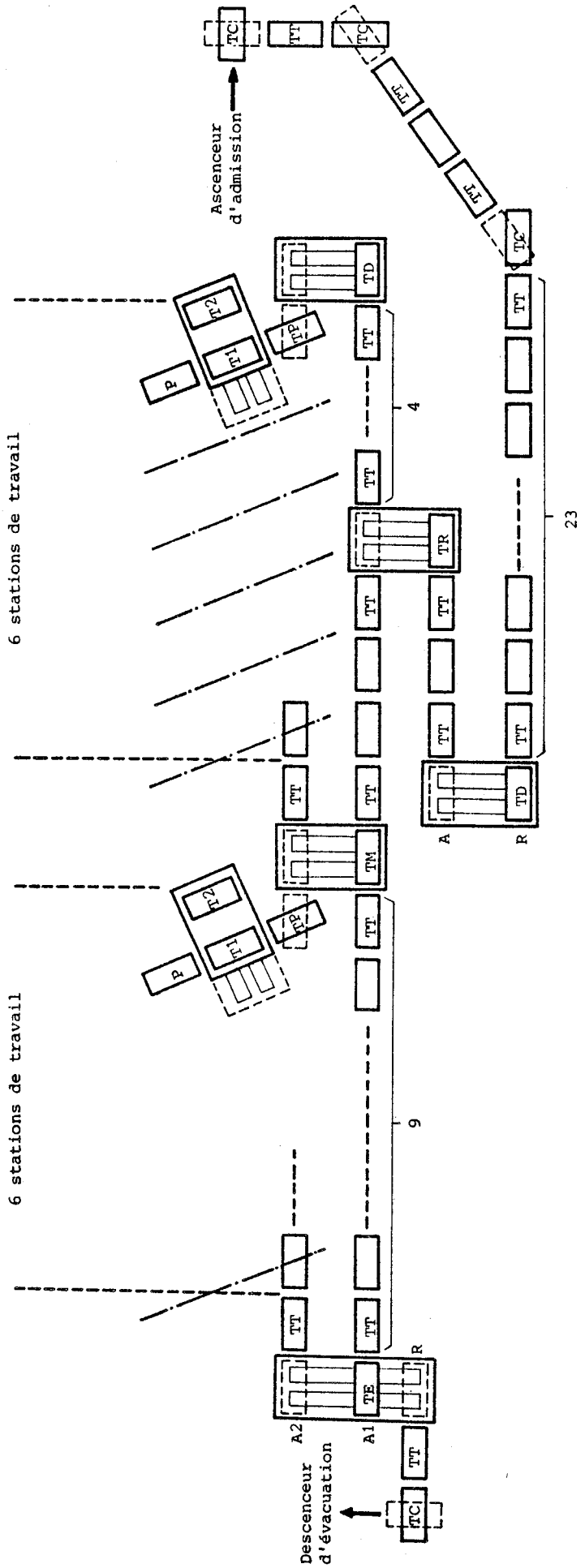


Figure 9. Schéma synoptique de l'atelier de masticage

tables à rouleaux (T1, T2) est destiné à réduire au mieux le temps séparant le départ d'une carrosserie mastiquée de l'arrivée d'une carrosserie à mastiquer au poste de travail.

Pour chacun des groupes de 6 postes, une ligne de tables à rouleaux pivotantes (TP) assure l'acheminement mixé des carrosseries à destination ou en provenance des bancs de transfert.

Le flux des carrosseries à mastiquer est divisé en amont des 2 groupes (TR) et chacun d'eux est alimenté par un petit stock.

Le flux des carrosseries mastiquées sur le premier groupe croise celui des carrosseries à mastiquer sur le second, sur une table de transfert dite de multiplexage (TM).

Les flux des carrosseries mastiquées sur les deux groupes sont ensuite regroupés avant de passer dans les ateliers suivants.

IV - 2. PROBLEMES POSES PAR L'AUTOMATISATION

Conçue pour éviter les inconvénients d'une chaîne traditionnelle, l'exploitation satisfaisante de cette installation conduit à prendre en compte divers problèmes nouveaux.

D'une part, pour adapter les moyens mis en oeuvre aux évolutions du programme de fabrication, la modularité de l'installation permet de faire varier :

. le nombre de postes de masticage en service ;

. les effectifs mis en place à chaque poste.

D'autre part, le temps de traitement d'une carrosserie dans un poste de masticage dépend de trois paramètres principaux :

. le type de carrosserie mastiquée (4 types différents) ;

. le nombre d'opérateurs au poste considéré (de 1 à 4) ;

. le niveau d'activité des opérateurs, que ces derniers doivent pouvoir moduler à leur gré dans les limites possibles d'approvisionnement et d'évacuation de leur poste de travail.

Le complet asynchronisme du fonctionnement de chacun des postes pose la nécessité d'un système de gestion. La spécification de ce système de gestion devra permettre de préciser ses deux fonctions principales, qui sont les suivantes :

1) Suivi de l'activité au niveau de chaque poste de travail :

En fonction du nombre d'opérateurs affectés par poste, un temps normal d'exécution est alloué pour chaque type de carrosserie. Partant de ces données et de la succession des types de carrosseries à chaque poste, ce suivi devra permettre, à tout instant, de comparer la somme des temps alloués aux carrosseries mastiquées depuis le début du travail au temps réellement écoulé.

Cette comparaison permanente est nécessaire aux opérateurs pour leur permettre de gérer leur niveau d'activité. Elle est aussi nécessaire, bien évidemment, au responsable de la fabrication. Ces données sont visualisées au poste de travail et au poste central de contrôle de la fabrication.

Ce système devra aussi prendre en compte des aléas de fabrication tels que les temps de manque d'approvisionnement, de pannes partielles ou totales des installations.

2) Gestion des approvisionnements et des évacuations des postes de travail :

Comme cela a été dit précédemment, cet atelier s'insère dans une suite d'opérations. Il est alimenté depuis l'installation amont par l'intermédiaire d'un stock.

L'installation de masticage elle-même, par sa structure mécanique, peut fonctionner avec un en-cours de carrosseries (nombre de carrosseries existant, à un instant donné, entre l'entrée et la sortie de l'installation considérée) très variable.

L'installation aval est alimentée par l'intermédiaire d'un stock.

De l'état de remplissage de ces stocks amont et aval, et du niveau d'en-cours de l'installation, dépendent les possibilités réelles de modulation d'activités des postes de travail.

Aussi, dans toutes les configurations possibles, la gestion des postes devra tendre à ne pas favoriser l'un des groupes de six postes ou l'un des postes dans un même groupe malgré les dissymétries ou particularités de l'implantation.

La non-satisfaction de cette contrainte complexe présente de gros

risques de conflits sociaux dans l'atelier.

IV - 3. APPORTS ATTENDUS DES OUTILS CAO

La conduite de projets de ce type pose tout d'abord la nécessité d'un langage de description et d'une méthodologie qui permettent de :

- maîtriser la complexité des problèmes posés ;
- mettre clairement en évidence les possibilités de parallélisation et d'ordonnancement des activités des postes de travail ;
- étudier en conséquence les détails de l'architecture de l'installation mécanique, ses performances et ses dimensionnements ;
- analyser les algorithmes de gestion possibles ;
- dégager, progressivement, les éléments de l'instrumentation (capteurs, actionneurs, ...) qui doivent constituer l'interface automatisme-installation.

Ensuite apparaît le besoin d'outils d'évaluation et de simulation qui permettent de :

- quantifier de façon précise les performances de l'installation ;
- valider les dimensionnements ;
- comparer et choisir parmi les algorithmes de gestion envisagés.

Enfin apparaît le besoin d'outils CAO d'aide à l'implantation et à la mise au point, qui permettent de faciliter la concrétisation de l'automatisation tout en y intégrant les exigences de sûreté de fonctionnement et de qualité de service de façon générale.

IV - 4. UTILISATION DES RdPI

L'analyse du schéma synoptique donné en figure 9 permet de distinguer 7 types de modules : des tables charnières (TC) ; des tables de transfert (TT) ; une table de décalage pour changement de file (TD) ; une table de répartition des carrosseries à mastiquer (TR) ; des stations de travail (ST), chaque station comprenant une table pivot (TP), un banc de transfert à deux tables (T1, T2) et une table poste de travail (P) ; une table de multiplexage (TM) et, enfin, une table d'évacuation (TE).

Les modules des trois types TC, TT et TD sont amenés à avoir des fonctionnements répétitifs dépendant exclusivement du défilement du flux de carrosseries. Par contre, les fonctionnements des modules des types

TR, ST, TM et TE vont devoir dépendre des décisions prises au niveau d'un algorithme de gestion en temps réel de l'ensemble de l'atelier. Par exemple, le fonctionnement de la table de répartition (TR) va forcément dépendre de l'algorithme de gestion de la répartition des carrosseries en fonction, d'une part, de la surcharge ou de la pénurie des carrosseries à mastiquer arrivant dans l'atelier et, d'autre part, des cadences de travail dans les deux groupes de postes de travail (nombre de postes en fonctionnement dans chaque groupe, affectation des opérateurs entre les postes, ...).

Le type de module où les décisions à prendre par l'algorithme de gestion sont les plus nombreuses et les plus importantes est celui des stations de travail (ST). Nous l'étudierons en premier. Nous étudierons ensuite un module des autres types, en l'occurrence le module table de transfert (TT).

1. Etude de l'automatisation d'un module station de travail

Le RdPI de la figure 10 donne une modélisation de haut niveau du fonctionnement d'une station de travail. Cette modélisation a pour but de :

- visualiser les différents états possibles des trois éléments qui constituent une station de travail (P, (T1, T2), TP) ;
- mettre clairement en évidence les évolutions possibles entre ces états ;
- dégager les points de décision où l'algorithme de gestion de l'atelier doit intervenir ;
- analyser les algorithmes de gestion possibles.

Les places P_1, P_t et P_p représentent respectivement les états "table P libre", "table P occupée par une carrosserie en cours de masticage" et "table P occupée par une carrosserie mastiquée". A l'instant initial, la table est dans l'un de ces états ; on spécifie lequel de ces états en déposant une marque dans l'une des trois places.

Le banc de transfert a deux positions d'arrêt : "gauche" et "droite" représentées respectivement par les places G et D. Une des deux places doit être marquée initialement.

Dans la position gauche, on utilise la table T2 comme zone tampon pour l'approvisionnement du poste de travail en carrosseries à mastiquer.

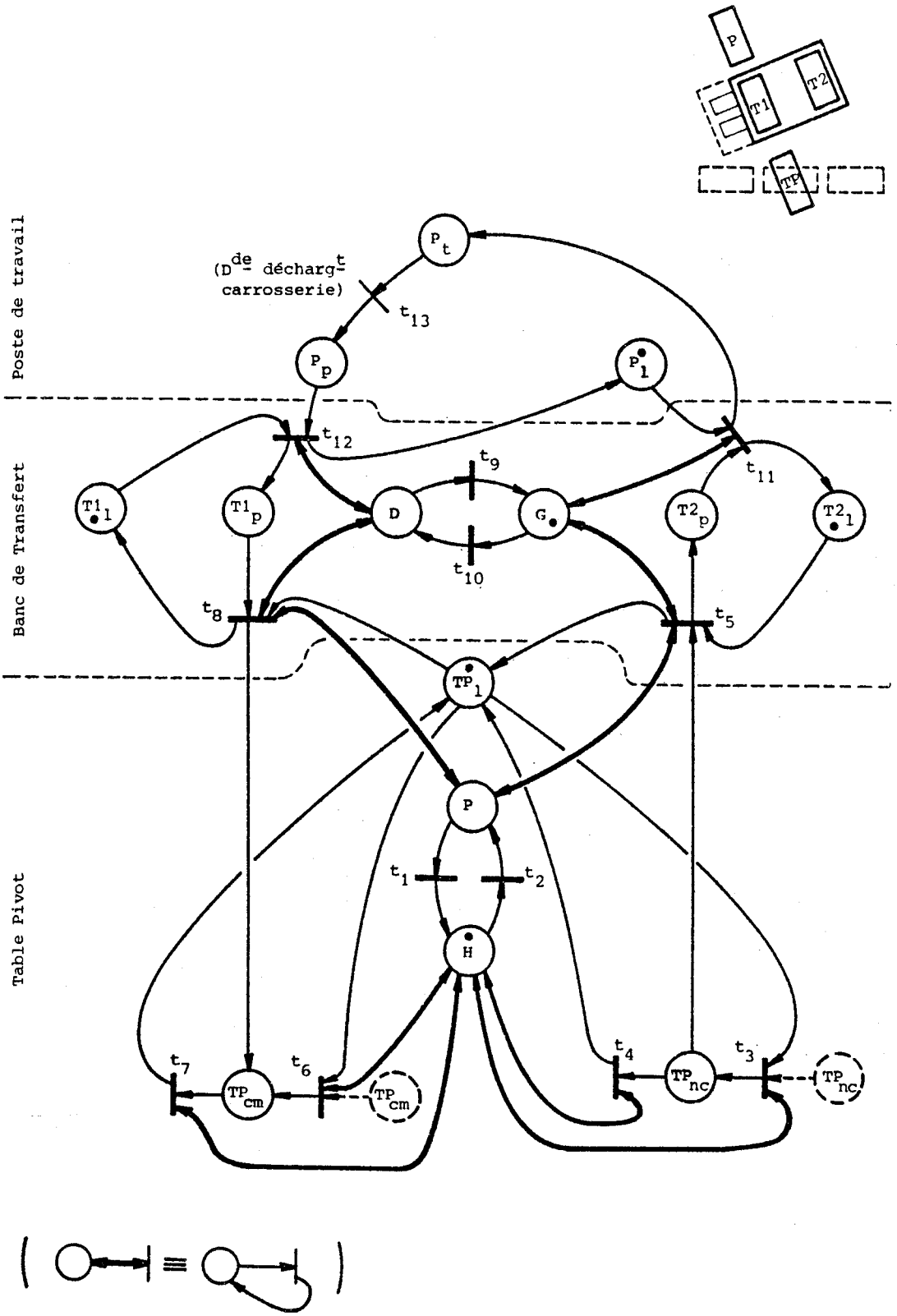


Figure 10. Evolutions possibles des états d'une station de travail

Les places $T2_1$ et $T2_p$ représentent respectivement les états "table T2 libre" et "table T2 chargée". Une des deux places doit être marquée initialement.

Dans la position droite, on utilise la table T1 comme zone tampon pour le déchargement du poste de travail. Les places $T1_1$ et $T1_p$ représentent respectivement les états "table T1 libre" et "table T1 chargée". Une des deux places doit être marquée initialement.

Le choix d'utiliser la table T2 pour l'approvisionnement et la table T1 pour le déchargement est suggéré par le mécanicien ayant conçu l'installation. L'étude de l'automatisation de la station peut valider ce choix ou le remettre en cause.

Les places H et P représentent les positions "horizontale" et "en pivot" de la table pivot. Une des deux places doit être marquée initialement.

Enfin, les places TP_1, TP_{nc} et TP_{cm} représentent respectivement les états "table pivot libre", "table pivot chargée par une nouvelle carrosserie à mastiquer" et "table pivot chargée par une carrosserie mastiquée à évacuer". Une des trois places doit être marquée initialement.

Les transitions décrivent les évolutions possibles entre ces états. Ainsi, la transition t_3 indique que si la table pivot est libre, si elle est en position horizontale et si la table pivot de la station en amont est chargée par une nouvelle carrosserie à mastiquer (place représentée en pointillé), alors l'évolution à l'état "table pivot chargée par une nouvelle carrosserie" est possible. On peut comprendre facilement la signification de toutes les autres transitions.

Cette modélisation est systématique. Le concepteur du projet peut vérifier sur cette modélisation si toutes les possibilités d'évolutions qu'il a envisagées sont représentées. Il peut discuter avec l'automaticien les raisons de ces possibilités et, éventuellement, l'opportunité ou l'inconvénient d'autres possibilités d'évolutions.

L'étude de l'automatisation peut alors commencer, et la première question que va se poser l'automaticien est de savoir : quelles sont les évolutions qui doivent être effectuées de façon systématique chaque fois que les conditions nécessaires sont remplies, et quelles sont celles qui sont assujetties à la décision de l'algorithme de gestion de l'atelier ?

On s'aperçoit, qu'en fait, seule l'évolution représentée par la transition t_{13} doit être effectuée de façon systématique dès que la transition est validée et qu'une demande de déchargement émanant des opérateurs du poste de travail indique que le masticage de la carrosserie dans le poste est terminé. (Un interface opérateur-automatisme doit être prévu à ces fins).

Toutes les autres transitions représentent des points de décision de l'algorithme de gestion de l'atelier. Par exemple, le franchissement de la transition t_3 conduit à occuper la table pivot par une carrosserie à mastiquer. Avant de décider ce franchissement, l'algorithme de gestion doit tout d'abord vérifier que la station en question ou l'une au moins des stations en aval est demandeuse d'une nouvelle carrosserie (ce qui revient à vérifier que la place $T2_1$ de la station ou de l'une des stations en aval contient une marque). D'un autre côté, si au même instant la table T1 se trouve chargée par une carrosserie mastiquée à évacuer (place $T1_p$ contient une marque), l'algorithme de gestion peut trouver plus opportun d'utiliser la table pivot pour la décharger. Tout dépend, en fait, des urgences, à cet instant précis, du déchargement de la table T1 et de l'approvisionnement de la station elle-même ou des stations en aval en carrosserie à mastiquer.

Une analyse plus poussée montre que l'évolution représentée par la transition t_3 est régie par trois sous-ensembles de conditions :

C1 : des conditions qui, étant satisfaites, rendent possible la décision d'effectuer l'évolution : 1°) il faut que la table pivot soit libre (place TP_1 contient une marque), et 2°) il faut que la table pivot de la station en amont (représentée par la place en pointillé) soit chargée par une carrosserie à mastiquer ;

C2 : des conditions liées à l'état du système global, qui permettent à l'algorithme de gestion de décider quand il est opportun d'effectuer cette évolution ;

C3 : des conditions que l'algorithme de gestion doit assurer s'il décide d'effectuer l'évolution : 1°) il faut verrouiller la table pivot dans la position horizontale. Si la table se trouve en position pivot, il faut décider le franchissement de t_1 pour amener la table en position horizontale puis la verrouiller ; 2°) il faut également verrouiller la table pivot de la station en amont dans la position horizontale.

Une fois l'évolution effectuée (transition t_3 franchie), l'algorithme de gestion est appelé de nouveau à décider entre le franchissement de la transition t_4 ou celui de la transition t_5 ...

On voit bien à quel point une telle modélisation peut aider l'analyse de l'automatisation et favoriser un dialogue non ambigu entre les concepteurs du projet spécialistes du processus de production, les automaticiens et les futurs utilisateurs de l'installation.

La conception de l'automatisation peut être dérivée directement à partir d'une telle modélisation.

2. Modélisation de l'atelier en vue de valider les algorithmes de gestion

De la modélisation décrite ci-dessus, on peut en déduire une modélisation plus simplifiée mais qui prend en compte les temps correspondant aux mouvements mécaniques et à la durée des opérations de masticage effectuées aux postes de travail. L'interconnexion de telles modélisations effectuées pour l'ensemble des modules-types de l'atelier permet de constituer une modélisation globale de l'atelier dont la simulation permet de valider les algorithmes de gestion envisagés. Dans cette modélisation de l'atelier, on peut utiliser des places pouvant contenir simultanément plusieurs marques pour représenter un tronçon à n tables de transfert.

3. Etude de l'automatisation d'une table de transfert

La figure 11 donne une modélisation du fonctionnement d'une table de transfert. Comme nous l'avons dit plus haut, les évolutions de la commande d'une table de transfert dépendent exclusivement du flux de carrosseries.

Cette première description suppose inconnus encore les choix technologiques concernant l'instrumentation de commande qui sera utilisée pour concrétiser les actions associées aux places et les réceptivités associées aux transitions. Elle vise précisément à poser ce problème du choix de l'instrumentation de commande conduisant à la spécification de l'interface d'une table à rouleaux avec l'automatisme.

En plus du moteur d'entraînement, commandé par une entrée M, on décide d'installer à l'extrémité de la table un détecteur de proximité qui indiquera par un signal à niveau D la présence d'une carrosserie sur la table (figure 12).

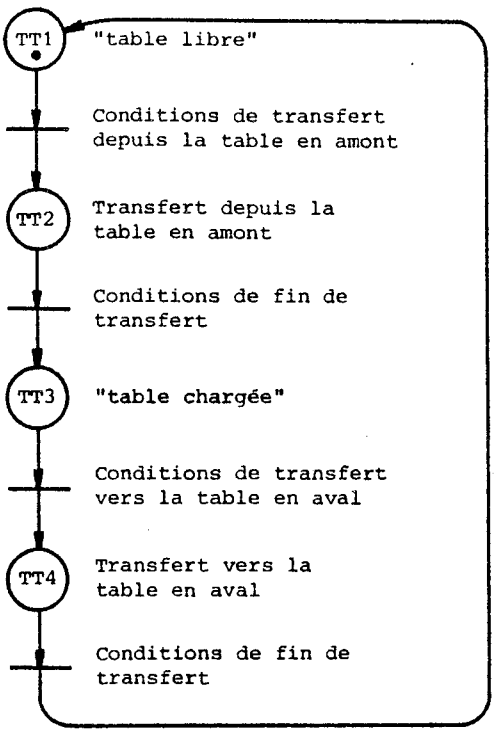


Figure 11. Fonctionnement d'une table de transfert

Le schéma de commande de la table n devient celui de la figure 13.

En fait, il peut arriver que, lors de la remise en fonctionnement de l'automatisme après un arrêt momentané de l'exploitation, la table ne soit pas à l'état "libre" mais à l'état "chargée" ; auquel cas, l'initialisation doit marquer la place TT3 au lieu de la place TT1. On peut résoudre ce problème de l'initialisation en transformant le schéma de commande de la figure 13 en le schéma donné par la figure 14. Un invariant de cette commande, qui peut être contrôlé en permanence à des fins de sûreté de fonctionnement, est que une et une seule des trois places doit être marquée à tout instant.

Le travail de l'automaticien serait quasiment terminé s'il pouvait disposer d'un outil d'implantation qui accepterait directement en entrée des descriptions de ce type.

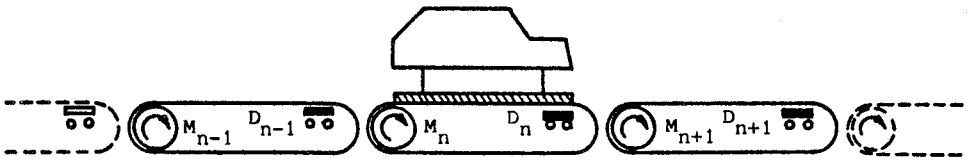


Figure 12. Instrumentation de commande et de contrôle d'une table de transfert

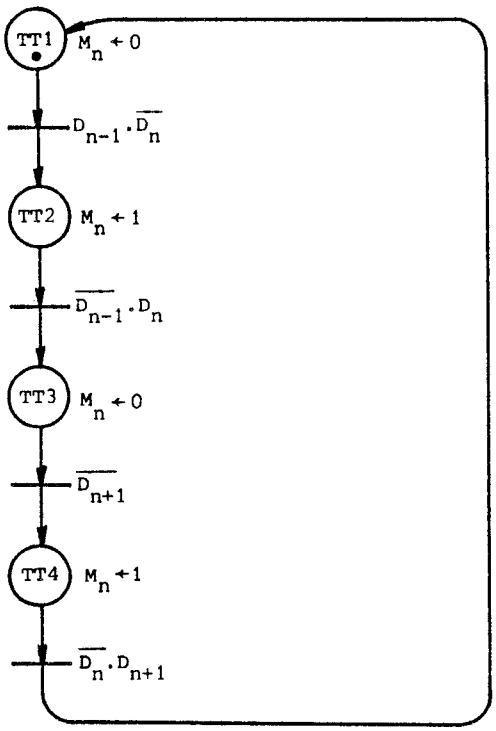


Figure 13. Etude de la commande d'une table de transfert

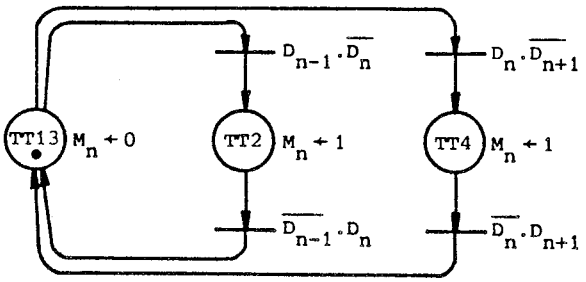


Figure 14. Automate de commande d'une table de transfert

V - CONCLUSION

L'exemple que l'on vient d'étudier montre bien que face à des systèmes assez complexes, l'outil RdPI, qui apporte déjà par lui-même des facilités de description et d'analyse des problèmes, gagne cependant à être enrichi d'un simulateur tel que MAS.

Nous avons énoncé dans ce chapitre les caractéristiques souhaitables de ce simulateur et nous en avons proposé une définition externe qui réponde à ces caractéristiques. Nous avons tenu, au niveau de cette définition, à rester cohérent avec les points clés de notre démarche, à savoir la hiérarchisation, la modularité et la validation. Nous avons aussi esquissé la démarche qui permettrait le passage automatique d'une solution simulée, et donc "validée" par MAS, à une réalisation programmée sur mini ou microcalculateur, ce qui représenterait l'aboutissement de l'approche qui sous-tend ce mémoire.

En ce qui concerne le simulateur, il reste maintenant à en étudier de façon approfondie la réalisation technique : fixer sa syntaxe, choisir un langage hôte, comparer différents algorithmes et structures de données du point de vue vitesse de simulation, volume mémoire, etc.

Pour ce qui est du passage à la réalisation effective, la solution envisagée de l'interpréteur devra être confrontée, en fonction des domaines d'utilisation, avec des solutions du type "compilées" produisant directement des programmes, des microprogrammes ou des schémas de câblage.

BIBLIOGRAPHIE (*Partie III*)

[BAE.74] BAER J.L. : Models for the Design, Simulation and Performance of Distributed-Function Architecture, Computer, Mars 1974, pp. 25-30.

[CAP.2.78] CAPLAIN M., MOALLA M. : Rapports de travaux concernant la conception et la réalisation d'un simulateur de réseaux de PETRI Interprétés, Contrat DRET - SINTRA - ENSIMAG, n° 77/34/302/00480/7501, 1978.

[CAV.81] CAVANNA B., DOLLE F., MOALLA M. : Outils C.A.O. pour l'analyse, la spécification et la réalisation de l'automatisation d'un équipement de production mécanique, 3èmes Journées Scientifiques et Techniques de la Production Automatisée (ADEPA), TOULOUSE, Juin 1981.

[CHA.79] CHAMBON Ph. : Simulateur de Réseaux de PETRI, 7ème Colloque sur le traitement du signal et de ses applications, NICE, Mai 1979.

[DAH.66] DAHL. O.J., NYGAARD K. : SIMULA : an Algol Based Simulation Language, Comm. ACM, Vol. 9, Sept. 1966.

[DAV.79] DAVID R., MOALLA M., SAUCIER G., SILVA M. : Conception d'Automatismes Logiques Répartis - Outils et Mise en Oeuvre, Rapport Rapport de contrat DGRST, n° 77-7-0646, Octobre 1979.

[DUK.71] DUKE K.A., SCHNURMANN H.D., WILSON T.I. : System Validation by Three-Level Modeling Synthesis, IBM J. RES. DEVELOP., Mars 1971, pp. 166-174.

[GAR.75] GARDNER R.I. : A methodology for Digital System Design based on Structural and Functional Modeling, UCLA-ENG - 7488, Janvier 1975.

[IVE.72] IVERSON K.E. : A Programming Language, Ed. J. Willey, NEW YORK, 1972.

[KIV.69] KIVAT P.J., VILLAVUEVA R.E., MARKOWITZ H.M. : The SIMSCRIPT II Programming Language,Prentice Hall, Englewood Cliffs, N.J. 1969.

[LED.76] LE DANOIS P., MOALLA M., SAUCIER G., SIFAKIS J., ZACHARIADES M. : Multilevel Description and Simulation of Parallel Cooperating Processors, Kolloquium über das Parallelismus in der Informatik, ERLANGEN, RFA, Juin 1976.

[LER.76] LEROUDIER J., PARENT M. : Quelques aspects de la modélisation des systèmes informatiques par simulation à événements discrets, RAIRO Informatique, Vol. 10, n° 1, Janvier 1976, pp. 5-26.

[MER.78] MERLE D., POTIER D., VERAN M. : Q.N.A.P. - A tool for Computer performance analysis, Int. Conf. on Performance of Computer Inst., North-Holland, 1978.

[MOA.75] MOALLA M., SIFAKIS J., ZACHARIADES M. : Un langage d'aide à la conception et à la simulation de systèmes complexes, Rapport de recherche IMAG, n° 16, Oct. 1975.

[MOA.76] MOALLA M., SAUCIER G., SIFAKIS J., ZACHARIADES M. : A Design Tool for the Multilevel Description and Simulation of Systems of Interconnected Modules, 3rd Annual Symp. on Comp. Architecture, TAMPA (USA), Janvier 1976.

[MOA.2.76] MOALLA M., SIFAKIS J., ZACHARIADES M. : M.A.S, un outil d'aide à la Description et à la Conception des Automatismes Logiques, Colloque AFCET : Automatismes Logiques, Recherches et Applications Industrielles, PARIS, Déc. 1976.

[MOA.3.76] MOALLA M. : L'approche fonctionnelle dans la vérification des systèmes informatiques - Proposition d'un ensemble de méthodologies, Thèse Doc. Ingénieur, INPG-ENSIMAG, GRENOBLE, Déc. 1976.

[MOA.77] MOALLA M., SIFAKIS J. : A Design Methodology for Complex Logical Systems, 2nd Int. Symp. on Discrete Systems, DRESDEN, Mars 1977.

[MOA.2.79] MOALLA M. : Spécification de réalisation de la centrale de sécurité CALOR 40.11, Doc. OPTION, Réf. 78-12-158.041-02, Mai 1979.

[MOA.3.79] MOALLA M. : Spécification de réalisation du programmateur de four CIT-ALCATEL, Doc. OPTION, Réf. 79-11-046, Déc. 1979.

[MOA.80] MOALLA M. : Modélisation et analyse du système de masticage de carrosseries, Note interne DAST-RENAULT, Août 1980.

[SCH.74] SCHRIBER T.J. : Simulation using GPSS, John Wiley and Sons, 1974.

[SIF.74] SIFAKIS J. : Modèles temporels des systèmes logiques, Thèse Doc. Ingénieur, USMG-INPG, GRENOBLE, Mars 1974.

[WIR.76] WIRTH N. : Algorithms + Data Structures = Programs, Prentice Hall, Englewood Cliffs, N.J., 1976.

[ZAC.77] ZACHARIADES M. : MAS : Réalisation d'un langage d'aide à la description et à la conception de systèmes logiques, Thèse 3ème cycle, Univ. de GRENOBLE, Sept. 1977.

CONCLUSION

Ce mémoire a tenté de présenter de façon synthétique un ensemble de travaux ayant porté sur l'utilisation des Réseaux de PETRI (RdP) et de modèles dérivés pour l'aide à la conception de systèmes à fonctionnement non-autonome.

En 1975, date à laquelle nous avons commencé ces travaux, l'utilisation des RdP pour la description et l'analyse des systèmes à évolutions parallèles était orientée essentiellement vers la modélisation des systèmes autonomes. Quelques travaux seulement avaient abordé l'utilisation des RdP pour la modélisation des systèmes non-autonomes, en introduisant la "temporisation" [RAM.73] puis l'"interprétation" (essentiellement les travaux effectués en France visant à dériver des RdP des modèles adaptés à la description et à l'implantation des automatismes logiques ; une bibliographie assez complète peut être trouvée dans [MOA.2.78] et [BLA.2.79]).

Notre premier objectif a été d'expérimenter ces modèles sur des applications concrètes afin de dégager les concepts utiles et de définir un outil de simulation multiniveaux basé sur l'utilisation des RdP : le système MAS.

Cette expérimentation a très vite révélé la nécessité, outre la temporisation et l'interprétation, d'une troisième extension de base qui est la "synchronisation sur occurrence d'événements". La recherche d'un algorithme de simulation nous a conduit à introduire les notions d'"environnement" et de "tir sur occurrence d'événements". Ces deux notions ont notamment contribué à clarifier, par rapport aux travaux précédents, les règles de prise en compte de la temporisation et de l'interprétation.

Dès lors, nous nous sommes préoccupés de formaliser les définitions du nouveau modèle obtenu et d'étudier ses propriétés. Cette formalisation est remontée au niveau même des RdP autonomes. En particulier, l'étude de la classe constituée par les RdP à Priorité Elémentaire (RdPPE) a été intéressante à deux points de vue : (chap. IV et V)

- D'une part, elle nous a permis de situer la puissance de description de cette classe de réseaux par rapport à celles offertes par les principales extensions apportées aux RdP dans le contexte d'un fonctionnement autonome, à savoir les RdP à Arcs Inhibiteurs (RdPZ), les RdP Généralisés (RdPG) et les RdP à Priorité (RdPP). Nous avons démontré

que, sur le plan théorique, les RdPPE offrent une puissance de description au moins égale à celles de ces trois extensions (chap. IV).

- D'autre part, un lien entre les fonctionnements autonomes et le fonctionnement synchronisé des RdP a pu être établi. Nous avons montré qu'à la priorité des transitions dans les RdPPE correspond la notion d'"événement toujours présent" dans les RdP Synchronisés (RdPS). Nous avons utilisé les propositions démontrées pour les RdPPE pour situer de nouveau la puissance de description des RdP Ordinaires Synchronisés par rapport aux RdP à Arcs Inhibiteurs Synchronisés et aux RdP Généralisés Synchronisés (chap. V).

Notre préoccupation a été également de rechercher, dans la définition du modèle global constitué par les RdP Interprétés (RdPI), un compromis entre la généralité du modèle pour adapter au mieux les modalités de son utilisation à la pratique et l'intérêt de lui conserver le maximum de possibilités d'analyse et de vérification déjà étudiées pour les RdP autonomes. Nous avons montré dans le chapitre V, au fur et à mesure de la définition du fonctionnement synchronisé puis du fonctionnement synchronisé temporisé, en quoi ces extensions modifient les propriétés initiales des RdP. Nous avons étudié ensuite, dans le chapitre VI, les conditions sous lesquelles les méthodes d'analyse statique développées pour les RdP autonomes peuvent être étendues aux RdPI. Dans le cas général, les possibilités d'analyse des systèmes décrits par les RdPI s'avèrent assez faibles. Néanmoins, ce qu'il en reste est très utile pour enrichir les moyens d'une vérification partielle par simulation, à défaut de résultats absolus pouvant être assurés par l'analyse statique.

Le système MAS que nous avons défini reste ouvert à l'implantation de telles vérifications. D'une façon générale, il permet de prendre en compte la vérification de toute propriété déclarée comme devant être invariante dans le fonctionnement du système simulé. Dans la présentation que nous en avons donnée dans le chapitre VIII, nous avons surtout mis l'accent sur son usage général pour simuler des descriptions modulaires et multiniveaux, en maintenant séparées les descriptions de la partie opérative et de la partie commande de chaque module. Nous avons montré qu'un tel outil peut aussi bien servir à la validation fonctionnelle progressive de la conception d'un automatisme, qu'à l'étude et à

l'évaluation de la conception du procédé à automatiser.

Le langage de transcription des RdPI à l'entrée de MAS peut être, à peu de choses près, le langage d'un "interpréteur" qui accepterait le même type de description d'entrée en vue de l'implantation effective de l'automatisme. Le risque d'erreur lors du passage d'une description validée par MAS à une réalisation définitive se trouve alors réduit.

En parallèle à cette recherche, nous avons participé aux travaux de la commission du groupe "Systèmes Logiques" de l'AFCET qui a défini le GRAFCET (Juin 75 - Août 77). Les travaux de cette commission avaient pour objectif d'analyser les besoins créés par la représentation des spécifications du cahier des charges d'un automatisme logique et de réunir dans le modèle proposé les concepts qui permettent de répondre au mieux à ces besoins.

L'expérimentation et l'analyse des possibilités et limites de ce modèle se situaient tout à fait dans le prolongement logique de nos travaux.

- Tout d'abord, nous avons mis à profit l'expérience acquise lors de l'étude des RdPI pour mettre en évidence certaines ambiguïtés, tant syntaxiques que sémantiques, dans les définitions initiales de ce modèle. Nous avons pensé alors à appliquer la notion de synchronisation sur occurrence d'événements pour lui donner une interprétation non ambigüe. Cette interprétation reste conforme aux intentions de la commission AFCET et a l'avantage d'instaurer une grande similitude de fonctionnement avec les RdPI (chap. II).

- Cette expérimentation nous a également amenés à nous intéresser aux modalités d'emploi de cet outil et, dans le même objectif, à définir une méthodologie de structuration et de hiérarchisation de la présentation du cahier des charges d'un automatisme complexe. Nous avons développé dans le chapitre I les lignes directrices d'une telle méthodologie.

- Enfin, nous avons dégagé un certain nombre d'extensions simples qui, sans rien remettre en cause des règles de fonctionnement établies, permettent d'étendre le domaine d'utilisation du grafcet à la spécification des automatismes numériques et des systèmes de commande temps réel de façon générale (chap. III). Nous avons montré, en particulier, l'intérêt de ces extensions pour adapter le grafcet aux besoins de la

méthodologie proposée.

Le grafcet, un modèle de plus, pourquoi faire ? Cette question a souvent alimenté des débats dans des journées d'étude nationales et dans des revues industrielles, entre les promoteurs du grafcet et les partisans de modèles issus plus directement des RdP ; ces derniers tirant argument des possibilités d'analyse et de vérification qu'offrent les RdP et dont l'"extrapolation" au grafcet apparaît difficile.

Nous pensons avoir apporté, dans le chapitre VII, une réponse claire à cette question en comparant les deux modèles une fois formalisés, et en précisant leurs objectifs.

Le grafcet se veut être, avant tout, un outil de dialogue entre spécificateur et concepteur. Il autorise, à ces fins, des facilités intéressantes qui permettent au spécificateur d'exprimer de façon directe et claire ses intentions. Les extensions que nous avons proposé de lui apporter contribuent à renforcer ces facilités en réduisant le plus possible les risques de surspécification qui peuvent résulter des difficultés de manipulation de cet outil ou de la limitation de ses primitives. Les spécifications qui utiliseraient certaines de ces extensions ne seront pas "analysables" de façon systématique ni directement "réalisables" telles qu'elles, mais ceci est un souci secondaire pour le spécificateur. Ce qui importe est que le cahier des charges soit précis et reflète le plus fidèlement possible les intentions du spécificateur, tout en laissant au concepteur le maximum de latitude dans les choix de la réalisation.

Le modèle RdPI se veut être, par contre, un outil de description de solutions, de simulation et d'implantation. Les restrictions de ses primitives par rapport au grafcet—nous avons montré que ces restrictions sont davantage d'ordre pratique que théorique—visent à maintenir le plus de possibilités d'analyse et de vérification et offrir des moyens de contrôle en ligne lors de l'exploitation même du système réalisé.

On aboutit ainsi à mettre à la disposition des automaticiens une véritable gamme d'outils compatibles qui permettent de couvrir l'ensemble des étapes de la conception d'un automatisme, depuis l'étude de sa définition jusqu'à sa réalisation.

De l'avis des industriels avec qui nous avons discuté de ce travail, l'approche est séduisante. Mais on sait bien que cela n'est en général pas suffisant pour convaincre l'industrie d'introduire des méthodes nouvelles dans ses bureaux d'études.

L'expérience avec divers partenaires nous montre que cette conviction s'instaure progressivement, à mesure que l'on met en oeuvre ces méthodes, avec eux, sur des applications qui leur sont propres [MOA.2.79] [MOA.3.79] [DAV.79]. C'est notamment le cas actuellement avec la régie RENAULT où cette recherche est mise à l'épreuve sur divers problèmes posés [BUG.81] [CAV.81].

Les prolongements de ce travail nous semblent devoir porter sur quatre axes complémentaires :

1. Une étude plus approfondie des méthodes et critères qui permettent d'appréhender une conception progressive et sûre des automatismes complexes.

2. Des investigations en vue d'améliorer les possibilités d'analyse et de vérification des systèmes décrits par les RdPI.

3. La définition et la réalisation d'une version industrielle du système MAS en incluant le maximum de facilités de mise en oeuvre. En particulier, les extensions permettant la prise en compte de modèles type files d'attente doivent être approfondies.

4. L'étude de méthodes d'implantation qui permettent le passage quasi-systématique de descriptions en termes de RdPI à des réalisations dans diverses technologies câblées, microprogrammées et programmées, en fonction de la complexité de l'automatisme. Ces méthodes devront alors tenir compte des critères de sûreté de fonctionnement et d'aide au diagnostic souhaités de ces produits.

BIBLIOGRAPHIE

[ABR.78] ABRIAL J.R. : Z : A specification language, IFIP Congress, TOKYO, 1978.

[AGE.73] AGERWALA T., FLYNN M. : Comments on capabilities, limitations and correctness of Petri nets, First Annual Symp. on Comp. Architecture, TAMPA, Floride, 1973, pp. 81-86.

[AND.81] ANDRE C. : Systèmes à évolutions parallèles : Modélisation par réseaux de Petri à capacité et analyse par abstraction, Thèse ès-Sciences, Univ. de NICE, Février 1981.

[AZE.77] AZEMA P., DIAZ M. : Checking experiments for concurrent systems, Proc. FTCS-7, p. 206, Juin 1977.

[BAE.74] BAER.J.L. : Models for the Design, Simulation and Performance of Distributed-Function Architecture, Computer, Mars 1974, pp. 25-30.

[BAR.75] BARBACCI M.R. : A comparison of register transfer languages for describing computers and digital systems, IEEE Trans. on Computers, Vol. C-24, n° 2, Février 1975.

[BER.76] BERTHELOT G., ROUCAIROL G. : Reduction of Petri nets, Math. Found. of Comp. Sci., GDANSK, Pologne, Septembre 1976.

[BER.78] BERTHELOT G. : Vérification des réseaux de Petri, Thèse 3e cycle, Université de PARIS VI, Janvier 1978.

[BER.79] BERTHOMIEU B. : Analyse structurelle des réseaux de PETRI : Méthodes et outils, Thèse Doc. Ingénieur, Université Paul Sabatier, TOULOUSE, Septembre 1979.

[BES.75] BEST E., SCHMID H.A. : Systems of open paths in Petri nets, Notes in Comp. Sci., n° 32, Springer Verlag, BERLIN, 1975.

[BLA.76] BLANCHARD M., CAVARROC J.C., GILLON J., THUILLIER G. : Conception modulaire d'automatismes séquentiels asynchrones, DERA - Télémécanique Electrique, Rapport DGRST n° 71-7-2912-01, Janvier 1976.

[BLA.79] BLANCHARD M. : Automatismes logiques : Grafcet ou réseaux de PETRI ?, Le Nouvel Automatisme, Mai 1979.

[BLA.2.79] BLANCHARD M. : Comprendre, maîtriser et appliquer le GRAFCET, Editions CEPADUES, 1979.

[BUG.81] BUGAUD J.P., DOLLE F., GUERIN P., GOUTIERRE F., MOALLA M. : Expérimentation du grafcet à la spécification de l'automatisme de commande d'une machine d'assemblage, Rapport interne DAST-RENAULT, Avril 1981.

[BYR.75] BYRN W.H. : Sequential processes, deadlocks and semaphore primitives, TR 7-75, HARVARD University, 1975.

[CAP.78] CAPLAIN M. : Langage de spécification, Thèse ès-Sciences, USMG-INPG, GRENOBLE, Mai 1978.

[CAP.2.78] CAPLAIN M., MOALLA M. : Rapports de travaux concernant la conception et la réalisation d'un simulateur de réseaux de PETRI Interprétés, Contrat DRET - SINTRA - ENSIMAG, n° 77/34/302/00480/7501, 1978.

[CAS.80] CASPI P., HALBWACHS N., MOALLA M. : Approche comportementale pour la spécification des systèmes temps réel, Journées d'étude AFCET : la spécification des problèmes, des programmes et des systèmes, TOULOUSE, Octobre 1980. (texte paru dans le bulletin GLOBULE de l'AFCET, n° 2, 1981).

[CAV.81] CAVANNA B., DOLLE F., MOALLA M. : Outils C.A.O. pour l'analyse, la spécification et la réalisation de l'automatisation d'un équipement de production mécanique, 3èmes Journées Scientifiques et Techniques de la Production Automatisée (ADEPA), TOULOUSE, Juin 1981.

[CHA.79] CHAMBON Ph. : Simulateur de Réseaux de PETRI, 7ème Colloque sur le traitement du signal et de ses applications, NICE, Mai 1979.

[CHE.79] CHEZALVIEL-PRADIN B. : Un outil graphique interactif pour la vérification des systèmes à évolution parallèle décrits par réseaux de PETRI, Thèse Doc. Ingénieur, Université Paul Sabatier, TOULOUSE, Décembre 1979.

[COU.77] COURTOIS. P.J., GEORGES J. : On starvation prevention, RAIRO Informatique, Vol. 11, n° 2, 1977, pp. 127-141.

[COU.78] COUSOT P. : Méthodes itératives de construction et d'approximation de points fixes d'opérateurs monotones sur un treillis, analyse sémantique des programmes, Thèse ès-Sciences, USMG-INPG, GRENOBLE, Mars 1978.

[COU.80] COUSOT P., COUSOT R. : Semantic analysis of communicating sequential processes, ICALP, Lecture Notes in Comp. Science, Vol. 85, Juillet 1980.

[DAH.66] DAHL O.J., NYGAARD K. : SIMULA : An Algol Based Simulation Language, Comm. ACM, Vol. 9, Sept. 1966.

[DAV.79] DAVID R., MOALLA M., SAUCIER G., SILVA M. : Conception d'automatismes logiques répartis - Outils et mise en oeuvre, Rapport de contrat DGRST, n° 77-7-0646, Octobre 1979.

[DJI.71] DIJKSTRA E.W. : Hierarchical ordering of sequentiel processes, Acta informatica, Vol. 1, 1971, pp. 115-138.

[DUK.71] DUKE K.A., SCHNURMANN H.D., WILSON T.I. : System Validation by Three-Level Modeling Synthesis, IBM J. RES. DEVELOP., Mars 1971, pp. 166-174.

[FLO.78] FLORIN G., NATKIN S. : Evaluation des performances d'un protocole de communication à l'aide des réseaux de Petri et des processus stochastiques, Journées AFCET : Multiprocesseurs et multi-ordinateurs en temps réel, PARIS, Mai 1978.

[GAR.75] GARDNER R.I. : A methodology for Digital System Design based on Structural and Functional Modeling, UCLA-ENG - 7488, Janvier 1975.

[GHO.77] GHOSH S. : Some comments on timed Petri nets, Journées d'étude AFCET : Réseaux de Petri, PARIS, Mars 1977, pp. 151-163.

[GRA.77] Rapport final de la commission AFCET : Normalisation de la représentation du cahier des charges d'un automatisme logique, Août 1976. Publié intégralement dans la revue Automatique et Informatique Industrielles, numéros 61-62, Nov.-Déc. 1977.

[GRA.79] Document ADEPA : Le GRAFCET, diagramme fonctionnel des automatismes séquentiels, Avril 1979.

[HAC.72] HACK M. : Analysis of production schemata by Petri nets, MAC TR.94, M.I.T., Février 1972.

[HAC.75] HACK M. : Decision problems for Petri nets and vector addition systems, MAC Techn. Memo. 59, M.I.T., Mars 1975.

[HAN.77] HAN Y.W., HEIMERDINGER W.L. : Transformation of a Petri net-like labelled graph to a directed graph for design error detection, Proc. FTCS-7, p. 205, Juin 1977.

[HEN.79] HENINGER K.L. : Specifing software requirements for complex systems : New Techniques and their application, Conf. on Specifications of Reliable Software, IEEE n° 79 CH 1401-9C.

[HOL.68] HOLT A.W. : Information system theory project, Applied Data Research inc., PRINCETON University, New Jersey, Septembre 1968.

[HOL.70] HOLT A.W., COMMONER F. : Events and Conditions, Record of the project MAC Conference on Concurrent Systems and Parallel Computation, A.C.M., NEW YORK, 1970, pp. 3-52.

[IVE.72] IVERSON K.E. : A Programming Language, Ed. J. Willey, NEW YORK, 1972.

[JAC.76] JACK L.A. : Functional faults, Workshop on distributed fault-tolerant systems, PORTO RICO, Novembre 1976.

[KAR.69] KARP R.M., MILLER R.E. : Parallel Program Schemata, JCSS, vol. 3, 1969, pp. 147-195.

[KEL.74] KELLER R.M. : Vector replacement systems : a formalism for modeling asynchronous systems, Techn. Rep. 117, Comp. Sci. Lab., PRINCETON University, Janvier 1974.

[KIV.69] KIVAT P.J., VILLAVUEVA R.E., MARKOWITZ H.M. : The SIMSCRIPT II Programming Language, Prentice Hall, Englewood Cliffs, N.J. 1969.

[KOS.73] KOSARAJU S.R. : Limitations of DIJKSTRA's semaphore primitives and Petri nets, Research Report 25, John Hopkins University, BALTIMORE, May 1973.

[LAU.74] LAUTENBACH K., SCHMID H.A. : Use of Petri nets for proving correctness of concurrent process systems, Proc. IFIP Congress 1974, North-Holland Publ. Co., 1974, pp. 187-191.

[LAU.E.75] LAUER P.E., CAMPBELL R.H. : Formal Semantics of a Class of High-level Primitives for Coordinating Concurrent Processes, Acta Informatica, Vol. 5, 1975, pp. 297-332.

[LAU.T.75] LAUTENBACH K. : Liveness in Petri nets, G.M.D. Int. Rep. ISF-75-02.1, BONN, Juillet 1975.

[LED.76] LE DANOIS P., MOALLA M., SAUCIER G., SIFAKIS J., ZACHARIADES M. : Multilevel Description and Simulation of Parallel Cooperating Processors, Kolloquium über das Parallelismus in der Informatik, ERLANGEN, RFA, Juin 1976.

[LER.76] LEROUDIER J., PARENT M. : Quelques aspects de la modélisation des systèmes informatiques par simulation à événements discrets, RAIRO Informatique, Vol. 10, n° 1, Janvier 1976, pp.5-26.

[LIE.76] LIEN Y.E. : Termination properties of generalized Petri nets, S.I.A.M Journal Comp., Vol. 5, n° 2, Juin 1976, pp. 251-265.

[LIP.76] LIPTON R. : The reachability problems and the boundedness problem for Petri nets are exponential-space hard, TR-62, Dept. Comp. Sci., Yale University, NEW HAVEN, Conn., Janvier 1976.

[MAR.76] MARIN J., ANDRE C., BOERI F. : Conception de systèmes séquentiels totalement autotestables à partir des réseaux de Petri, revue RAIRO-Automatique, Vol. 10, n° 11, Novembre 1976, pp. 23-40.

[MEM.77] MEMMI G. : Semiflows and Invariants. Applications in Petri nets theory, Journées d'étude AFCET : Réseaux de Petri, Mars 1977, PARIS, pp. 145-150.

[MER.78] MERLE D., POTIER D., VERAN M. : Q.N.A.P. A tool for computer performance analysis, Int. Conf. on Performance of Computer Inst., Inst., North-Holland, 1978.

[MER.76] MERLIN P.M., FARBER D.J. : Recoverability of communication protocols. Implications of a theoretical study, IEEE Trans. Comm., Vol. COM-24, Septembre 1976, pp. 1036-1043.

[MOA.75] MOALLA M., SIFAKIS J., ZACHARIADES M. : Un langage d'aide à la conception et à la simulation de systèmes complexes, Rapport de recherche IMAG, n° 16, Oct. 1975.

[MOA.76] MOALLA M., SAUCIER G., SIFAKIS J., ZACHARIADES M. : A design Tool for the Multilevel Description and Simulation of Systems of Interconnected Modules, 3rd Annual Symp. on Comp. Architecture, TAMPA (USA), Janvier 1976.

[MOA.2.76] MOALLA M., SIFAKIS J., ZACHARIADES M. : M.A.S, un outil d'aide à la Description et à la Conception des Automatismes Logiques, Colloque AFCET : Automatismes Logiques, Recherches et Applications Industrielles, PARIS, Déc. 1976.

[MOA.3.76] MOALLA M. : L'approche fonctionnelle dans la vérification des systèmes informatiques - Proposition d'un ensemble de méthodologies, Thèse Doc. Ingénieur, INPG-ENSIMAG, GRENOBLE, Déc. 1976.

[MOA.77] MOALLA M., SIFAKIS J. : A Design Methodology for Complex Logical Systems, 2nd Int. Symp. on Discrete Systems, DRESDEN, Mars 1977.

[MOA.78] MOALLA M., PULOU J., SIFAKIS J. : Synchronized Petri Nets : A model for the description of non-autonomous systems, Mathematical Foundations of Computer Sciences, Ed. Springer Verlag, 1978, pp. 374-383. (Aussi : Réseaux de Petri Synchronisés, RAIRO-Automatique, Vol. 12, n° 2, 1978, pp. 103-130).

[MOA.2.78] MOALLA M., SIFAKIS J., SILVA M. : A la recherche d'une méthodologie de conception sûre des automatismes logiques basée sur l'utilisation des réseaux de Petri, Rapport de recherche IMAG, n° 138, Oct. 1978. (Texte reproduit dans la monographie d'informatique de l'AFCET : sûreté de fonctionnement des systèmes informatiques, Ed. Hommes et Techniques, 1980).

[MOA.79] MOALLA M. : Modèle de présentation du cahier des charges d'un automatisme complexe, Rapport de recherche IMAG, n° 168, GRENOBLE, Septembre 1979.

[MOA.2.79] MOALLA M. : Spécification de réalisation de la centrale de sécurité CALOR 40.11, Doc. OPTION, réf. 78-12-158.041-02, Mai 1979.

[MOA.3.79] MOALLA M. : Spécification de réalisation du programmateur de four CIT-ALCATEL, Doc. OPTION, réf. 79-11-046, Décembre 1979.

[MOA.80] MOALLA M. : Modélisation et analyse du système de masticage de carrosseries, Note interne DAST-RENAULT, Août 1980.

[MOA.81] MOALLA M., DAVID R. : Extension du grafcet pour la représentation de systèmes temps réel complexes, Revue RAIRO-Automatique, Vol. 15, n° 2, 1981.

[PAT.70] PATIL S.S : Coordination of asynchronous events, PhD Thesis, Dept. Electrical Engineering, MIT, CAMBRIDGE, USA, Mai 1970.

[PET.62] PETRI C.A. : Kommunikation mit Automaten, Schriften des Rheinisch-Westfalischen Institutes für instrumentelle Mathematik an der Universitat Bonn, BONN, 1962 (thèse traduite en anglais dans Communication with automata, Tech. Rep. n° RADC-TR-65-337, Vol. 1, Rome Air Develop. Center, Griffis Air Force Base, NEW YORK, Janvier 1966.

[PET.72] PETERSON W.W, WELDON E.J. Jr : Error Correcting Codes, The M.I.T. Press, CAMBRIDGE, Mass., 1972.

[PET.75] PETRI C.A. : Interpretation of net theory, Interner Bericht 75-07, G.M.D., BONN, Juin 1975.

[PET.77] PETERSON J.L. : Petri Nets, ACM Comp. Surveys, Vol. 9, n° 3, Septembre 1977, pp. 223-251.

[PRU.74] PRUNET R., DUMAS J.M. : Introduction à la modélisation naturelle des structures de commande : l'organiphase, Revue RAIRO-AUTOMATIQUE, Juillet 1974, pp. 45-75.

[PUL.79] PULOU J. : Expression de la synchronisation dans les systèmes informatiques et conception d'architectures tolérant les pannes, Thèse Doc. Ingénieur, INPG-ENSIMAG, GRENOBLE, Septembre 1979.

[RAM.73] RAMCHANDANI C. : Analysis of asynchronous concurrent systems by timed Petri nets, PhD. Thesis, M.I.T., Septembre 1973.

[RAM.75] RAMOS NIEMBRO G. : Contribution à l'étude de la commande séquentielle des procédés complexes, Thèse 3ième cycle, INPG-LAG, GRENOBLE, Septembre 1975.

[SCH.74] SCHRIBER T.J. : Simulation using GPSS, John Wiley and Sons, 1974.

[SIF.74] SIFAKIS J. : Modèles temporels des systèmes logiques, Thèse Doc. Ingénieur, USMG-INPG, GRENOBLE, Mars 1974.

[SIF.77] SIFAKIS J. : Use of Petri nets for performance evaluation, Measuring, Modeling and Evaluating Computer Systems, North-Holland Publ. Co., 1977, pp. 75-93.

[SIF.2.77] SIFAKIS J. : Homomorphisms of Petri nets - Applications to the realization of fault tolerant systems, Rapport de recherche IMAG, n° 90, GRENOBLE, Octobre 1977.

[SIF.78] SIFAKIS J. : Structural properties of Petri nets, 7th Symp. on Math. Foundations of Computer Sci., ZAKOPANE, Pologne, Septembre 1978.

[SIF.79] SIFAKIS J. : Le contrôle des systèmes asynchrones : concepts, propriétés, analyse statique, Thèse ès-Sciences, USMG-INPG, GRENOBLE, Juin 1979.

[SUR.76] Mémoire de définition du projet pilote "Sûreté de fonctionnement des systèmes informatiques (SURF)", Janvier 1976.

[SZL.77] SZLANKO J. : Petri nets for proving some correctness properties of parallel programs, IFAC-IFIP Workshop on Real-time Programming, EINDHOVEN, Juin 1977.

[VAL.76] VALETTE R. : Sur la description, l'analyse et la validation des systèmes de commande parallèles, Thèse ès-Sciences, Université Paul Sabatier, TOULOUSE, Novembre 1976.

[VAL.78] VALETTE R. : Etude comparative de deux outils de représentation : Grafcet et réseau de Petri, Le Nouvel Automatisme, Décembre 1978.

[WIR.76] WIRTH N. : Algorithms + Data structures = Programs, Prentice Hall, Englewood Cliffs, N.J., 1976.

[ZAC.77] ZACHARIADES M. : MAS : Réalisation d'un langage d'aide à la description et à la conception de systèmes logiques, Thèse 3ème cycle, Univ. de GRENOBLE, Sept. 1977.

- INDEX -

GRAFCET : Définitions de base, fonctionnement

actions associées à des étapes II-9
action à niveau II-10 et II-18
action impulsionnelle II-10 et II-18
étape II-8
événement toujours présent II-22 et II-29
franchissement d'une transition II-8
franchissement simultané sur occurrence d'événement II-10
nouvelle activation d'une étape II-10
situation II-8
situation stable II-28
réceptivités associées à des transitions II-9
transition II-8
validation d'une transition II-8

EXTENSIONS DU GRAFCET

action de calcul II-25
action impulsionnelle différée III-5 et III-8
échéance limite de franchissement d'une transition III-5 et III-8
macroétape III-11
parallélisme simplement autorisé III-2
parallélisme obligatoire III-2
pseudo-macroétape III-13
fonction III-15

RESEAUX DE PETRI AUTONOMES

graphe d'états IV-4
graphe de transitions IV-4
marquages d'un RdP IV-4
opération de mise à feu de transition IV-5
$^{\bullet}p$ = *ensemble des transitions d'entrée de p* IV-3
$p^{\bullet}$ = *ensemble des transitions de sortie de p* IV-3

RdP borné pour un marquage IV-6
RdP étiqueté IV-10
RdP persistant pour un marquage IV-7
RdP sauf pour un marquage IV-6
RdP vivant pour un marquage IV-7
réseau de Petri à Arcs Inhibiteurs (RdP7) IV-8
réseau de Petri Généralisé (RdPG) IV-9
réseau de Petri libre choix IV-3
réseau de Petri ordinaire (RdP) IV-3
réseau de Petri à Priorités (RdPP) IV-9
réseau de Petri à Priorité Elémentaire (RdPPE) IV-9
séquence de simulation IV-5
séquence de mises à feu IV-5
$^{\bullet}t$ = *ensemble des places d'entrée de t* IV-3
$t^{\bullet}$ = *ensemble des places de sortie de t* IV-3

RESEAUX DE PETRI NON-AUTONOMES

environnement d'un RdPS V-5
environnement le moins contraignant (LL) V-5
environnement le plus strict (LS) V-6
événement V-1
événements compatibles V-5
événement interne à un système V-1
événement externe à un système V-1
marquage stable V-3
occurrence d'événements V-1
RdP Synchronisé (RdPS) V-2
RdPS borné pour un marquage V-7
RdP Etiqueté Synchronisé (RdPES) V-9
RdP Interprété (RdPI) V-17
RdPI déterminé V-21
RdPI déterministe V-21
RdPS k-prompt pour un marquage et un environnement donné V-6
RdPS persistant pour un marquage V-9
RdPS prompt pour un marquage et un environnement donné V-6
RdPS vivant pour un marquage V-8

RdP Temporisé (RdPT)	V-14
RdP Temporisé Synchronisé (RdPTS)	V-12
RdP totalement synchronisé	V-3
relation d'équivalence entre RdPS	V-10
relation de réalisation entre RdPS	V-10
séquence de simulation complète	V-2
séquence de simulation complète instantanée	V-13
système autonome	V-1
système non-autonome	V-1
tir non effectif	V-3
tir itéré sur occurrence d'événements	V-3 et V-13
tir sur occurrence d'événements	V-3 et V-13
transition réceptive à un ensemble d'événements	V-2

AUTORISATION DE SOUTENANCE

Vu les dispositions de l'article 5 de l'arrêté du 16 avril 1974

Vu les rapports de présentation de :

- Madame SAUCIER
- Monsieur COSTES
- Monsieur DAVID

Monsieur M O A L L A Mohamed

est autorisé à présenter une thèse en soutenance pour l'obtention du grade de DOCTEUR D'ETAT ES SCIENCES.

Fait à Grenoble le 26 juin 1981

Le Président de l'U.S.M.G.

UNIVERSITÉ SCIENTIFIQUE ET MÉDICALE DE GRENOBLE

J.J. PAYAN

Le Président de l'I.N.P.G.

B. BLOCH
Président
de l'Institut National Polytechnique
de Grenoble