Mémoire d'Habilitation à Diriger des Recherches

Spécialité Informatique

Présenté à
l'Université Toulouse 1
par

Janette Cardoso

Les aspects temporels qualitatifs et quantitatifs dans les systèmes embarqués

---

Soutenance prévue le 6 juillet 2007 devant la commission d'examen composée de :

Rapporteurs

Jean-Pierre Elloy, Professeur Ecole Centrale de Nantes, IRCCyN

Laurent Foulloy, Professeur Université Savoie

Manuel Silva, Professeur Universidad de Zaragoza, Espagne

Examinateurs

Didier Dubois, Directeur de recherche IRIT-CNRS

Christophe Sibertin-Blanc, Professeur UT1-Université des Sciences Sociales

Frédéric Thivet, Professeur SUPAERO

Robert Valette, Directeur de Recherches LAAS-CNRS

Table des matières

Première partie

Résumé des activités

Chapitre 1

Curriculum Vitæ

Nom :	Mme CARDOSO, Janette
Née le :	2 juillet 1958, à Araquari, Brésil
Mère de :	2 enfants : Thomas, né le 17.7.88 et Aïda, née le 16.10.98
Nationalité :	Française
Adresse personnelle :	33e, chemin de Pécettes, 31520 Ramonville téléphones : 0561 734 452, 0687 675 696
Adresse professionnelle :	SUPAERO 10, av. Édouard Belin, 31055 Toulouse e-mail : cardoso@supaero.fr téléphone : 0562 178 092
Fonction :	Professeur

Titres universitaires

- Doctorat : *Sur les réseaux de Petri avec Marquages Flous*, Université Paul Sabatier/LAAS, Toulouse, octobre 1990. Directeur : R. Valette. Jury : M. Courvoisier, R. David, D. Dubois, J.-C. Gentina, C. Sibertin-Blanc, R. Valette.
- D.E.A. : *Modélisation de la commande et de la surveillance d'un atelier flexible par réseaux de Petri*, D.E.A. Automatique, Informatique Industrielle et Traitement du Signal, Université Paul Sabatier/LAAS, Toulouse, juin 1987. Responsable : M. Courvoisier.
- Master of Science : *Commande numérique utilisant un micro-ordinateur de 16 bits avec arithmétique de point fixe*, Université Fédérale de Santa Catarina, 1984, Brésil.
- Ingénieur de l'Université Fédérale de Santa Catarina (UFSC), 1980, Brésil.

Emplois

- depuis octobre 2006 : détaché comme *Professeur* à Supaéro,
- depuis septembre 1999 : *Maître de Conférences en Informatique* à l'Université Toulouse 1,
- février 1999 à août 1999 : *Chargée de Recherche associée au CNRS*, groupe Outils et Logiciels pour la Communication (OLC) du LAAS,
- 1990 à 1999 : *Maître de Conférences* au Département d'Automatique et d'Informatique et au Département de Génie Électrique de l'Université Fédérale de Santa Catarina (UFSC), dont
  - de septembre 1994 à août 1995 : année sabbatique conjointement au L.A.A.S. et à l'I.R.I.T.,
- 1986 à 1990 : *Chercheur Doctorant* au LAAS/CNRS.
- 1982 à 1986 : *Professeur Assistent* au Département de Génie Électrique de l'UFSC,

Langues : maîtrise orale et écrite en portugais, français et anglais.

Chapitre 2

Résumé des activités

Activités d'enseignement :

Ma carrière d'enseignant chercheur s'est déroulée :

- au Brésil, à l'Université Fédérale de Santa Catarina (UFSC), de 1982 à 1986 en tant que Professeur Assistant (Département de Génie Électrique - DGE) et de 1990 à 1999 en tant que Maître de Conférences (DGE et Département d'Automatique et d'Informatique),
- et en France, à Supaéro depuis octobre 2006, à l'UFR d'Informatique de l'Université Toulouse 1 depuis 1999, ainsi qu'à l'Institut National des Sciences Appliquées de Toulouse où j'ai effectué deux cours.

Mon expérience d'enseignement couvre un spectre large des Systèmes Embarqués allant de l'électronique aux outils informatiques, en passant les réseaux de Petri (RdP) et les représentations et modèles de bases des systèmes dynamiques. Ceci est résumé dans les tableaux ci-dessous :

Matières (**au Brésil**)	Niveaux	Période	Volume
Réseaux de Petri	3^{eme} cycle	90-94, 95-98	20hCM
	2^{eme} cycle	92-94, 95-98	2 × (20hCM+10hTP)
Intelligence Artificielle	3^{eme} cycle	90-94	40hCM
	2^{eme} cycle	93-94	2 × (72hCM+36hTP)
Logiques non classiques	3^{eme} cycle	95-98	20hCM
Mathématiques Discrètes	3^{eme} cycle	96-98	30hCM
Introduction à l'Informatique	2^{eme} cycle	84-86	2 × (30hCM+10hTP)
Génie logiciel	2^{eme} cycle	84-86	2 × (30hCM+10hTP)
Syst. Logique et Combinatoire	2^{eme} cycle	82-86, 90-94	2 × (54hCM+36hTP)
Instrumentation Électronique	2^{eme} cycle	82-84	2 × (30hCM+45hTP)

Matières (en **France**)	Niveaux	Période	Volume
Modèles de systèmes embarqués	3^{eme} année (SUPAERO)	07-08*	15hCM +5TD
Planification et décision	MS Systèmes Embarqués	07-08*	15hCM +5TD
Approche Systèmes : concept et exemples	2^{eme} année (SUPAERO)	06-07	20hTD
Modélisation Analyse SED	5^{eme} année et DEA (INSAT)	94-95	15hCM+6hTP
Modélisation de la Dynamique	DESS IGSI (UT1)	99-04	6hCM+3hTP
Évaluation de performance	DEA SRIC (INSAT)	03-04	4hCM
Tableur et Programmation	2^{eme} cycle Maîtrise (UT1)	00-06	15hCM+30hTP
	2^{eme} cycle Licence (UT1)	05-06	9hCM+9hTP
Systèmes d'Exploitation	2^{eme} cycle Licence (UT1)	00-06	12hCM +12hTP
Bases de Données	DESS Banque Finance (UT1)	00-03	8hCM+9hTP
	2^{eme} cycle Cyber Maîtrise Droit	04-06	20hCM+20hTP
	2^{eme} cycle (UT1)	99-04	30hTP

* *En préparation.*

Activités de Recherche :

- Domaines : Conception de logiciels pour les systèmes embarqués : description, vérification et implémentation. Spécification temporelles des systèmes, traitement d'informations incomplètes. Logique possibiliste, réseaux de Petri.
- Encadrement et participation à des jurys :
  - 3 co-encadrements de thèses de doctorat,
  - 3 co-encadrements de M2R, 6 encadrements de thèses de *Mestrado* (Master of Science), 1 encadrement de *mastère*,
  - 1 participation à un jury de thèse,
  - 11 participations à des jurys de thèses de *Mestrado* et
  - 3 participations à des jurys de *Qualification* (voir section 6).
- Publications :
  - 1 livre publié,
  - 1 livre co-édité,
  - 6 articles de revue,
  - 6 contributions à des ouvrages,
  - 28 articles dans des congrès internationaux avec comité de lecture et actes à diffusion large,
  - 3 articles dans des workshops internationaux,
  - 5 articles dans des congrès francophones, 4 dans des congrès français et 8 dans des congrès brésiliens.
- Participation à des projets de recherche : 3 projets internationaux (ALFA-AIR, CAPES/DAAD et CAPES/COFECUB) ; 2 projets Féria ; 4 projets formellement soutenus par le pouvoir public brésilien.

Collaborations, animation et responsabilités :

- Collaborations industrielles : participation à 3 projets :
  - Commande floue d'un hélicoptère (Gyron Technology),
  - Spécification par réseaux de Petri d'un système de contrôle en pétrochimie (Petrobrás, Brésil),
  - Gestion d'une cellule de production industrielle robotisée (DGM/UFSC e Weg Moteurs S.A.) ;
- Animation scientifique : conférencier invité à un workshop et à une conférence ; membre de comité de programme de 2 conférences, lecteur de plusieurs revues et conférences ; membre actif des sociétés savantes ;
- Responsabilités administratives à Supaero : Responsable du Mastère *Systèmes Embarqués* proposé par Supaéro et l'ENSHEEIT (première accréditation accordée par la Conférences de Grandes Écoles pour l'année 2007/2008) ;
- Responsabilités administratives à l'UT1 – Université Toulouse 1 : membre du conseil de l'UFR d'Informatique ; membre de la Commission de Spécialistes d'Informatique ; responsable du certificat CIAD (Certificat d'Informatique Appliqué au Droit) ;
- Responsabilités administratives à l'UFSC – Université Fédérale de Santa Catarina : coordinateur-adjoint de la formation d'Ingénieur en Automatique et Informatique ; responsable du Laboratoire d'Electronique Numérique ; membre de la commission permanente pour la Formation Doctorale en Informatique ; membre de la commission de recrutement de maîtres de conférences de la formation d'Ingénieur.

Chapitre 3

Formation

3.1 Fonctions occupées

Mes activités d'enseignement se sont déroulées à l'UFSC – Université Fédérale de Santa Catarina (au Brésil), à l'UT1 – Université Toulouse 1 et à SUPAERO – École Nationale Supérieure de l'Aéronautique et de l'Espace comme :

- *Professeur Assistent*[1] au Département de Génie Électrique de l'UFSC dans le cadre de la formation d'Ingénieur Électricien (1982–1986) ;
- *Maître de Conférences* à l'UFSC, de 1990 à 1999 :
  - au Département de Génie Électrique, cadre formation d'Ingénieur en Automatique et Informatique, second cycle,
  - au Département d'Automatique et Informatique, cadre formation d'Ingénieur en Automatique et Informatique, second cycle,
  - à la Formation Doctorale en Automatique et Informatique et Formation Doctorale en Informatique (à partir de 1992), troisième cycle ;
- *Maître de Conférences en Informatique* à l'UT1, de 1999 à septembre 2006 :
  - au niveau Licence (L3) et Master (M1) des différentes formations (AES-Administration Économique et Sociale, Sciences Économiques, Droit), ainsi qu'à la Licence Professionnelle Système Informatiques et Logiciels ;
  - au niveau Master, M2P IGSI-Ingénierie et Gestion des Systèmes d'Information et M2P Banque et Finances Européennes.
- *Professeur* à Supaéro, au département Informatique, dans le cadre de la formation ingénieur en aéronautique et espace, les masters recherche et mastères spécialisés (détaché depuis octobre 2006).

De septembre 1986 à novembre 1990, j'ai bénéficié d'un détachement en vue de la réalisation d'un D.E.A. puis d'un Doctorat au L.A.A.S.. D'août 1994 à septembre 1995, j'ai bénéficié d'une année

[1] Le *Professeur assistent* est recruté sur concours. Le candidat doit posséder au moins un diplôme de thèse de *Mestrado*

sabbatique de recherche, conjointement au L.A.A.S. et à l'I.R.I.T., au cours de laquelle j'ai effectué un enseignement à l'INSAT.

3.2 Enseignements effectués

Mes activités d'enseignements se déclinent en trois volets : les modèles et représentations fondamentales pour les systèmes embarqués, l'électronique – principalement dans le cadre des formations d'ingénieurs –, et les outils informatiques – principalement dans le cadre de l'informatique de gestion. Le contenu des cours que j'ai assuré est détaillé ci-dessous.

3.2.1 Les modèles et représentations fondamentales pour les systèmes embarqués

Mathématiques Discrètes (Formation doctorale) Introduction (relation, alphabet et mots). Automates finis. Langages formels et automates. Langages et expressions régulières. Composition d'automates. Minimisation d'automates. Logique des prédicats. Logique du première ordre. Théorie des modèles et sémantique. Démonstrateur de théorèmes. Méthodes de résolution. Programmation en logique. (30h CM)

Modélisation et Analyse de Systèmes à Événements Discrets (INSAT) Introduction. Définitions. Analyse de propriétés. Les données et le temps. Modélisation et conception. Mise en œuvre et simulation. (15h CM + 6h TP). 5^{eme} année/ D.E.A. d'Automatique et Informatique Industrielle.

Réseaux de Petri (Formation doctorale) Introduction. Définitions. Analyse de propriétés. Interprétation. Réseaux de haut niveau. Réseaux de Petri et temps. Réseaux de Petri et logiques non classiques. Systèmes hybrides. (20 h CM + 10h TP)

Réseaux de Petri (Formation d'ingénieur) Introduction. Définitions. Analyse de propriétés. Interprétation. Réseaux de Petri et temps. (20 h CM + 10h TP)

Réseau de Petri temporel possibiliste (INSAT) Introduction à la théorie de possibilités. Informations temporelles floues dans un réseau de Petri. Comparaison avec le réseau de Petri stochastique. Applications.(4h CM). Module du cours *Évaluation de Performances*, 5^{eme} année/D.E.A. Systèmes Automatiques, parcours SRIC- Systèmes et Réseaux Informatiques Critiques.

Approche Systèmes : concept et exemples Importance de l'approche systèmes. La construction du cahier de charges. Analyse fonctionnelle interne. Application sur une étude de cas réel (20h TD).

Modèles de systèmes embarqués : Modèles discrets, modèles hybrides (Formation d'ingénieur) Motivation. Modèles pour les SED (automates, réseaux de Petri). Rappel pour la modélisation de systèmes continus. Modélisation conjointe des parties continue et discrète d'un systèmes hybride. Dérivation des lois de commande. (14h CM + 6h TP). Nouvelle discipline à partir de septembre 2007.

Planification et décision (Formation *mastère*) Modèle d'états, d'actions et de transitions (déterministe ou aléatoire). Planification d'action. Décision séquentielle dans l'incertain : représentation de l'incertitude. (20h CM). Nouvelle discipline à partir de septembre 2007 en collaboration avec un collègue.

Modélisation de la Dynamique (M2P) Introduction. Définitions. Analyse de propriétés. Modélisation et conception. Relation avec les diagrammes de classe et diagrammes de séquence UML. (15h CM + 6h TP). M2P IGSI – Ingénierie et Gestion des Systèmes d'Information.

Logiques non-classique (Formation doctorale) Logique floue. Logique linéaire. Logique modale. Logique non monotone. (20h CM).

Intelligence Artificielle (Formation doctorale) Introduction. Résolution de problèmes (recherche heuristique, réduction des problèmes, jeux). Représentation de la connaissance (systèmes de production, logique, frames, réseaux sémantiques). Systèmes experts (composants, méthodes d'inférence, exemples d'application, outils de développement de systèmes experts). Logique floue. Aspects temporels. (40h CM)

Introduction à l'Intelligence Artificielle (Formation d'ingénieur) Introduction. Représentation par espaces d'états et algorithmes de recherche (recherche heuristique, réduction des problèmes, jeux). Représentation de la connaissance (systèmes de production, logiques, frames, réseaux sémantiques). Systèmes experts (historique, exemples d'application, outils de développement). Langage LISP. (72h CM + 36h de TP)

Introduction à l'Informatique (Formation d'ingénieur) Systèmes d'exploitation configurables, systèmes d'exploitation temps réel, systèmes multitâches. (30h CM + 10h TP)

Génie logiciel (Formation d'ingénieur) Cycle de vie. Spécification (SADT, SART, DFD). Conception (HOOD). Vérification et validation. Gestion de projet. Assurance qualité. Langage Pascal. (30h CM + 10h TP)

Pour tous ces enseignements j'ai assuré les cours magistraux et les travaux pratiques, ainsi que la responsabilité d'organisation des examens.

J'ai participé à l'organisation d'une formation continue en Automatique Industrielle, troisième cycle (productique, automatique, informatique industrielle, etc.), d'un total de 405h, destinée à des ingénieurs de petites et moyennes entreprises d'une région riche en industries métallurgiques. J'ai en particulier réalisé un support de cours sur la commande des systèmes de production.

3.2.2 Électronique

Électronique Numérique Algèbre de Boole. Circuits Combinatoires (fonctions logiques, simplification et minimisation de fonctions logiques, synthèse). Circuits séquentiels (notion d'état, bascules, analyse et synthèse de circuits séquentiels). Familles logiques. Mémoires. (54h CM + 36h TP+ mini projets)

Instrumentation Électroniques Instruments électroniques analogiques (générateur de signal, oscilloscope, amplificateur pour instruments, enregistreur). Instruments électroniques numériques (conversion analogique-numérique, voltmètre, fréquencemètre, comparaison entre instruments analogiques et numériques). Transducteurs électriques. (30h CM + 45h TP)

Pour tous ces enseignements j'ai assuré les cours magistraux et les travaux pratiques, ainsi que la responsabilité d'organisation des examens.

3.2.3 Outils Informatiques

Systèmes d'Exploitation (UT1 : Licence Professionnelle) Introduction. Vue générale d'un système d'exploitation (SE). Gestion de processus. Système de fichiers. Protection et sécurité. Win-

dows et Linux. (12hCM + 12hTP)

Tableur et Programmation (UT1 : L3, M1) Outils d'analyse et simulation à l'aide d'un tableur (fonctions avancés : étude de sensibilité, recherche d'objectif, optimisation, etc.). Méthode de développement de programmes avec le langage de macro-commandes d'un tableur.

Base de Données (UT1 : L3, M1, M2P) Introduction aux SGBD. Le modèle Entité-Association, le modèle Relationnel et le Modèle Physique. Les Requêtes en SQL et en mode graphique (ACCESS).

Ces trois cours ont été effectués à l'UT1. Les programmes ont été adaptés selon le volume horaire et le public. J'ai aussi assuré les cours magistraux et les travaux pratiques, ainsi que la responsabilité d'organisation des examens de tous ces enseignements, à par le cours de Base de Données en AES–Administration Économique et Sociale.

Le cours de Tableur a été donné aux formations suivantes : niveau M1 es Sciences Économiques (15hCM + 30hTP) et Licence Professionnelle RTIC – Responsable des Technologies de l'Information et de la Communication en Petites et Moyennes Organisations (9hCM + 9hTP).

Le cours de Base de Données a été donné aux formations suivantes : Licence en AES–Administration Économique et Sociale (30hTP), M2P Banque et Finances Européennes (8hCM + 9TP) et cyber maîtrise de Droit au Centre Universitaire de Formation et de Recherche Jean-François Champollion à Albi (20hCM + 20hTP).

3.3 Ouvrage et support de cours

– R. Valette, J. Cardoso : Réseaux de Petri (en portugais), 212 pages, 1997, Editora da UFSC (Université Fédérale de Santa Catarina), ISBN 85-328-0095-5.

  Sommaire : Partie I : Le modèle de base Vocabulaire et concepts de base. Définitions. Analyse des propriétés. Partie II : Les données, le temps et l'environnement externe Réseaux interprétés. Les réseaux de Petri de haut niveau. Les réseaux de Petri et la représentation du temps. Réseaux de Petri, logiques non classiques et systèmes hybrides.

  Ce livre est un des principaux ouvrages de références sur les réseaux de Petri au Brésil. Avant son édition, la version polycopié était utilisée depuis 1991 dans les formations doctorales en Automatique et Informatique, en Informatique et en Génie Mécanique de l'UFSC.

– J'ai fait des polycopiés pour les cours de Électronique Numérique, Instrumentation Électronique, Mathématiques Discretes, Introduction à l'Intelligence Artificielle, Modélisation de la Dynamique, Systèmes d'Exploitation et Tableur et Programmation. J'élabore actuellement les polycopiés pour les cours *Modèles de systèmes embarqués : modèles discrets, modèles hybrides* et *Planification et décision*.

3.4 Encadrements

Je co-encadre actuellement 1 thèse de Doctorat et j'ai co-encadré 2 autres. En plus de l'encadrement de stagiaires élèves ingénieurs et élèves M2P, j'ai encadré 3 stages M2R EDSYS - École Doctorale Systèmes, 5 thèses de Master of Science[2] aux formations doctorales en Automatique et en Informatique (4) et en Génie Mécanique (1) de l'Université Fédérale de Santa Catarina et j'encadre actuellement un stage mastère spécialisé :

- Luciana Bolan Frigo, *Un outil de conception de cours adaptatifs pour les systèmes tuteurs intelligents*, thèse de doctorat en Informatique (cotutelle entre l'Université Fédérale de Santa Catarina et l'Université Toulouse 1/IRIT). Co-encadré avec G. Bittencourt, 8 janvier 2007.
- Omar Tahir, *Étude des diagrammes dynamiques UML via les réseaux de Petri de haut niveau*, thèse de doctorat en Informatique, Université Toulouse 1/IRIT. Co-encadré avec C. Sibertin-Blanc, 8 décembre 2006.
- Eliane Pozzebon, *Une Architecture multiagent pour supporter lapprentissage en groupe dans un Système Tuteur Intelligent*, thèse de doctorat EDIT en cotutelle entre l'Université Fédérale de Santa Catarina et l'Université Toulouse 1. Co-encadré avec G. Bittencourt *Examen de qualification pour doctorat*, avril 2005 (soutenance prévue pour février 2008).
- Zakaria Saharaoui, *Modélisation et simulation de systèmes hybrides*, projet de fin d'études du mastère Commande et Systèmes embarqués, Supaéro, en cours (soutenance prévue septembre 2007).
- Abdia Hamdani, *Amélioration de l'outil GraphC pour les réseaux de Petri t-temporels.* M2R SAID, parcours Systèmes Industriels. Co-encadré avec R. Valette (LAAS), juin 2006.
- Xiaoyu Mao, *L'outil GraphC : un graphe de classes pour les réseaux de Petri t-temporels préservant les contraintes temporelles quantitatives.* Projet de fin d'études INSAT et M2R Systèmes Automatiques, Informatiques et Décisionnels, parcours SRIC - Systèmes et Réseaux Informatiques Critiques. Co-encadré avec R. Valette (LAAS), septembre 2005.
- Sébastien Cousy, *Les Réseaux de Petri et le traitement de l'imperfection de l'information.* Projet de fin d'études ENSEEIHT et DEA Systèmes Automatiques. Co-encadré avec G. Juanole (LAAS, septembre 2004).
- Adriana Postal, *Un outil d'auteur pour le système tuteur intelligent MathTutor* (en portugais). Thèse de *mestrado* co-encadrée avec M. Bittencourt, Florianópolis, Brésil, avril 2004.
- L.L. Caimi, *Simulateur de réseaux de Petri avec marquage flou* (en portugais). Thèse de *mestrado*, Brésil, février 1998.
- C. Cavalcante, *Système de navigation pour hélicoptère sans pilote utilisant la commande floue)* (en portugais). Thèse de *mestrado*, Brésil, juillet 1994.
- F. A. Soares, *Un environnement intégré d'outils pour la conception de systèmes à événements discrets* (en portugais). Thèse de *mestrado*, Brésil, juillet 1994.
- M.C.L.F. Braga, *Un gestionnaire de cellule flexible de production* (en portugais). Thèse de *mestrado*, Brésil, septembre 1993.

[2]La thèse de *mestrado* (Master of Science) au Brésil comporte une année de cours avec un minimum de 405 heures, et d'une à deux années de travail de recherche (souvent d'un niveau comparable à l'ancienne thèse de 3ème cycle ou de docteur-ingénieur).

Chapitre 4

Recherches

Mes activités de recherche se déroulent depuis octobre 2006 à l'ONERA-Toulouse et se sont déroulées entre juillet 2000 et septembre 2006 dans l'équipe SOC – Systèmes à Objets Coopératifs du thème Sûreté de développement du logiciel à l'IRIT - Institut de Recherches en Informatique de Toulouse. Entre 1999 et 2000 j'étais chercheur associé au LAAS – Laboratoire d'Analyse et d'Architecture des Systèmes. Entre 1982 et 1999, mes activités se sont déroulées au Laboratório de Controle e Microinformática (LCMI) de l'Université Fédérale de Santa Catarina (UFSC) au Brésil.

4.1 Problématique

Mes travaux de recherche ont en commun un même outil de base, les réseaux de Petri, avec des domaines d'application très divers, des systèmes tuteurs intelligents aux systèmes embarqués. Dans le cadre de conception de logiciels pour des systèmes embarqués et des systèmes critiques, autre la spécification dynamique du système que l'on veut piloter, je m'intéresse à la vérification des propriétés telle que l'atteignabilité d'un état redouté, caractérisé par un temps de réponse quantifié.

Les différents aspects abordés dans mes travaux de recherche sont présentés ci-dessous.

Sémantique des diagrammes UML de la dynamique
Ces travaux, fait en collaboration avec C. Sibertin-Blanc, portent sur l'utilisation des réseaux de Pétri pour proposer une sémantique formelle aux diagrammes d'interaction (diagramme de séquence et diagramme de collaboration) et pour étudier leurs relations avec les diagrammes de comportement (diagramme d'activité et diagramme d'état-transition) d'UML. Ces deux types de diagrammes sont en effet en étroite relation, puisque les premiers, expression des besoins fonctionnels, sont la spécification de la partie observable du comportement des composants, alors que les comportements des composants déterminent le fonctionnement du système, notamment les scénarios qu'il est capable d'exécuter [47, 21, 6, 46, 52, 18, 19] (voir avant-projet Féria section 4.2). Co-encadrement d'une thèse de Doctorat, O. Tahir.

Vérification et mise-en-œuvre de systèmes à événements discrets : Les réseaux de Petri permettent de capturer, d'une façon formelle, l'aspect dynamique des systèmes à événements discrets. C'est un modèle formel de représentation, mais, surtout, d'analyse, permettant de vérifier certaines propriétés du système modélisé. J'ai développé un environnement intégré d'outils, à base de réseaux de Petri, pour la vérification, la simulation et la mise-en-œvre de la commande de SED en considérant aussi la structure hiérarchique de tels systèmes [35, 38, 39, 58]. Encadrement de deux dissertations de *Mestrado*, M. Braga et F. Soares.

Des travaux plus récents sur ce volet traitent de la conception des Systèmes Tuteurs Intelligents (STI) basée sur une architecture de Systèmes Multi-Agents distribués (les agents tuteurs) en utilisant les réseaux de Petri à Objets [3, 20, 55, 42, 16]. Ils se situent dans le cadre d'un projet Capes/Cofecub (voir section 4.2) en collaboration avec l'UFSC, Brésil. Une partie de ces travaux s'est inspirée de la notion de protocole de communication comme un composant à part entière dans les systèmes coopératifs [62, 5, 48]. Co-encadrement d'une dissertation de *Mestrado*, A. Postal, et de deux thèses de Doctorat, L. Bolan et E. Pozzebon.

Spécification et vérification de contraintes floues : Le réseaux de Petri, malgré les extensions apportées à leur pouvoir expressif, ne permettent de prendre en compte que des informations certaines et précises. Pour pouvoir représenter et traiter des informations entachées d'incertitude j'ai développé le formalisme de Réseaux de Petri avec marquage flou, formalisme qui combine les réseaux de Petri pour traiter l'aspect dynamique des SED et la théorie des Possibilités pour traiter l'incertitude [9, 8, 12, 13, 27, 22, 23]. J'ai proposé une autre approche utilisant les informations du graphe de marquages pour établir des préférences entre les états possibles d'un système [7, 24, 26, 36]. Encadrement d'une dissertation de *Mestrado*, L. Caimi.

L'utilisation de la logique floue ou possibiliste permet de représenter de façon plus complète le monde réel. Cependant, il est important aussi de pouvoir construire des raisonnements au-dessus des modèles du système physique. La logique classique ne peut pas être directement utilisée. La logique linéaire, avec la notion de consommation de ressources, semble une bonne candidate. Dans un travail conjoint avec B. Pradin-Chézalviel et R. Valette (LAAS), nous avons montré que les marquages imprécis et les séquences de tir imprécises peuvent être représentés par la logique linéaire et surtout que celle-ci permet une caractérisation claire des séquences, ce que ne fait pas la théorie des réseaux de Petri [37, 11, 10, 44, 29, 32, 53].

Spécification et vérification de contraintes temporelles quantitatives imprécises et floues

Pour la vérification de certaines propriétés des systèmes embarqués critiques, on est parfois amené à considérer des scénarii de fonctionnement spécifiques et à analyser très exactement les contraintes temporelles entre les événements de ces scenarii, pour extraire le pire cas. D'autres propriétés impliquent la recherche exhaustive de tous les états d'un système et l'on construit alors un graphe des classes. J'ai proposé, avec R. Valette, un nouveau graphe des classes[1] pour les réseaux de Petri temporels qui permet d'associer à tout chemin de ce graphe l'ensemble des séquences de franchissements effectivement franchissables [4, 43, 45, 50]. Co-encadrement de trois *Master Recherche*, S. Cousy, X. Mao et A. Hamdani.

[1] L'outil GraphC – `http ://graphc.sourceforge.net`

Un point important est l'expressivité alliée à la simplicité des modèles. Dans un système réactif, toutes les évolutions du systèmes ne sont pas connues avec certitude. Le modèle de réseaux de Petri temporel flou permet de traiter des informations dont l'incertitude apparaît sous la forme d'une ignorance partielle (plutôt que de nature aléatoire). Ce type d'information est traité de façon efficace par la théorie de possibilités qui permet d'allier l'expressivité et la simplicité. Cette théorie permet de traiter, au lieu d'un cas, l'ensemble de cas de façon ordonnée : les fenêtres temporelles floues représentent en fait des ensembles emboîtées d'intervalles temporels. Cela est utile dans une phase d'ajustement des paramètres. Par exemple, dans le cas d'une procédure d'exécution à distance, on souhaite trouver la valeur d'un paramètre de façon à pouvoir avoir un compromis entre attendre trop longtemps ou perdre des réponses tardives, sans avoir à tester plusieurs cas possibles. Des extensions des graphes de classe au cas flou ont déjà été faites dans le cas du parallélisme par entrelacement [15, 17].

4.2 Participation à des projets de recherche

Les projets suivants ont porté principalement sur des aspects productiques (CNPq/Conselho National de Pesquisa –*Conseil Nationale de Recherche*) :
- *Logique floue, logique linéaire et réseaux de Petri dans la commande de systèmes*. Projet CNPQ n^o 300845/93-6, août 1993 à juillet 1994 et de septembre 1995 à février 1996 [33, 35, 37] ;
- *Méthodologies et outils pour la commande de procédés et l'automatisation industrielle*, Projet CNPQ/AI n^o 501360/91, août 1991 à juillet 1993 [38, 39], [34, Far93, 59].

Dans le cadre de l'équipe d'Intelligence Artificielle du LCMI, je me suis intéressée à étudier et à analyser les apports possibles entre les SED et les bases de données et de connaissances évolutives. L'idée est d'enrichir les outils et modèles des systèmes à base de données ou de connaissances dynamiques à l'aide de ceux des SED et vice-versa. Les projets suivants ont abordé cette piste de recherche :

- *Programme de Recherches Coordonnées (PRC-GDR) du Centre National de la Recherche Scientifique - CNRS Gestion de l'Evolutif et de l'Incertain dans une Base de Connaissance (GEI)*, Toulouse, août 1994 à août 1995. J'ai utilisé les réseaux de Petri pour décrire le fonctionnement normal et les logiques linéaire et floue pour construire les étapes de localisation et d'identification après la détection de l'anomalie [27, 29, 32, 56, 53].
- *Un environnement pour la spécification formelle de bases de connaissances représentant les systèmes dynamiques*. Projet CNPQ/AI n^o 350794/91-0, mars 1996 à février 1998 [7] [23]-[26] ;
- *Formalismes et outils pour l'automatisation : dès systèmes dynamiques aux bases de connaissances distribuées*. Projet CNPQ/AI n^o 523169/95-7, de mars 1998 à février 2000 ;

J'ai participé au projet *Gestion de l'Evolutif et de l'Incertain dans une Base de Connaissance (GEI)* du programme PRC-GDR dans le cadre de mon année sabbatique conjointement au LAAS et à l'IRIT. C'est en fait à partir de discussions au sein de ce projet que je me suis intéressée aux liens entre les SED et les bases de connaissances.

J'ai également participé, en tant que responsable locale, au projet ALFA dont le but est de favoriser la formation doctorale en Amérique Latine.
- Projet Formation pour l'Amérique Latine ALFA-AIR B1 e B2 (Automation Industrielle Robuste, *Prise en compte de l'incertitude dans les systèmes dynamiques*), avec des universités européennes

(France, Portugal, Espagne, Allemagne et Finlande) et latino-américaines (Brésil, Colombie, Vénézuéla, Argentine) ; de 1996 à 1998.

J'ai proposé un projet de coopération international entre l'équipe Systèmes à Objets Coopératifs de l'IRIT-UT1 et le Laboratoire de Contrôle et MicroInformática de l'UFSC, qui a été renouvellé à 2 reprises :

– *Interaction entre modèles formels pour les systèmes de supervision et les systèmes d'information.* Projet Capes-Cofecub 400/02, janvier 2002 à décembre 2005 entre l'IRIT-UT1 et l'UFSC - Université Fédérale de Santa Catarina, Brésil [3, 20, 55, 42]

Le thème central de ce projet est l'étude de différents formalismes pour traiter les problèmes de modélisation, et plus particulièrement ceux liés aux interactions, dans les systèmes complexes - systèmes de contrôle d'automatismes ou systèmes d'information distribués. Le projet est basé sur les systèmes multiagents - qui intègrent les technologies du génie de logiciel, des systèmes distribués et de l'intelligence artificielle, thèmes dont les deux laboratoires étaient complémentaires. Ce projet a permit l'encadrement de deux thèses en cotutelle et un M2R soutenu au Brésil en 2004.

Dans le cadre de Féria – Fédération de Recherche en Informatique et Automatique, j'ai participé aux projets suivants :

– *Structure des interactions et comportement des composants d'un système* (IRIT, LAAS et ONERA). Coordinateurs : C. Sibertin-Blanc (IRIT), J. Fanchon (LAAS) et P. Michel (ONERA), 2003/2004. Le but était de proposer des procédés pour passer de l'ensemble des diagrammes de séquences d'un cas d'utilisation d'un système au comportement (sous la forme de diagrammes d'états-transitions) de chacun des objets qui participent à ce cas d'utilisation [52, 46].
– *Les réseaux de Petri temporels flous : étude de propriétés et application aux SMA* (IRIT, LAAS). Coordinateurs : J. Cardoso (IRIT) et G. Juanole (LAAS), 2004/2006. Le cadre général de ce projet est l'analyse d'un système dynamique où les informations temporelles sont connues de façon imparfaite [4, 15, 17, 43, 50, 45].

Chapitre 5

Collaborations, animation et responsabilités

5.1 Collaborations Scientifiques

J'ai entretenue de nombreuses collaborations scientifiques principalement avec :
– S. Sandri, du INPE (Instituto National de Pesquisas Espaciais, São Paulo, Brésil) dans le domaine de l'Intelligence Artificielle et plus particulièrement sur la théorie des possibilités ;
– B. Pradin-Chézalviel du groupe OLC du LAAS, dans le domaine des réseaux de communications et l'utilisation de la logique linéaire dans ce contexte ;
– D. Dubois, du groupe d'Intelligence Artificielle de l'IRIT ;
– G. Bittencourt, de l'Université Fédérale de Santa Catarina, Brésil, dans le domaine de l'Intelligence Artificielle et plus particulièrement les systèmes tuteurs intelligents ;
– G. Juanole du groupe OLC du LAAS, pour l'étude comparative entre les réseaux de Petri Stochastiques et les réseaux de Petri temporels flous ;
– R. Valette du groupe OLC du LAAS, dans le domaine de la productique, avec un intérêt pour les liens entre réseaux de Petri et logique, et
– C. Sibertin-Blanc, du groupe Systèmes à Objets Coopératifs de l'IRIT.

5.2 Collaborations Industrielles

J'ai contribué à plusieurs projets industriels, en particulier dans le cadre des collaborations suivantes :
– conception et simulation d'un système de navigation d'un hélicoptère radio-commandé, avec rotor de 2m (utilisé pour l'inspection de lignes électriques), basé sur la commande floue, en collaboration avec l'entreprise Gyron Technologie [30, 34] ;

– conception et mise-en-œuvre d'un environnement intégré pour la spécification de systèmes à base de réseaux de Petri en collaboration avec la PETROBRÁS (Pétrole du Brésil) appliqué à l'automatisation de stations de pompage de pétrole au Nord-est du Brésil [35] ;
– conception et mise-en-œuvre d'un logiciel de gestion d'une cellule de production, utilisant les réseaux de Petri, dans un travail conjoint avec des chercheurs du Département de Génie Mécanique et de l'industrie de fabrication de moteurs Weg Motores S.A.. La cellule est composée d'un robot, d'un tour, d'un micro-mètre laser, d'un magasin d'entrée et d'un magasin de sortie [58].

5.3 Animation scientifique

Conférencier invité :
– 2^{nd} International Workshop on Manufacturing and Petri nets, Toulouse, France, 23 Juin 1997.

– 23^{rd} International Conference on Information Technology Interfaces, ITI 2001, Pula, Croatia, 19-22 Juin 2001.

Lecteur de revues :
– International Journal of Uncertainty ;
– Fuzziness and Knowledge Based Systems ;
– International Journal of Intelligent Control and Systems ;
– IEEE Trans. on Fuzzy System ;
– *Controle e Automação* (revue brésilienne IFAC/SBA (Sociedade Brasileira de Automação) ;

Membre de comités de programmes et/ou lecteur de conférences :
International Fuzzy Systems Association World Congress ; International Workshop Petri Nets and Performance Models - PNPM ; International Conference on Intelligent Systems. Journées Formalisation des Activités Concurrentes ; Congresso Brasileiro de Automática - CBA (Congrès Brésilien d'Automatique, de la SBA) ;

Chairwoman :
Chairwoman de la session Fuzzy Petri nets, au International Fuzzy Systems Association World Congress, IFSA'95 et aux Congrès Brésilien d'Automatique.

5.4 Activités d'administration et autres responsabilités collectives

– Responsable du Mastère *Systèmes Embarqués* proposé par Supaéro et l'ENSHEEIT (première accréditation par la Conférences de Grandes Écoles pour l'année 2007/2008) ;
– Coordinateur du Projet Capes-Cofecub 400/02 entre l'IRIT-UT1 et l'UFSC - Université Fédérale de Santa Catarina, Brésil, de janvier 2002 à décembre 2005 (voir section 4.2) ;
– Responsable du projet Féria-SVF *Les réseaux de Petri temporels flous : étude de propriétés et application aux SMA* (IRIT, LAAS), d'octobre 2004 à septembre 2006, (voir section 4.2) ;

- Membre du conseil de l'UFR d'Informatique, Université Toulouse 1, de avril 2002 à avril 2006 ;
- Membre de la Commission de Spécialistes d'Informatique à l'Université Toulouse 1, partir de 2003 ;
- Responsable du certificat CIAD (Certificat d'Informatique Appliqué au Droit), de septembre 2003 à septembre 2006 ;
- Coordinateur local (Brésil) du projet ALFA-AIR (5 pays européens et 6 pays latino-américains) de 1996 à 1998 (voir section 4.2) ;
- Coordinateur-adjoint de la formation d'Ingénieur en Automatique et Informatique de l'Université Fédérale de Santa Catarina, de novembre 1996 à octobre 1997 ;
- Membre de la commission pédagogique de la formation d'Ingénieur en Automatique et Informatique de l'UFSC, de octobre 1995 à octobre 1997 ;
- Participation à la commission pédagogique chargée de la définition des cours des formations d'ingénieurs électricien et d'ingénieur en automatique et informatique de l'UFSC ;
- Responsable du Laboratoire d'Electronique Numérique de l'Université Fédérale de Santa Catarina, de janvier 1990 à juillet 1994, tant au niveau pédagogique qu'administratif ;
- Membre de la commission permanente pour la Formation Doctorale en Informatique (*Mestrado* du Département d'Informatique de l'UFSC), pendant l'année 93-94. Les buts de cette commission, sont, entre autres,
  - d'analyser et d'homologuer le contenu des cours,
  - de sélectionner les dossiers des candidats au *mestrado* et au doctorat,
  - de décider de la composition des jurys de thèse.
- Membre de la commission de recrutement de maître de conférences de la formation d'Ingénieur Électricien de l'Université Fédérale de Santa Catarina.

5.5 Participation à des jurys

Outre les six thèses de Mestrado que j'ai dirigées (voir section 3.4) et les deux thèses de doctorat que j'ai co-encadré, j'ai participé aux jurys suivants :

- E. MINCA, *Contribution à la supervision des systèmes de production à l'aide des réseaux de Petri flous : Application à la e-maintenance*, thèse de doctorat en cotutelle entre Université de Franche-Comté et l'Université Valahia de Targoviste, Roumanie, Septembre 2004.
- J. E. PELLICER, *Contrôle de SED utilisant les réseaux de Petri (en espagnol), Commande de SED par réseaux de Petri*, Faculdad de Ingeniería, Universidade de San Juan, Argentine, Mestrado, 120 p., 1998.
- C. W. SEIBEL, *Une méthodologie basé sur les automates hybrides pour la planification de mission d'aéronefs sans pilote* (en portugais), *Examen de qualification pour doctorat*, EEL-UFSC, 1997.
- G. A. COSTA, *Un système spécialiste basé sur logique floue avec censures pour la maintenance de locomotives* (en portugais), 188 p., Engenharia Elétrica, Universidade Federal do Espírito Santo, Vitória-ES, Brésil, outubro 1996.
- A. M. DALTRINI, *Modélisation et traitement de la connaissance basés sur un Réseau de Petri flou étendu* (en portugais). Thèse de *mestrado*, Faculté de Génie Electrique de l'Université Estadual de Campinas, Campinas-SP, Brésil, juin 1993,
- R.A. KROHLING, *Algorithmes de contrôle non conventionnels : un cas d'étude* (en portugais). Thèse de *mestrado* en Génie Electrique, Université Fédérale de Espírito Santo, Vitória-ES, Brésil, février 1994.

– M. MOECKE, *Une méthode de réduction de l'arbre d'accessibilité d'horloges temps réel pour applications industrielles*. Thèse de *mestrado*, Département de Génie Électrique, UFSC, Florianópolis, Brésil, mars 1991,
– W. de ABREU, *Méthodologie et outils de développement pour la programmation d'applications distribuées*. Thèse de *mestrado*, Département de Génie Électrique, UFSC, Florianópolis, Brésil, mars 1991,
– R. ZILLER, *L'approche Ramadge-Wonham dans la commande de systèmes à événements discrets*. Thèse de *mestrado*, Département de Génie Electrique, UFSC, Florianópolis, Brésil, août 1993,
– R. C. FERNANDES, *Système d'aide à l'opération de haute tension*, EEL-UFSC, Brésil, outubro 1995.
– A. S. CUNHA Jr., *Vérification de systèmes hybrides*, abril de 1996, EEL-UFSC, Brésil, Mestrado.
– I. TONIN, *Méthode d'inférence logique basé sur la transformation duale*, EEL-UFSC, Mestrado, junho de 1997, 67p.

Chapitre 6

Équivalence de diplômes

- *Examen de Qualification* : c'est l'équivalent au Brésil du *Qualifying Examination for PhD* américain. Le doctorant doit présenter, à la fin de sa première année de thèse une proposition de thèse dont la pertinence est jugé par un jury.
- *Mestrado* : c'est l'équivalent au Brésil du diplôme américain *Master of Science.* Il comporte une année de cours avec un minimum de 405 heures, et de un à deux années de travail de recherche (souvent d'un niveau comparable à l'ancienne thèse de 3ème cycle ou de docteur-ingénieur).
- *Professeur Assistent :* recruté sur concours. Le candidat doit posséder au moins un diplôme de thèse de *mestrado*.

Bibliographie

Publication et édition de livres :

[1] J. Cardoso, R. Valette : Réseaux de Petri (en portugais), 212 pages, 1997, Editora da UFSC (Université Fédérale de Santa Catarina), ISBN 85-328-0095-5 ;

[2] *Fuzziness in Petri nets* : J. Cardoso and H. Camargo (Ed.), Series Studies in Fuzziness, Physica-Verlag, 318 p., ISBN 3-7908-1158-0, 1998.

Articles de revues et contribution à des ouvrages :

[3] L. Frigo, J. Cardoso, G. Bittencourt. A Method for Modeling Adaptive Interactions in Intelligent Tutoring Systems. IJCEELL Special Issue on "Integrating Intelligent and Adaptive Hypermedia Techniques in Web-Based Education Systems", 10 p., Inderscience Publishers, I. Hatzilygeroudis (Editor), à paraître.

[4] X. Mao, J. Cardoso, R. Valette : A New Graph of Classes for the Preservation of Quantitative Temporal Constraints Automated Technology for Verification and Analysis : Third International Symposium, ATVA 2005, Taipei, Taiwan, October 4-7, 2005, Lecture Notes in Computer Science, Volume 3707 / 2005, pp.278-292 Springer-Verlag.

[5] C. Sibertin-Blanc, J. Cardoso, C. Hanachi. Les protocoles comme composants à part entière des systèmes coopératifs. RERIR n. 11 (Revue Electronique sur les Réseaux et l'Informatique Répartie, ISSN 1262-3261), mars 2001 (article présenté à NOTERE'2000 et sélectionné pour publication dans la revue RERIR).

[6] J. Cardoso, C. Sibertin Blanc. An operational semantics for UML interaction : sequencing of actions and local control, APII-JESA 36/2002. Reactive Systems, pg 1015-1028, ISBN 2-7462-0573-4.

[7] S. Sandri, J. Cardoso : On Possibilistic timed safe Petri nets. *Threads in Fuzzy Petri nets research, International Journal of Intelligent Systems*, vol. 14, n^o 7, 1999.

[8] J. Cardoso, R. Valette, D. Dubois : Possibilistic Petri nets. *IEEE Trans. on System, Man and Cybernetics, vol. 29, part B : Cybernetics, n^o 5, Oct. 1999.*

[9] J. Cardoso : Time Fuzzy Petri nets. *Fuzziness in Petri nets*, J. Cardoso and H. Camargo (Ed), Studies in Fuzziness, Physica Verlag, pp 115-145, 1998.

[10] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Handling Uncertainty and Resources : Petri nets, Fuzzy and Linear Logic. *Journal of the Interest Group in Pure and Applied Logics (IGPL)* 1996, p. 491-493, ISBN 0945-9103.

[11] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Linear logic for imprecise firings in Fuzzy Petri nets, In *Fuzzy logic and soft computing*, B. Bouchon-Meunier, L. Zadeh and R. Yager (Eds), World Scientific, p. 119-128, 1995.

[12] R. Valette, D. Andreu, J. Cardoso, J.C. Pascal : Fuzzy Petri nets and their application in CIME, *Special Issue of Recent Applications of Petri Nets of the Transactions of the Institute of Electrical Engineers of Japan*, Vol. 114-C, N^o 9, p. 876-880, 1994.

[13] J. Cardoso, R. Valette, D. Dubois : Petri nets with incertain markings. In *Lecture Notes in Computer Science*, G. Rozenberg (Ed), Springer Verlag, 483 :64-78, 1990.

[14] R. Valette, J. Cardoso, H. Atabakhche, M. Courvoisier, T. Lemaire : Petri nets and production rules for decision levels in FMS control. In *Artificial Intelligence in Scientific Computation : towards second generation systems*, R. Huber et al. (Eds), JC Baltzer AG, Scientific Publishing Co (IMACS), p. 301-305, 1989.

Communication à des congrès internationaux :

[15] J. Cardoso, X. Mao, R. Valette. State Class Graph for Fuzzy Time Petri Nets, ESM'2006, European Simulation and Modelling Conference, October 23-25, 2006, Toulouse, France.

[16] E. Pozzebon, J. Cardoso, G. Bittencourt, C. Hanachi. A Multi-Agent Architecture for Group Learning, IADIS International Conference e-Society, V1, ISBN : 972-8924-16-X, pg 60-67, July 2006, Dublin, Ireland.

[17] J. Cardoso, S. Cousy, G.Juanole : Extending Time Petri Nets to Fuzzy Time Petri Nets : Definition of the Graph of Fuzzy State Class, 16th IFAC World Congress, Juillet 2005, Prague, République Tchèque.

[18] C. Sibertin-Blanc, O. Tahir, J. Cardoso. Interpretation of UML Sequence Diagrams as Causality Flows. In Advanced Distributed Systems, 5th International School and Symposium (ISSASD'05), LNCS 3563, pp. 126-140, Guadalajara, Mexique, janvier 2005.

[19] O. Tahir, C. Sibertin-Blanc and J. Cardoso. A Causality-Based Semantics for UML Sequence Diagrams. In Peter Kokol editor, Proceedings of the 23rd IASTED International Conference on Software Engineering, Acta Press, pp. 106-111, Innsbruck, Austria, 2005.

[20] J. Cardoso, G. Bittencourt, L. B. Frigo, E. Pozzebon, A. Postal. MathTutor : A Multi-Agent Intelligent Tutoring System, 1st IFIP Int. Conf. on Artificial Intelligence Applications and Innovations - AIAI 2004, Toulouse, 23- 25 August 2004, Kluwer Academic Publishers, 2004. v. 1. p. 231-242. ISSN/ISBN : 1-4020-8150-2.

[21] J. Cardoso, C. Sibertin-Blanc : Ordering actions in sequence diagrams of UML, ITI 2001 Proceedings of the 23rd International Conference on Information Technology Interfaces, Pula, Croatia, June 19-22, 2001, pg 3-14.

[22] J. Cardoso, G. Bittencourt, L.L. Caimi : A rule-based tool for Fuzzy Petri net simulation. IPMU'98, Paris 6-10 juillet 1998.

[23] G. Bittencourt, J. Cardoso, L.L. Caimi : A Frame-Based Representation for Fuzzy Petri Net. IEEE WCCI'98 et FUZZ-IEEE'98, Anchorage, Alaska, 4-9 mai, 1998.

[24] S. Sandri, J. Cardoso : Possibilistic timed Petri nets. IEEE WCCI'98, Anchorage, Alaska, 4-9 mai, 1998. Fuzzy Systems Proc., 1998, V1, pg 89-94. ISBN : 0-7803-4863-X.

[25] L.L. Caimi, J. Cardoso, G. Bittencourt : A Rule-based tool for Petri net management. CESA'98, Tunisie, 1-4 avril 1998, p. 589-594.

[26] S.A. Sandri, J. Cardoso : Management of incomplete information in the processing of safe Petri nets with fuzzy durations. 7^{th} *International Fuzzy Systems Association World Congress - IFSA 97*, Prague, Rep. Tchèque, 25-29 June 1997, p. 300-305.

[27] J. Cardoso, R. Valette, D. Dubois : Fuzzy Petri Nets : an overview. 13^{th} *IFAC World Congress*, San Francisco, USA, June 30-July 5, 1996.

[28] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Handling Uncertainty and Resources : Petri nets, Fuzzy and Linear Logic. 3^{rd} rd Workshop on Logic, Language, Information and Computation - WoLLIC'96, Salvador, May 8-10, 1996 (publié dans [10]).

[29] R. Valette, F. Girault, L. A. Künzle, B. Pradin-Chézalviel, J. Cardoso : Vérification de contraintes temporelles pour des systèmes de contrôle-commande à l'aide des réseaux de Petri. In *Colloque Applications des méthodes formelles au développement de systèmes VLSI et systèmes de contrôle-command temps réel*, pages 255–267, Montréal, Canada, 2–4 Octobre 1996.

[30] C. Cavalcante, J. Cardoso, J.J.G. Ramos, O.R. Neves : Design and Tuning of a Helicopter Fuzzy Controller,. *Fourth IEEE International Conference on Fuzzy Systems and the Second International Fuzzy Engineering Symposium FUZZ-IFES 95*, Yokohama, Japan, March 20-24, 1995, p. 1549-1554.

[31] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Linear logic for imprecise firings in Fuzzy Petri nets. 5^{th} *IPMU - Information Processing and Management of Uncertainty in Knowledge-based Systems*, p. 1269-1274, Paris-France, 4-8 July 1994 (publié dans [11]).

[32] J. Cardoso, L.A. Künzle, R. Valette : Petri net based reasoning for the diagnosis of dynamic discrete event systems. *VI International Fuzzy Systems Association World Congress*, São Paulo, Brazil, July 22-28, 1995.

[33] J. Cardoso, G. Bittencourt, A. Castilho : Rule-Based Simulation of Petri nets with Imprecise Markings, *Brasil-Japan Joint Symposium on Fuzzy Systems*, p. 233-238, Campinas-Brésil, 20-22 Juillet 1994.

[34] C. Cavalcante, J. Cardoso, J.J.G. Ramos, O.R. Neves : Application of Fuzzy control to helicopter navigation. *Brasil-Japan Joint Symposium on Fuzzy Systems*, p. 72-76, Campinas-Brésil, 20-22 July 1994.

[35] F.A. Soares, J. Cardoso, J.E. Cury : An integrated environment of tools for the design of manufacturing systems. *International Workshop on Intelligent Manufacturing Systems - IMS'94*, p. 489-494, Vienna-Austria, 13-15 June 1994.

[36] S.A. Sandri, J. Cardoso : On necessity-valued Petri nets,. 5^{th} *International Fuzzy Systems Association World Congress - IFSA 93*, p. 1338-1341, Seoul-Korea, 4-9 July 1993.

[37] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Fuzzy Petri nets and linear logic. *1993 IEEE Int. Conference on Systems, Man and Cybernetics*, 2 :258-263, Le Touquet, 17-20 October 1993.

[38] J.E.R. Cury, J.M. Farines, J. Cardoso : An FMS Coordination System : an approach based on high-level Petri net. *International Symposium on Robotics, Mechatronics and Manufacturing Systems - RM2S'92*, Kobe, Japan, 16-20 September 1992.

[39] J.M. Farines, J. Cardoso, J.E.R. Cury : Specification and implementation of an FMS Coordination System Based on high-level Petri net. *International Workshop on Intelligent Manufacturing Systems - IMS'92*, p. 67-71, Dearborn-USA, 30th September-2nd October 1992.

[40] R. Valette, J. Cardoso, D. Dubois : Monitoring Manufacturing Systems by means of Petri Nets with Imprecise Markings. *IEEE International Symposium on Intelligent Control*, p. 233-238, Albany N.Y., U.S.A., 25-26 September 1989.

[41] J. Cardoso, R. Valette, D. Dubois : Petri nets with uncertain markings. *10th International Conference on Applications and Theory of Petri nets*, p. 35-51, Bonn, Germany, juin 1989 (publié dans [13]).

Communication à des workshops internationaux :

[42] L. Frigo, J. Cardoso, G. Bittencourt. Adaptive Interaction in Intelligent Tutoring Systems. CIAH-2005, International Workshop on Combining Intelligent and Adaptive Hypermedia Methods/Techniques in Web-based Education Systems, Salzburg-Autriche, 4 septembre 9 septembre 2005. Ioannis Hatzilygeroudis (Editor), p. 33-38.

[43] J. Cardoso, X. Mao, R. Valette : A graph of classes preserving quantitative temporal constraints considering unbounded transitions, 7th International Workshop on Performability Modeling of Computer and Communication Systems (PMCCS), Turin (Italie), 23-24 Septembre 2005.

[44] J. Cardoso, B. Pradin-Chézalviel : Logic and Fuzzy Petri nets. *Invited speaker* in 2^{nd} *Int. Workshop on Manufacturing and Petri nets*, Toulouse, France, June 23 1997, pp 17-34.

Communication à des congrès francophones :

[45] J. Cardoso, R. Valette, X. Mao : Un nouveau graphe de classes pour la préservation des contraintes temporelles quantitatives, MSR 2005, Modélisation des sytèmes réactifs, Grenoble (Autrans), 5-7 octobre 2005, JESA Vol.39 n.1-2-3/2005, Hermès, pp. 191-206.

[46] O. Tahir, J. Cardoso, C. Sibertin-Blanc. Génération automatique de Diagrammes d'États-Transitions à partir de Diagrammes de Séquence UML : Une approche basée sur la sémantique des Réseaux de Petri. MSR'03 Colloque Francophone sur la Modélisation des Systèmes Réactifs, Metz, France, 6-8 octobre 2003.

[47] J. Cardoso, C. Sibertin-Blanc, C. Soulé-Dupuy. Une sémantique formelle des diagrammes d'interaction d'UML via les réseaux de Petri, In Modélisation de Systèmes Réactifs (Actes MSR 2001, Toulouse, France), pg 497-512, ISBN 2-7462-0329-4, Hermès.

[48] C. Sibertin-Blanc, J. Cardoso, C. Hanachi. La spécification de protocoles d'interaction par réseaux de Petri. Dans : Journées francophones pour l'intelligence artificielle distribuée et les systèmes multi-agents JFIADSMA'01, A. El Fallah Seghrouchni & L. Magnin (Eds), Montréal, Canada, 12-14.11.2001, Hermes, p. 121-147, novembre 2001.

[49] C. Sibertin-Blanc, J. Cardoso, C. Hanachi. Les protocoles comme composants à part entière des systèmes coopératifs. Actes de NOTERE '2000, 3ème Colloque International sur les NOuvelles TEchnologies de la REpartition, I. Demeure & E. Najm (Eds.), Paris, 21 - 24 Novembre 2000. Repris dans RERIR n**?** 11 (Revue Electronique sur les Réseaux et l'Informatique Répartie, ISSN 1262-3261), mars 2001.

Communication à des congrès français :

[50] J. Cardoso, R. Valette. Un graphe de classes pour les réseaux de Petri p-temporels. Journées FAC'06 - Formalisation des Activités Concurrentes. 14 pages.

[51] J. Cardoso, R. Valette. Un nouveau graphe de classes pour la préservation des contraintes temporelles quantitatives, Journées FAC'05, Toulouse, France, 12 et 13 Mars 2005.

[52] O. Tahir, J. Cardoso, C. Sibertin-Blanc. Une nouvelle approche d'intégration de diagrammes de séquence UML basée sur les RdP de haut niveau. Journées FAC'03, Toulouse, France, 12 et 13 Mars 2003.

[53] L. A. Künzle, B. Pradin-Chézalviel, R. Valette, J. Cardoso ; "Raisonnement Temporel pour des Systemes Paralleles en vue du Diagnostic". Colloque AGI'96, Tour, France, p. 259-262, 2-3 juin 1996.

Communication à des congrès brésiliens :

[54] Eliane Pozzebon, G. Bittencourt, J. Cardoso. Une architecture multi-agent pour l'aprentissage en groupe dans les systèmes tuteurs intelligents, XVI SBIE - Simpósio Brasileiro de Informática na Educação, 2005.

[55] J. Cardoso, G. Bittencourt, L. B. Frigo et al. Petri Nets for Authoring Mechanism. In : XV SBIE - Simpósio Brasileiro de Informática na Educação, 2004, Manaus - AM. Anais do XV SBIE 2004, v. 1. p. 378-387. ISSN/ISBN : 8574011614.

[56] L. A. Kunzle, B. Pradin-Chézalviel, R. Valette, J. Cardoso ; Raisonnement temporel et diagnostique dans les systèmes parallèles (en portugais). 11^o CBA, setembro, 1996, São Paulo, Brasil, p. 435-440.

[57] J.M. Farines, J.E.R. Cury, J. Cardoso : Un environnement d'outils intégrés pour la conception de systèmes à événements discrets (en portugais). *Workshop Nacional de Projetos Cooperativos de Informática*, Itaipava-RJ, Brésil, 15-17 décembre 1993.

[58] M.C.L.F. Braga, C.A. Martin, J. Cardoso : Une expérience d'utilisation des réseaux de Petri pour le développement de logiciel pour la gestion d'une cellule flexible (en portugais). 12^o *Congresso Brasileiro de Engenharia Mecânica,* Brasília, Brésil, p. 425-428, décembre 1993.

[59] J. Cardoso, J.M. Farines, J.E.R. Cury : Un système de coordination pour des processus de fabrication basé sur le modèle réseaux de Petri (en portugais). 9^o *Congresso Brasileiro de Automática CBA 92-IFAC*, p. 1066-1071, Vitória-ES, Brésil, 14-18 septembre 1992.

[60] J. Cardoso, R. Valette, D. Dubois : Réseaux de Petri avec marquages imprécis (en portugais). 8^o *Congresso Brasileiro de Automática, CBA 90-IFAC*, p. 578-584, Belém, Brésil, 10-14 Septembre 1990.

[61] B. Bako, R. Valette, J. Cardoso : Mise en œuvre d'un système de règles de production pour la supervision des ateliers flexibles. 4^o *Congresso Nacional de Automação Industrial - CONAI'90*, p. 164-173, São Paulo, Brésil, 23-27 Juillet 1990.

Poster à des congrès internationaux :

[62] C. Sibertin-Blanc, C. Hanachi , J. Cardoso. Protocols as First-class Components of Multiagent Systems. In Fourth Intern. Conf. on Multi-Agent Systems, ICMAS 2000, S. Kraus, H. Nakashima & M. Tambe (Eds.), 10-12 July 2000, Boston (MA), USA, IEEE Computer Society Press, p. 21-22.

Deuxième partie

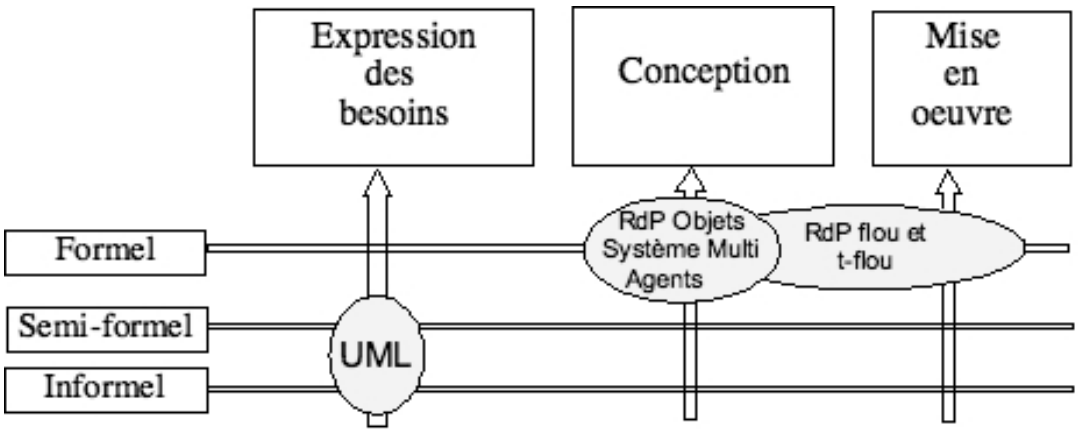
Travaux de recherche

Chapitre 7

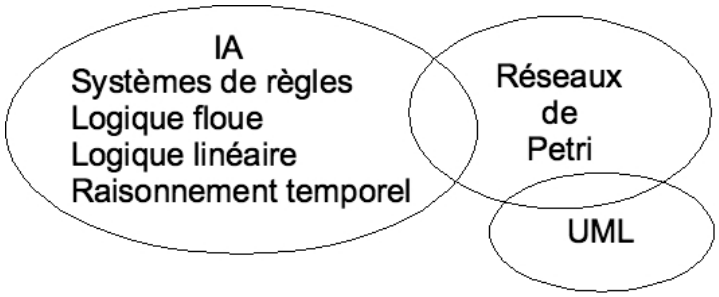
Introduction

Cette partie détaillera les quatre axes principaux de mes travaux de recherche. Ces travaux ont en commun un même outil de base, les réseaux de Petri.

Ils sont situés à différentes étapes du cycle de vie d'un système comme le montre la figures 7.1.a. Certains de mes travaux ont ainsi concerné l'expression des besoins et plus précisément le passage du semi-formel (diagrammes UML) au formel pour des besoins de vérification. D'autres ont concerné les phases de conception et de mise en œuvre en s'appuyant sur des systèmes multi-agents, et des réseaux de Petri avec ou sans information imprécise (flou).



a)



b)

FIG. 7.1 – Mes travaux de recherche a) dans le cycle de vie d'un système, b) dans les différents disciplines

Si l'on prend comme repère l'ensemble des champs disciplinaires, mes travaux se trouvent à l'intersection de différentes disciplines comme représenté figure 7.1.b. En effet, les travaux concernant l'expression des besoins peuvent intéresser aussi bien la communauté des chercheurs travaillant sur UML que celle travaillant sur les réseaux de Petri. Par contre les travaux concernant les phases de conception et de mise en œuvre se trouvent, eux, à la frontière entre les réseaux de Petri et l'intelligence artificielle.

Enfin, si on se place du point de vue des domaines d'application, ils ont été eux aussi très divers puisqu'ils vont des systèmes tuteurs intelligents aux systèmes embarqués en s'attachant à résoudre des problèmes de commande de vérification ou de supervision.

Derrière cette apparente diversité se cache une visée unique : *dans quels domaines l'application de la théorie des réseaux de Petri est-elle la plus prometteuse* et quels sont les *fondements logiques* de cette théorie ?

C'est pourquoi tous les points mentionnés ci-dessus sont toujours abordés avec comme vision principale celle que l'on peut avoir en utilisant des modélisations du type *Systèmes Dynamiques à Événements Discrets*.

Dans le cadre de cette présentation de mes travaux, j'ai choisi les regroupements suivants.

Sémantique des diagrammes UML décrivant la dynamique

Lors de l'étape d'expression des besoins, les notions de cas d'utilisation et de diagramme de séquence (DS), présentes dans le langage UML, sont très utiles. Mais elles posent problème car elles ne permettent pas une analyse formelle. Ceci est dû au manque de sémantique formelle des diagrammes de séquence. Les travaux correspondant à cet axe de recherche ont été menés en collaboration avec C. Sibertin-Blanc et correspondent principalement au co-encadrement de la thèse d'Omar Tahir [Tah06].

Dans ce travail, les diagrammes de séquence UML ont été dotés d'une sémantique opérationnelle basée sur les ordres partiels et les réseaux de Petri. Ceci a permis de définir une procédure d'obtention automatique des diagrammes d'états transitions (DET) pour les objets participant aux diagrammes de séquence formalisés. L'obtention des DET est liée à la réalisabilité d'une spécification donnée par un ensemble de DS. Un aspect important de ce travail est qu'il permet la détection de comportements ne faisant pas partie de la spécification de départ. Ces comportements, appelés scénarios impliqués peuvent alors être étudiés par le concepteur pour savoir s'il sont acceptables ou non.

Ces travaux, en plus de la thèse d'Omar Tahir, ont donné lieu aux publications suivantes : [Car01a, Car01c, Car02, Tah03a, Tah03b, Sib05, Tah05b].

Vérification et mise en œuvre de systèmes à événements discrets

Cet axe regroupe des travaux anciens avec des travaux plus récents. La vérification de propriétés temporelles, essentielle pour les systèmes embarqués, fait l'objet d'une section à part. Cet axe concerne donc l'utilisation des réseaux de Petri pour la coordination de systèmes. Il s'agit soit de réseaux de Petri ordinaires, soit de réseaux de Petri à Objets (RPO) et le domaine d'application, après avoir été celui de la commande d'ateliers de fabrication a été plus récemment celui

de la coordination de systèmes multi-agents – en particulier dans le cadre de systèmes tuteurs intelligents.

Les travaux sur réseaux de Petri ordinaires pour la coordination correspond aux encadrements des *Mestrados*[1], dans le cadre de collaborations industrielles, de J. Braga [Bra93a] et F. Soares [Soa94b].

Les réseaux de Petri à Objets ont été appliqués dans le cas particulier de la conception de Systèmes Coopératifs et de Systèmes Tuteurs Intelligents (STI). Ces systèmes sont basés sur une architecture de Systèmes Multi-Agents (SMA) distribués (les agents tuteurs). Le travail a été effectué dans le cadre d'un projet Capes/Cofecub en collaboration avec l'UFSC, Brésil, et a donné lieu aux co-encadrements de *Mestrado* de A. Postal [Pos04] et de deux thèses de doctorat en co-tutelle, celle de L. Frigo [Fri07] et celle de E. Pozzebon [Poz07]. Le couplage entre les trois modèles représentant un STI, celui du Domaine, celui de l'Apprenti et le modèle Pédagogique, est basé sur les ontologies et les réseaux de Petri à Objets [Fri05b, Car04a, Car04b, Fri05a, Poz06]. Une partie de ces travaux s'est inspirée de la notion de protocole de communication considéré comme un composant à part entière dans les systèmes coopératifs [Sib00, Sib01c, Sib01a].

Spécification et vérification de contraintes floues

Lorsque l'on s'intéresse aux étapes d'expression des besoins et de conception à un niveau très amont (contexte de l'axe de recherche concernant la sémantique des diagrammes UML), on se rend vite compte que de nombreux paramètres et de nombreuses contraintes sont mal connues. Le même type de situation se rencontre lorsque l'on passe de la commande d'ateliers de fabrication (axe précédent) à celui de la supervision de ces ateliers. C'est pourquoi il m'a semblé judicieux de considérer non seulement le caractère dynamique de ces systèmes, mais aussi le caractère imparfait de la connaissance sur certaines données. J'ai alors choisi de représenter cette imperfection sous la forme d'ensembles flous.

Cela m'a amené à effectuer quelques travaux sur les applications des ensembles flous dans le cadre de la réalisation d'un contrôleur flou pour un hélicoptère radio-commandé (*Mestrado* de C. Cavalcante [Cav94a] dans le cadre d'une collaboration industrielle) [Cav94b, Cav95]. Puis j'ai concentré mes efforts sur l'utilisation de la logique floue ou possibiliste dans le cadre des systèmes à événements discrets. Je me suis aussi penchée sur la caractérisation logique d'une séquence de transitions utilisant la logique linéaire pour mieux comprendre ce que peut être la vision logique d'une séquence.

Le besoin de représenter des données mal-connues nous a ainsi amené à proposer une extension des réseaux de Petri, les réseaux de Petri à marquage flou, permettant de prendre en compte des informations incertaines et imprécises dans le cadre des systèmes à événements discrets [Car98, Car99, Val94, Car90b, Car96, Cai98b, Bit98]. Dans ces travaux, le domaine d'application privilégié a été la surveillance des Ateliers Flexibles dont une partie concerne le co-encadrement du *Mestrado* de L. Caimi [Cai98]. En particulier, j'ai proposé une approche originale utilisant les informations du graphe de marquages pour établir des préférences entre les états possibles d'un système [San99, San98, San97].

La logique possibiliste permet de représenter de façon relativement complète le monde réel. Cependant, il est important aussi de pouvoir construire des raisonnements au-dessus des modèles du

[1]La thèse de *mestrado* (Master of Science) au Brésil comporte une année de cours avec un minimum de 405 heures, et d'une à deux années de travail de recherche (souvent d'un niveau comparable à l'ancienne thèse de 3ème cycle ou de docteur-ingénieur).

système physique. La logique classique ne peut pas être directement utilisée. La logique linéaire, avec la notion de consommation de ressources, semble une bonne candidate. Dans un travail conjoint avec B. Pradin-Chézalviel et R. Valette (LAAS), nous avons montré que les marquages imprécis et les séquences de tir imprécises peuvent être représentés par la logique linéaire et surtout que celle-ci permet une caractérisation plus claire des séquences, et des scénarios avec parallélisme vrai que ne le fait pas la théorie classique des réseaux de Petri [Car93, Car95a, Car96a, Car95b, Val96, Car97, Kun96b].

Spécification et vérification de contraintes temporelles quantitatives imprécises et floues

Comme je l'ai dit précédemment, les contraintes temporelles jouent un rôle fondamental dans les systèmes embarqués. D'autre part, la justification de l'utilisation des réseaux de Petri est en général liée au fait qu'il s'agit d'un modèle formel à partir duquel des propriétés peuvent être vérifiées formellement. C'est pourquoi un chapitre de ma présentation est spécifiquement dédié à la spécification et à la vérification de contraintes temporelles quantitatives imprécises et floues.

Pour la vérification de certaines propriétés des systèmes embarqués critiques, il faut souvent rechercher de façon exhaustive tous les états d'un système. Le graphe des classes est alors l'outil adéquat. D'autres propriétés amènent à considérer des scénarii de fonctionnement spécifiques et à analyser très exactement les contraintes temporelles entre les événements de ces scenarii, pour extraire le pire cas. Pour traiter de ce problème, j'ai proposé, avec R. Valette, un nouveau graphe des classes[2] pour les réseaux de Petri temporels qui permet d'associer à tout chemin de ce graphe l'ensemble des séquences de franchissements effectivement franchissables [Mao05a, Car05b, Car05c, Car06a]. Une partie de ces travaux correspond aux co-encadrements des *M2R* X. Mao [Mao05] et A. Hamdani [Ham06]. La prise en compte du temps de façon quantitative dans le réseau de Petri peut être utilisée pour faire une abstraction du comportement continu d'un système hybride ; une application à la modélisation d'un drône terrestre a été faite dans le cadre du projet de fin d'études du Mastère de Z. Saharoui [Sah07].

Un point important est l'expressivité alliée à la simplicité des modèles. Dans un système embarqué, toutes les évolutions du systèmes ne sont pas connues avec certitude. Le modèle de réseaux de Petri temporel flou permet de traiter des informations dont l'incertitude apparaît sous la forme d'une ignorance partielle (plutôt que de nature aléatoire). Ce type d'information est traité de façon efficace par la théorie des possibilités qui permet d'allier l'expressivité et la simplicité. Cette théorie permet de traiter l'ensemble des cas de façon ordonnée au lieu de considérer les cas individuellement : les fenêtres temporelles floues représentent en fait des ensembles emboîtés d'intervalles temporels. Cela est utile dans une phase d'ajustement des paramètres. Par exemple, dans le cas d'une procédure d'exécution à distance, on souhaite trouver la valeur d'un paramètre de façon à pouvoir avoir un compromis entre attendre trop longtemps ou perdre des réponses tardives, sans avoir à tester plusieurs cas possibles. Une première extension d'un graphe de classes au cas flou a déjà été faite dans le cas du parallélisme par entrelacement [Car05a] lors de l'encadrement du *M2R* de S. Cousy [Cou04]. Nous avons aussi proposé une extension au cas flou pour l'outil GraphC [Car06b].

Les chapitres 8 à 11 détailleront les quatre axes présentés dans cette introduction et le chapitre 12 énoncera les perspectives.

[2] L'outil GraphC – `http ://graphc.sourceforge.net`

Chapitre 8

Sémantique des diagrammes dynamiques en UML

8.1 Introduction

Au niveau de l'expression des besoins, entre les méthodes formelles – utilisées surtout dans le milieu académique – et les méthodes informelles, un langage de modélisation semi-formel tel que UML - Unified Modelling Langage, proposant différents diagrammes pour représenter différents aspects intervenant dans la conception des systèmes, a connu un vif succès dans le milieu industriel. Le langage UML, avec sa représentation graphique, possède une sémantique semi formelle, définie de façon ambiguë et quelques fois contradictoire. Certains défenseurs d'UML prétendent que cela permet de laisser une flexibilité au concepteur dans une phase préliminaire, au prix cependant de rendre difficile ou impossible la vérification formelle. S'ajoute au problème d'une sémantique ambiguë, ou comme conséquence de ce problème, un manque de lien entre des diagrammes dynamiques étroitement liés tels que les diagrammes de séquence caractérisant un cas d'utilisation et les diagrammes d'état transition des objets qui y participent.

Bien que pouvant s'utiliser dans tout le cycle de développement de systèmes informatiques, c'est dans l'analyse des besoins que les scénarios sont très largement utilisés, à partir de l'identification des cas d'utilisation. Chacun des scénarios est modélisé dans la notation UML soit par un Diagramme de Séquence (DS) soit par un diagramme de collaboration. Les diagrammes de classes décrivent la structure statique du système en termes de classes et d'associations entre ces classes ; le comportement de chaque classe est décrit par un Diagrammes d'États-Transitions (DET).

La figure 8.1 montre les différents aspects (ou étapes) abordées dans les travaux que j'ai menés dans le cadre de la sémantique des diagrammes dynamiques en UML. Ces travaux sont discutés dans les sections suivantes. Étant donnée une spécification S exprimée par un ensemble de DS, l'étape 1 consiste à donner une sémantique formelle à chaque DS, selon l'une des quatre sémantiques formelles proposées (section 8.2). Nous avons choisi d'exprimer la sémantique par des réseaux de Petri (RdP) ce qui nous permet à l'étape 2 (section 8.3.1) de construire un RdP par objet R_o dont le langage (minimal) contient celui de S (donc pour tous les DS où cet objet participe). Une fois obtenu le comportement de chacun des n objets, nous avons travaillé sur deux volets : d'une part obtenir le réseau de Petri global R_s décrivant la spécification S (étape 3, section 8.3.2) et s'intéresser à la réalisabilité de la spécification S (étape 4, section 8.4), et d'autre part obtenir les DET des classes des n objets participant aux k scénarios de S (étape 5, section 8.3.3). L'étape 3 fournit le RdP du

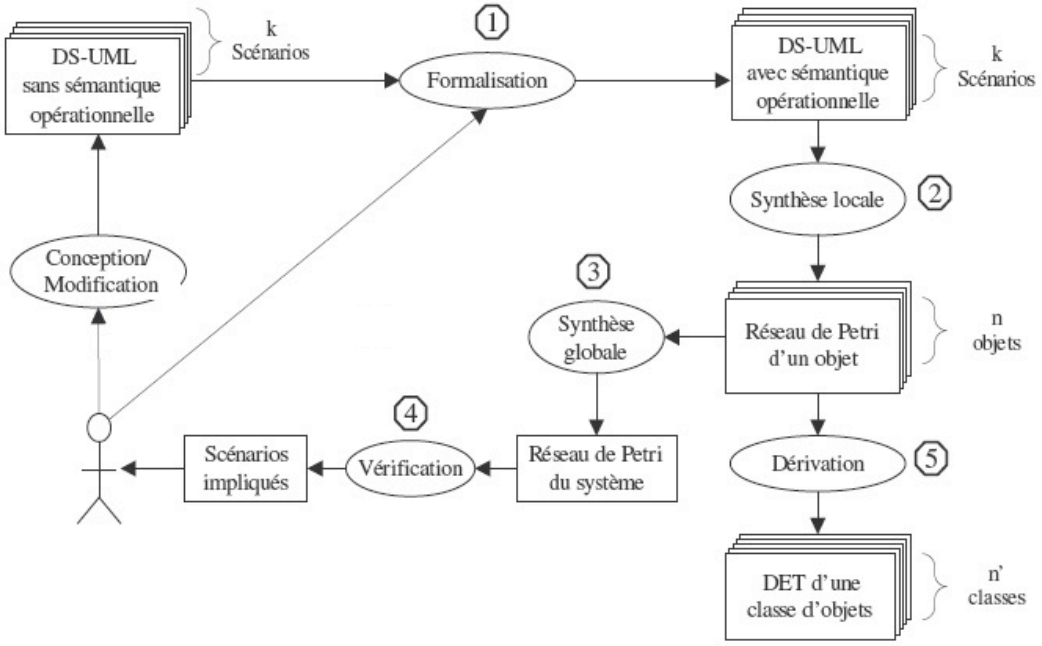


FIG. 8.1 – Vue générale d'étude des DS et des DET d'UML via les réseaux de Petri

système, en connectant les RdP de tous les objets par des places de communication correspondant aux messages échangés. Finalement, l'étape 4 réalise la détection de comportements ne faisant pas partie de la spécification de départ, appelés scénarios impliqués. L'étape 5 consiste basiquement à générer, pour chaque classe, le graphe de marquage du RdP R_o, en tenant compte de la notion de rôle.

Ces travaux ont eu comme point de départ un problème d'enseignement : le souhait de lier les trois démarches traitées (par trois professeurs différents) dans la discipline *Modélisation de la dynamique des systèmes* du DESS IGSI – Ingénierie et gestion des systèmes d'information. La possibilité de présenter ces trois démarches de façon cohérente dans le cadre d'un projet unique a permis aux étudiants d'avoir une vision d'ensemble. Cette expérience a donnée lieu à un premier article [Car01a] et par la suite à la thèse de doctorat de Omar Tahir [Tah06], co-encadrée par Christophe Sibertin-Blanc et moi-même, ainsi qu'au projet Féria *Structure des interactions et comportement des composants d'un système* (IRIT, LAAS et ONERA). Ces travaux fondés sur le standard UML 1.5, ont conduit aux publications suivantes : [Car01a, Car01c, Car02, Tah03b, Tah03a, Sib05, Tah05b].

8.2 Sémantiques pour les diagrammes de séquence UML

Les DS décrivent la dynamique d'un système sous la forme d'échange de messages entre le système et les acteurs de son environnement ou entre les composants du système. Le modèle de référence des DS d'UML est les *Messages Sequence Charts* (MSC) [zITU96]. En effet, les DS et les MSC ont la même représentation graphique, bien qu'ils se distinguent par leurs définitions. Dans les DS, une interaction est définie comme un ordonnancement global de tous les messages échangés, tandis que dans les MSC, une interaction est définie comme un ordonnancement, local dans chaque objet, des événements d'envoi et de réception de messages, les synchronisations entre les objets étant assurées

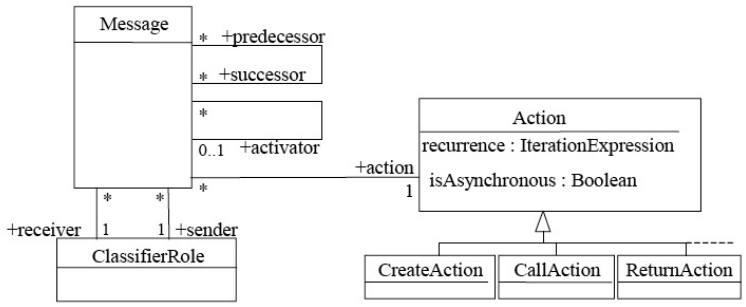


FIG. 8.2 – Meta-modèle UML concernant les interactions

par les transmissions des messages. Ainsi, les MSC ont une sémantique opérationnelle bien définie, fondée sur l'ordonnancement des événements d'émission et de réception des messages échangés ; par contre, dans les DS, l'ordonnancement des messages eux-mêmes ne définit aucune sémantique opérationnelle montrant comment ils s'exécutent.

Plusieurs approches ont été proposées pour doter les DS d'une sémantique formelle. Les sémantiques proposées sont définies par la translation des DS dans un formalisme ayant une sémantique formelle, tel que les réseaux de Petri [zGeh98, zSto99, zBer02], la logique temporelle [zKna99, zSeu02], les systèmes de transitions étiquetés [zOka03], les algèbre de processus [zLi04], ou encore les ordres partiels étiquetés [zKna99, zDem02].

Notre approche part de la syntaxe abstraite des interactions, en considérant qu'un DS est décrit sous la forme d'une instance du méta-modèle UML de façon à nous tenir au plus prêt de ses indications sur la sémantique [Car01a]. Sur l'extrait du méta-modèle de la figure 8.2 on peut voir les associations *activation* R_{act} (un message peut avoir 0 ou 1 *activator*) et *précédence* R_{pre}. Les contraintes sémantiques, données sous la forme d'un ensemble de règles bien formées UML, correspondent aux propriétés structurelles des DS qui ne peuvent pas être exprimées par leur syntaxe abstraite. Les contraintes (relatives à l'aspect dynamique des interactions) sur les messages sont les suivantes : i) un Message ne peut pas être son propre prédécesseur, ii) les prédécesseurs d'un Message doivent avoir le même activateur que celui-ci. Ces contraintes sont insuffisantes puisqu'elles interdisent de synchroniser les messages appartenant à plusieurs activations et autorisent la construction de DS incohérents.

Bien que certains auteurs aient choisi de n'utiliser qu'un seul type d'association entre les messages [zPik03], nous avons respecté les deux associations en les interprétant en termes d'un ordonnancement entre les événements de communication des messages. Pour ce faire, nous avons transcrit l'association d'activation en termes de précédence entre les messages obtenant ainsi une nouvelle association de précédence entre tous les messages qui *étend* celle du méta-modèle et qui considère tous les messages comme étant asynchrones : l'association de *précédence étendue* R_{msg}. Cette association lève les ambiguïtés d'UML sur les liens sémantiques entre les deux associations entre les messages d'UML, celles de précédence R_{pre} et d'activation R_{act}, et permet une définition des DS plus simple que nous appelons *DS-ordonnancé* de la forme $< O, M, From, To, R_{pre}, R_{act}, IsAsync >$.

Nous avons introduit le concept de type de message, concept non pris en compte dans le méta-modèle, et pourtant nécessaire pour l'intégration de plusieurs DS, en considérant que des messages appartenant à plusieurs DS peuvent être considérés identiques s'ils sont du même type. En fait, comme notre intérêt est celui d'obtenir le DET des classes d'objets participant aux différents DS d'une spécification, il faut être en mesure d'identifier les messages identiques dans

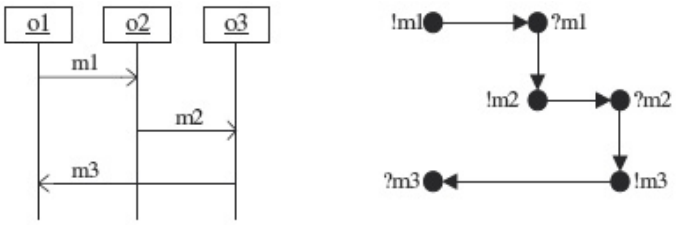


FIG. 8.3 – DS_1 et sa sémantique linéaire

les différents DS. La prise en compte du type de messages amène à une structure de la forme $< O, M, From, To, R_{msg}, S, Content >$, nommée DS-typé, où $Content : M \rightarrow S$ est une application qui associe à chaque message de M un service de S, l'ensemble des noms de méthodes invoquées et de signaux envoyés au cours de l'interaction.

Nous avons aussi introduit le concept de type d'événement de façon à vérifier si deux événements figurant dans des DS distincts peuvent être fusionnés. Cela nous a permis de définir la sémantique opérationnelle pour les DS comme une relation d'ordre partiel étiquetée entre les événements d'émission et de réception des messages, de la forme $< E, <, \Sigma, \mu_{ev} >$, où E est l'ensemble des événements de communication des messages, Σ l'ensemble de types d'événements et $\mu_{ev} : E \rightarrow \Sigma$ le type de l'événement.

Cette relation d'ordre partiel dépend de la relation de précédente étendue R_{msg} et d'une relation d'ordre $<_{sync}= (!m, ?m); m \in M$ correspondant à la synchronisation entre l'émission et la réception de chaque message. Selon l'interprétation de la relation R_{msg} entre les messages d'un DS-typé, la sémantique, traduite par la relation d'ordre partiel $<$ entre les événements, sera différente. En particulier, quelle interprétation donner à l'exécution d'un message (est-ce son émission ou sa réception ?) et à sa terminaison. Nous avons proposé les sémantiques suivantes :

Sémantique linéaire $<_{lin}$:
L'exécution d'un message correspond à l'exécution de ses deux événements d'émission et de réception, et un message est considéré terminé une fois qu'il est reçu. Cette sémantique est caractérisée par une *sérialisation totale de l'exécution* de tout DS puisque pour chaque couple de message m, m' tels que $mR_{msg}m'$, nous avons $!m <?m <!m' <?m'$ (figure 8.3 et figures 8.4.a et 8.4.b).

Sémantique émission $<_{emi}$ [Car02] :
Elle se focalise sur la production des messages. Elle considère que l'exécution d'un message correspond à l'exécution de son événement d'émission, et un message est terminé dès qu'il est émis. Cette sémantique est caractérisée par le fait que la précédence entre deux messages implique l'ordonnancement de leurs émissions respectives. Ceci entraîne par conséquent une *sérialisation totale des actions d'émission* de l'ensemble des messages d'un DS (figures 8.4.a et 8.4.c).

Sémantique MSC $<_{msc}$:
L'ordonnancement des messages est local à l'intérieur de chaque objet. Cette sémantique considère que l'exécution et la terminaison d'un message sont des concepts locaux à chaque objet. Pour un message m émis par l'objet $o1$ vers l'objet $o2$, du point de vue de $o1$, l'exécution de m correspond à son événement d'émission et m est considéré terminé dès qu'il est émis ; du point de vue de $o2$, l'exécution de m correspond à son événement de réception et m est considéré terminé une fois qu'il

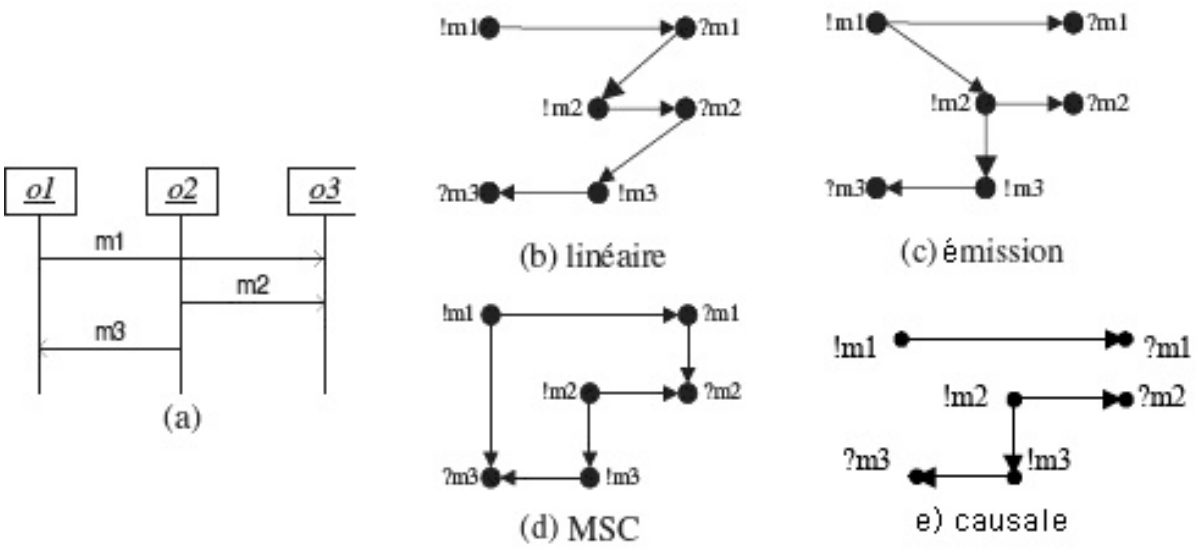


FIG. 8.4 – DS_2 et ses sémantiques

est reçu. L'ordonnancement global des messages est projeté sur (la ligne de vie de) chaque objet, et il induit un ordonnancement local des événements de chaque objet. Chaque objet exécute ses événements en séquence et les différents objets synchronisent leurs comportements respectifs par les transmissions des messages (figure 8.4.a et 8.4.d).

Sémantique causale $<_{cau}$ [Sib05, Tah05b] :
C'est un affaiblissement de la sémantique MSC. L'exécution et la terminaison d'un message sont analogues à la sémantique MSC, ce sont des concepts locaux à chaque objet. Cependant cette sémantique ne séquence que les paires d'événements de communication qui sont causalement liés : deux événements réalisés par un objet ne sont ordonnés que si l'un est la conséquence directe ou indirecte de l'exécution de l'autre ; à la différence de la sémantique MSC, l'ordonnancement des événements à l'intérieur de chaque objet n'est pas nécessairement un ordre total (figure 8.5).

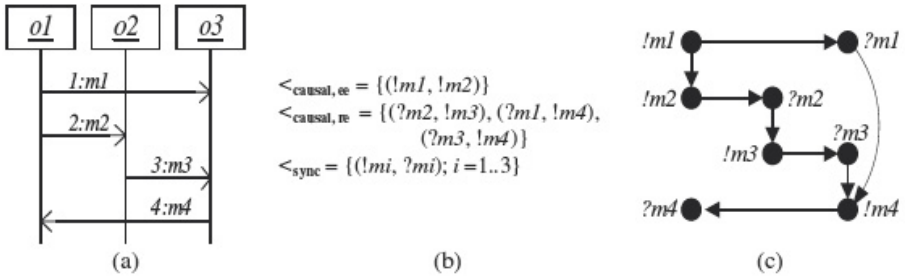


FIG. 8.5 – DS et sa sémantique causale

L'idée de proposer différentes sémantiques est que chacune peut être reliée à des classes d'applications ou de domaines de façon que le concepteur puisse choisir la sémantique la plus appropriée.

L'étude et la définition de ces quatre sémantiques nous a conduit à identifier trois propriétés structurelles des DS :

– la *séquentialité* qui implique que chaque message est émis par l'objet ayant reçu le message immédiatement précédent (DS_1 sur la figure 8.3 est séquentiel) ;
– la *localité du contrôle* qui implique que chaque message est émis par l'objet ayant envoyé ou reçu le message immédiatement précédent ;
– la *causalité locale* qui exprime que, si un objet réalise une émission puis une réception, il existe une séquence d'événements causalement liés depuis l'émission jusqu'à la réception, ce qui induit une causalité complète entre les événements exécutés par chaque objet.

En se basant sur ces propriétés, nous avons étudié la distributivité de chacune de ces sémantiques. Une sémantique est dite *distribuable* pour un DS si et seulement si toutes les contraintes d'ordonnancement qu'elle impose entre les événements exécutés par des objets différents sont assurées par des transmissions de messages. Les résultats auxquels nous sommes parvenus sont les suivants : les sémantiques causale et MSC sont distribuables pour tout DS. En revanche, la sémantique émission n'est distribuable que pour les DS localement contrôlés, et la sémantique linéaire n'est distribuable que pour les DS séquentiels.

Il est à remarquer que ces sémantiques opérationnelles des DS d'UML 1.4 sont définies sous la forme d'une relation d'ordre partiel entre les événements de communication des messages, qui s'apparente à celle des DS d'UML 2.0. Ceci a pour conséquence de pouvoir interpréter en UML 2.0 les DS conçus selon UML 1.4 et les versions antérieures.

Depuis la première proposition d'une sémantique pour UML basée sur les réseaux de Petri [Car01a], les travaux ont évolué jusqu'à la forme présentée ici et décrite en détail dans [Tah06] et partiellement dans [Sib05, Tah05b].

8.3 Synthèse par réseaux de Petri d'une spécification par DS

La représentation des interactions décrites par des scénarios en utilisant les réseaux de Petri a fait l'objet de plusieurs travaux, soit en partant des DS UML [zBer02, zSto99, zGeh98, Car01a] soit en partant des MSC [zGra93, zHey00, zOla02] .

Étant donné une spécification S décrite par un ensemble fini de DS, dont la sémantique est conforme à une des sémantiques décrites section 8.2, nous l'avons représentée par un seul réseau de Petri global R_s considérant à la fois les n objets et les k scénarios. Cependant nous avons choisi, plutôt qu'une transformation qui produirait directement un RDP global, de transformer les DS en conservant la structure d'objets à partir de deux étapes : d'abord construire, pour chacun des objets, un réseau de Petri R_o qui décrit toutes les séquences d'interactions que cet objet doit être capable de réaliser pour jouer son rôle dans les DS de la spécification (étape 2, fig. 8.1), et ensuite connecter par l'intermédiaire de places de communication, ces réseaux de Petri locaux en un réseau global R_s du comportement de l'ensemble du système (étape 3, fig. 8.1). Ces deux étapes permettent une meilleure analyse de la conception, vu que la structure du système, constitué d'un ensemble de processus qui interagissent les uns avec les autres de façon asynchrone par envois de messages, est conservée. À partir du réseau R_o décrivant le comportement d'un objet, nous générons le DET de la classe de cet objet en tenant compte de la notion de rôle (étape 5, fig. 8.1).

8.3.1 Synthèse d'un objet par un réseau de Petri R_o

Le réseau de Petri d'un objet participant à une spécification S (exprimée par k DS) doit décrire toutes les séquences d'interaction que cet objet doit être capable de réaliser pour y jouer son rôle. La synthèse d'un tel réseau est réalisée à partir de l'ensemble des traces des interactions réalisées par l'objet dans les DS dans lesquels il figure.

D'une façon générale, les DS suggèrent un modèle d'exécution dans lequel les objets sont des processus séquentiels, et le parallélisme provient de la concurrence entre ces processus. Pour respecter ce modèle, il faut donc considérer les réseaux de Petri selon une sémantique d'entrelacement, qui décrit bien comment un ensemble de processus séquentiels peuvent progresser de front tout en se synchronisant en certains points, dont la plus simple est la sémantique langage [zPom92]. Les sémantiques $<$ proposées section 8.2, définies sous la forme d'une relation d'ordre partiel entre les événements, de nature non entrelacée, doivent être transcrites sous la forme de plusieurs séquences par une sémantique entrelacée, de façon à prendre en compte les exécutions concurrentes. Ainsi, chaque exécution séquentielle d'un DS ds pour une sémantique $<$ est décrite par une linéarisation v ; l'ensemble des linéarisations de ds définit ce que nous appelons son langage $L(ds, <)$, ou $L(ds)$ s'il n'est pas nécessaire de préciser la sémantique.

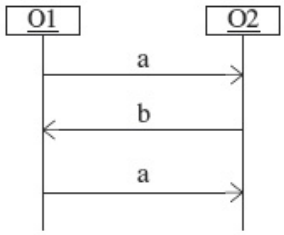
Le rôle que doit jouer un objet o pour réaliser une linéarisation décrivant une exécution séquentielle est donné par la projection du mot $v \in L(ds)$ sur cet objet, $v \mid o$. Le rôle que doit jouer chaque objet pour réaliser toutes les linéarisations d'un DS est exprimé par la notion de *langage d'un objet*, $L(ds) \mid o$.

L'ensemble de toutes les séquences d'événements exécutées par un objet o dans S est le langage de l'objet $L(o)$ (dans le contexte d'une spécification), obtenu en faisant l'union des langages $L(ds) \mid o$ de cet objet dans les DS auxquels il participe.

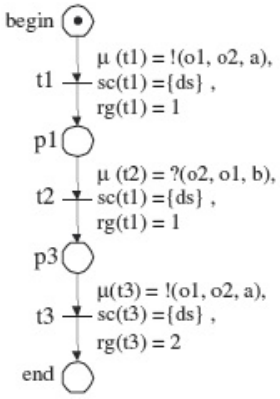
À cette étape (figure 8.1, étape 2), nous cherchons à construire un RdP décrivant un objet que l'on note R_o, dont le langage $L(R_o)$ soit égal au langage de l'objet o, $L(o) = L(R_o)$. La construction est constituée de deux phases :
- la construction d'un RdP R_w décrivant un mot w du langage d'un objet (dans un DS) $L(ds) \mid o$; cette description du comportement de l'objet est partielle ;
- la construction du RdP R_o intégrant les RdP R_w de tous les mots du langage $L(o)$ d'un objet (pour tous les DS) pour en avoir une description complète.

Construction de R_w
Le RdP R_w est du type machine à états, avec une place *begin* n'ayant aucune transition amont initialement marquée et qui correspond au début de l'exécution de la séquence d'événements que cette machine à états réalise, et une place *end* n'ayant aucune transition aval correspondant à la fin de l'exécution de la séquence d'événements.

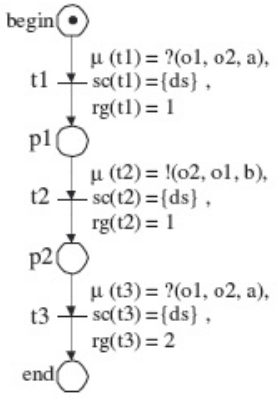
Afin que chaque transition décrive un événement d'émission ou de réception et pour pouvoir connecter chaque événement d'émission à son événement de réception correspondant, chaque transition de R_w est étiquetée par :
- le type d'événement μ à partir duquel elle est créée,
- le rang rg de l'occurrence de ce type d'événement dans le mot, ce qui permet de distinguer les occurrences d'un même symbole,
- l'ensemble sc des DS dans lesquels figure l'événement que la transition réalise.



(a) Un DS



b) R_w pour $w \in L(ds) \mid o_1$



c) $R_{w'}$ pour $w' \in L(ds) \mid o_2$.

FIG. 8.6 – Représentation du comportement d'un objet dans *un* DS par machine à états R_w.

Soit le DS de la figure 8.6.a ; les figures 8.6.b et 8.6.c montrent, respectivement, les machines à états R_w et $R_{w'}$ des mots $w = !(o_1, o_2, a).?(o_2, o_1, b).!(o_1, o_2, a) \in L(ds) \mid o_1$ et $w' = ?(o_1, o_2, a).!(o_2, o_1, b).?(o_1, o_2, a) \in L(ds) \mid o_2$.

Construction de R_o

Le réseau de Petri R_o (représentant le langage $L(o)$ d'un objet, tel que $L(R_o) = L(o)$) doit intégrer tous les réseaux R_w (représentant le langage $L(ds) \mid o$ d'un objet o pour un DS ds de S). Cette intégration – une opération de composition d'alternatives qui doit gérer le problème du *choix retardé des alternatives* [zJos94] – consiste à : 1) faire l'union disjointe des machines à états R_w, 2) fusionner les places *begin* et les places *end* de chaque R_w et 3) fusionner les fragments de RdP qui décrivent un même préfixe commun à plusieurs alternatives. Les machines à états à intégrer sont telles que toutes les transitions qui satisfont les conditions de fusion sont nécessairement de même rang. Ceci provient du fait qu'une occurrence de symbole d'un préfixe commun à plusieurs mots a le même rang dans chacun de ces mots.

Deux cas particuliers apparaissent lors de la construction du RdP R_o associé au langage d'un objet o : 1) quand un mot w_1, réalisation de o dans un DS ds_1, est un préfixe propre d'un mot w_2, réalisation de o dans un autre DS ds_2 et 2) quand un objet ne figure pas dans tous les DS de la spécification. Ces deux cas sont traités en utilisant le symbole vide ϵ.

Considérons une spécification S représentée par deux DS, le ds_1 de la figure 8.7.a et le ds_2 de la figure 8.7.b. Le réseau de Petri R_{o_1} représentant le comportement de l'objet o_1 dans S et montré figure 8.7.c, est obtenu en composant le RdP R_w représenté fig. 8.6.b (correspondant à $w = !(o_1, o_2, a).?(o_2, o_1, b).!(o_1, o_2, a) \in L(ds_2) \mid o_1$) et R_u (correspondant à $u = !(o_1, o_2, a).?(o_1, o_2, b) \in L(ds_1) \mid o_1$ et non montré sur la figure). Le mot $!a?b$ est préfixe propre du mot $!a?b!a$; la transition

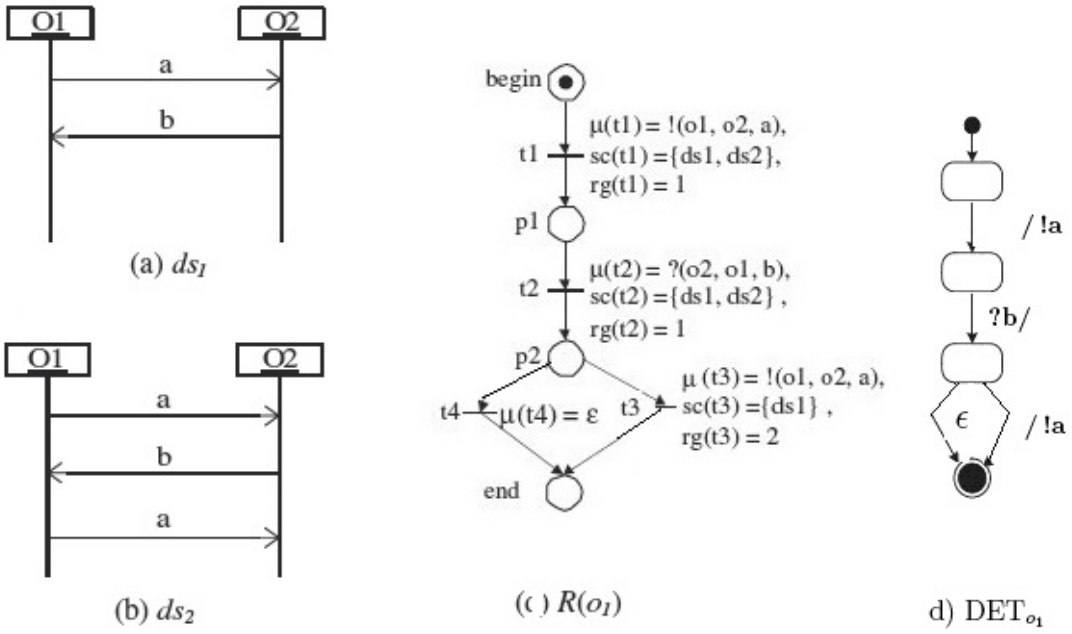


FIG. 8.7 – Représentation du comportement d'un objet dans *deux* DS.

$t_4 = \epsilon$ apparaît dû à la composition des R_w correspondant à $!a?b\epsilon$ et $!a?b!a$.

Le réseau de Petri R_o est considéré un réseau *local* dans le sens où il est local à un objet ; du point de vu de la spécification, il est aussi un *s*ous-réseau puisqu'il ne considère pas les autres objets appartenant à la spécification.

L'évolution dans la définition de la sémantique depuis [Car01a] a provoqué aussi un changement dans la construction du réseau de Petri. La justification formelle de cette démarche a été décrite dans [Tah06].

8.3.2 Synthèse d'une spécification par un réseau de Petri R_s

Une spécification S est constituée par un ensemble fini de DS, représentant des alternatives d'exécution exclusives les unes des autres, une seule pouvant se produire à la fois. Le langage de S, noté $L(S)$, est l'union des langages $L(ds)$ des DS constituant cette spécification. Nous cherchons à construire un RdP, que l'on note R_s, dont le langage soit égal à celui de S, $L(S)$, correspondant à l'étape 3 sur la figure 8.1. Cependant, compte tenu du pouvoir d'expression des RdP, il se peut qu'un tel réseau n'existe pas toujours ; dans ce cas, nous construisons un RdP dont le langage soit minimal à contenir celui de S, c'est-à-dire $L(S) \subseteq L(R_s)$.

Dans l'étape 2 (figure 8.1) décrite section 8.3.1, le réseau de Petri R_o décrivant le comportement (local) d'un objet participant à tous les DS de S a été obtenu. Il faut maintenant représenter le comportement (global) R_s du système spécifié par S considérant aussi les communications entre tous les objets. Il faut donc relier chaque événement d'émission d'un message à son événement de réception correspondant. Comme illustré sur la figure 8.8, cette connexion est réalisée par l'introduction d'une place de communication entre chaque paire de transitions (t, t') correspondant respectivement à l'émission $!m$ et à la réception $?m$ du même message. Les deux transitions doivent avoir le même type de message (et être de la forme envoi/réception) et même rang (la énième

émission d'un type de message est liée à la énième réception du même type de message).

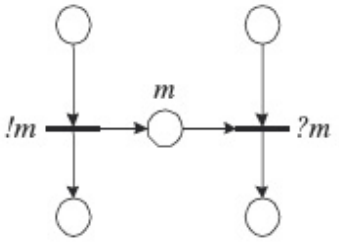


FIG. 8.8 – Motif de RdP représentant l'envoi et réception d'un message.

Le RdP global R_s est obtenu en connectant les sous-réseaux d'objets R_o deux à deux. La connexion se fait par paire de transitions en établissant un lien entre la transition d'émission et la transition de réception de chaque message comme indiqué figure 8.8. Cette modélisation proposée permet de décrire les quatre sémantiques définies section 8.2.

Cependant, il se peut que le réseau R_s admette d'autres exécutions ne figurant pas dans la spécification S, il s'agit des exécutions exprimées par le langage $L(R_s)\backslash L(S)$. Ces exécutions sont en fait impliquées par les interactions entre les objets figurant dans S, dans le cas où ces derniers ne disposent pas de suffisamment d'information sur l'activité les uns des autres. La détection de ces exécutions additionnelles est traitée dans l'analyse de la réalisabilité section 8.4.

Une première version de la synthèse d'une spécification a été faite en [Tah03a] puis [Tah03b] et a été remplacé par la démarche ici décrite et formellement présenté dans [Tah06] avec aussi un exemple d'application.

8.3.3 Dérivation de diagrammes d'états-transition UML

Le problème de la synthèse du comportement des objets par un diagrammes d'états-transition (DET) à partir de scénarios est un sujet bien étudié dans la littérature. On peut citer parmi les travaux faisant la synthèse à partir de DS [zKos04, zWhi00, zKhi01, zMak01, zZia04] et parmi ceux faisant la synthèse à partir des MSC [zKrü99, zUch03, zMan99, zHar02]. Certains de ces travaux sont sensibles à l'ordre de concaténation de traces, d'autres doivent définir un vecteur d'états pour détecter les conflits de spécification au sein de chaque scénario ou ont besoin d'une interaction avec le concepteur ; il y en a qui garantissent une égalité entre le langage engendré par le DET et le langage des séquences d'exécutions décrites par les DS de la spécification, d'autres pas.

Dans notre approche, l'étape consistant à générer le DET de chaque objet (figure 8.1, étape 5) est immédiate puisque ce DET_o correspond au graphe de marquage du réseau de Petri R_o (section 8.3.1). En fait, il y a une correspondance entre les concepts du méta-modèle des machines à états et ceux du méta-modèle des interactions, les méta-classes *Event* et *Action*, et d'une part entre les classes *CallAction* et *CallEvent*, et d'autre part entre les classes *SendAction* et *SignalEvent*, ce qui signifie que chaque action d'émission déclenche un événement de réception du coté du récepteur, et inversement chaque événement de réception est lié à une action d'émission. À chaque transition de R_o correspond une transition de DET_o de telle façon que s'il s'agit d'une *réception*, la partie *action* de la transition correspondante dans le DET_o est vide ; s'il s'agit d'une *émission*, la partie *événement* de la transition du DET_o est vide [Tah03b]. La figure 8.7.d représente le DET décrivant

le comportement de l'objet o_1 dans les DS des figures 8.7.a et 8.7.b, obtenu à partir du RdP de la figure 8.7.c.

Si un état e_0 est relié par un évènement $?r$ à un état e_1 d'où ne sont issus que des transitions reliées par des actions $!s_2, \ldots, !s_n$ vers des états $e_2, \ldots, e_n$, il est possible de réduire le DET en reliant directement e_0 aux états $e_2, \ldots, e_n$ par des transitions étiquettées $?r/!s_2, \ldots, ?r/!s_n$.

Cependant, conformément au méta-modèle UML, et en particulier au concept de State Machine, un DET est associé à une *classe* d'objets et non à un objet particulier. Le DET d'une classe doit décrire le comportement de toute instance de cette classe, et non pas seulement celui d'une instance particulière. Pour générer le DET d'une classe c à partir d'un ensemble de DS, il faut donc considérer que, dans ces DS, le nom donné à un objet n'est pas significatif et est donné à la place d'*une instance quelconque de la classe c représentant toutes les autres* ; seule la classe de cet objet, et éventuellement son rôle, doit être prise en compte. Lorsque plusieurs instances d'une même classe d'objets, par exemple i_1 et i_2, sont impliquées dans une même interaction sans que leurs rôles soient précisés, il faut considérer que, dans la construction du RdP de i_1 donnée section 8.3.1, R_{i_1} intègre aussi le comportements de i_2 puisque l'on peut tout aussi bien ajouter à l'ensemble des DS de la spécification un autre DS dans lequel les noms de i_1 et i_2 sont intervertis.

Dans le cas où, dans certains des DS de la spécification, un rôle est associé à certains objets (nommés NomObjet :/rôle :classe), pour que la spécification soit non ambiguë, il faut, dans la construction de R_o, produire en fait un RdP R_{o_r} pour chacun des rôles r de cette classe d'objets. C'est l'exemple bien connu de la modélisation des communications téléphoniques, où l'une des deux instances de la classe *abonné* tient le *rôle appelant* alors que l'autre joue le *rôle appelé*. En conséquence, étant donnée une classe d'objets, le processus de construction de son DET comprend les deux étapes suivantes :

1. pour chaque rôle de cette classe, représenté par un objet o_r, construire le sous-réseau R_{o_r},
2. intégrer les R_{o_r} de chacun de ces rôles en un seul R_o valable pour toutes les instances de la classe leur permettant d'occuper chacun des rôles, et générer le DET correspondant.

Cette démarche est décrite dans [Tah03b] et [Tah06].

8.4 Réalisabilité

Finalement, il nous reste décrire l'étape 4 représentée sur la figure 8.1 permettant d'effectuer la vérification du comportement décrit par le réseau global R_s et par cela sa réalisabilité. Cela consiste à vérifier que les exécutions engendrées par la composition des DET des objets sont conformes à celles décrites par les DS de la spécification et à détecter d'éventuels scénarios impliqués n'appartenant pas à la spécification mais impliqués par les interactions entre les objets.

La réalisabilité d'une spécification par MSC et a été formalisé par [zAlu01] en utilisant la notion de scénario impliqué ; cette notion y est illustré sur l'exemple suivant qui décrit un système distribué dans lequel les traitements et les données sont répartis sur plusieurs sites. Deux objets (ou processus) o_1 et o_2 mettent à jour à distance les données d'une base de données utilisées dans la commande d'une centrale nucléaire. Dans cette base la variable v_1 indique le niveau de carburant en uranium et la variable v_2 indique le niveau de l'acide nitrique. Les deux DS ds_1 et ds_2 montrés figure 8.9 (a et b respectivement) décrivent les mises à jour de ces deux variables. Le message *inc* dénote une requête d'incrémentation d'une unité soit du niveau de l'uranium soit de l'acide nitrique, alors que

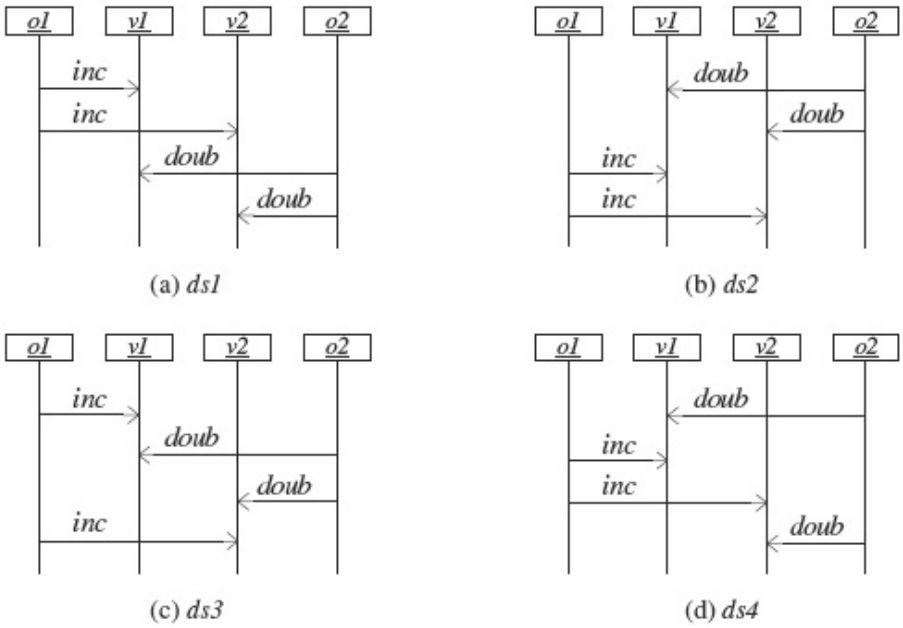


FIG. 8.9 – Les exécutions des DS ds_3 et ds_4 sont impliquées par celles des DS ds_1 et ds_2.

le message *doub* dénote une requête de doublement du niveau de chacun de ces deux liquides. Afin d'éviter tout accident, ces deux variables doivent être mises à jour de la même façon.

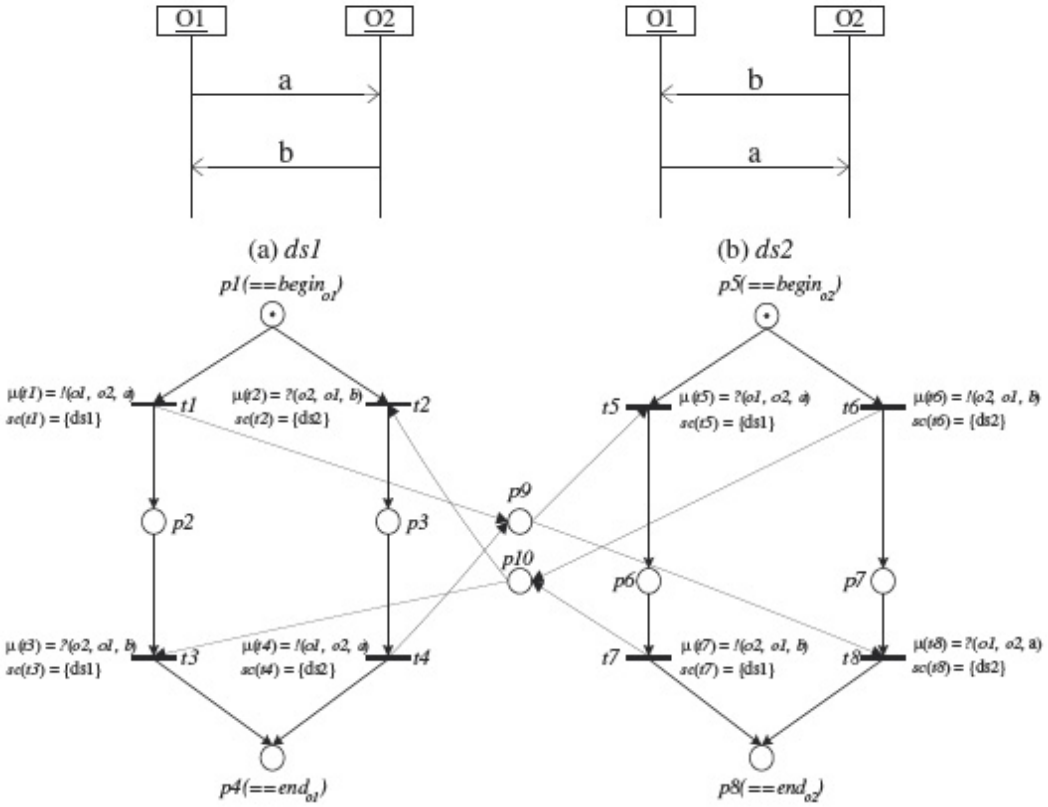
Les comportements impliqués apparaissent en particulier dans les systèmes distribués, où plusieurs objets peuvent être actifs simultanément et ne possèdent pas suffisamment d'information sur l'activité les uns des autres. Dans la spécification donnée par ds_1 et ds_2, toute implémentation des comportements des objets o_1 et o_2 permet deux autres nouvelles façons de mettre à jour les variables v_1 et v_2, non prévues par la spécification de départ, décrites par les DS ds_3 et ds_4 de la figure 8.9 (c et d) et aboutissant à un résultat non désiré.

Considérons une spécification S composée d'un ensemble de DS et son RdP associé R_s. Parmi l'ensemble des exécutions de R_s, nous cherchons à caractériser celles décrites par S, et celles n'appartenant pas à S, mais pouvant être produites par les interactions entre les objets.

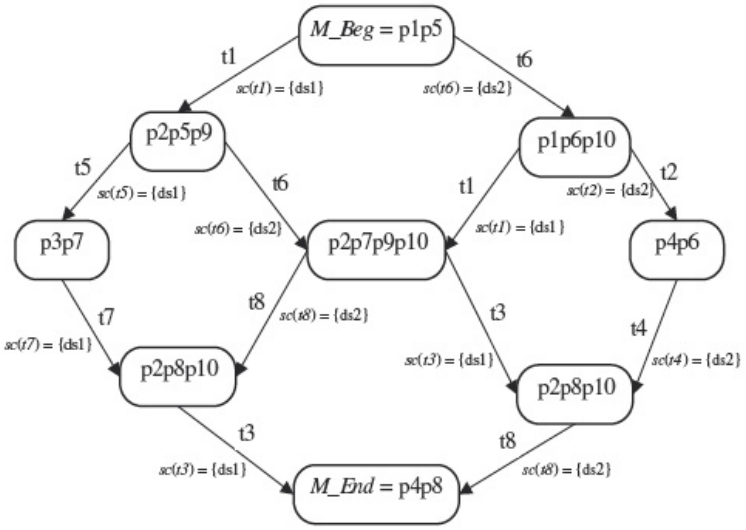
Tout d'abord, nous appellerons *calcul réussi* toute exécution associée à une séquence s de transitions dont le franchissement depuis le marquage initial M_{Beg} (toutes les places $begin_o$ marquées) conduit au marquage final M_{End} (toutes les places end_o marquées). Le calcul est dit *bloquant* si la séquence s allant de M_{Beg} à un marquage bloquant différent de M_{End}.

On peut définir l'ensemble des scénarios d'une séquence de transitions s, que l'on note $sc(s)$ comme l'ensemble des scénarios communs aux transitions de s, $sc(s) = \bigcap_{t \in \bar{s}} sc(t)$, où $sc(t)$ est l'ensemble des DS dans lesquels figure l'événement que la transition réalise (section 8.3.1) et $\bar{s}$ l'ensemble de transitions figurant en s. Si $sc(s) \neq \phi$ alors s décrit une exécution d'un DS appartenant à S, sinon s décrit une exécution *impliquée complète* de R_s qui ne correspond à aucun DS appartenant à S. Si l'exécution correspond à un calcul bloquant, elle est dite *impliquée partielle* et $sc(s) = \phi$. En termes d'événements d'émission et de réception de messages, un calcul bloquant décrit une exécution de R_s dans laquelle certains messages émis ne seront jamais reçus et/ou certains messages attendus ne seront jamais envoyés.

Une spécification par DS est réalisable si l'ensemble des exécutions qu'elle décrit est exactement le même que l'ensemble des exécutions pouvant être produites par les interactions entre les objets



c) R_s



d) graphe de marquage de R_s

FIG. 8.10 – Une spécification S, son réseau R_s et le graphe de marquage.

Calcul réussi s	$sc(s)$	Exécution représentée	Spécifiée ou impliquée
t1.t5.t7.t3	$\{ds1\}$	$!(o1, o2, a).?(o1, o2, a).!(o2, o1, b).?(o2, o1, b)$	Spécifiée par *ds1*
t6.t2.t4.t8	$\{ds2\}$	$!(o2, o1, b).?(o2, o1, b).!(o1, o2, a).?(o1, o2, a)$	Spécifiée par *ds2*
t1.t6.t8.t3	$\emptyset$	$!(o1, o2, a).!(o2, o1, b).?(o1, o2, a).?(o2, o1, b)$	Impliquée
t1.t6.t3.t8	$\emptyset$	$!(o1, o2, a).!(o2, o1, b).?(o2, o1, b).?(o1, o2, a)$	Impliquée
t6.t1.t3.t8	$\emptyset$	$!(o2, o1, b).!(o1, o2, a).?(o2, o1, b).?(o1, o2, a)$	Impliquée
t6.t1.t8.t3	$\emptyset$	$!(o2, o1, b).!(o1, o2, a).?(o1, o2, a).?(o2, o1, b)$	Impliquée

FIG. 8.11 – Calcul réussi et exécution impliquée.

qui y figurent. Les interactions entre les objets ne produisent donc pas d'exécutions impliquées complètes ou partielles : il n'existe aucun calcul bloquant dans R_s et pour tout calcul s réussi, $sc(s) \neq \phi$.

Soit la spécification S donnée par les deux DS de la figure 8.10.a, dont le réseau R_s est représenté figure 8.10.b et son graphe de marquage sur la figure 8.10.c. Sur ce graphe, l'exécution décrite par ds_1 est représentée par la séquence $t_1.t_5.t_7.t_3$ et celle décrite par ds_2 par la séquence $t_6.t_2.t_4.t_8$. Tout autre calcul réussi de ce réseau décrit donc une exécution impliquée, et c'est le cas des séquences $t_1.t_6.t_8.t_3$, $t_1.t_6.t_3.t_8$, $t_6.t_1.t_8.t_3$ et $t_6.t_1.t_3.t_8$. Ce réseau ne présente pas de calcul bloquant. La figure 8.11 montre les séquences impliquées et celles spécifiées par un DS correspondant aux calculs réussis. Ce réseau R_s n'est donc pas réalisable. La justification formelle de cette démarche est décrite en [Tah06].

Ces travaux ont montré que les réseaux de Petri permettent d'exprimer une sémantique pour les diagrammes de séquences UML dont la sémantique semi formelle rend difficile d'une part l'analyse et d'autre part le lien entre d'autres diagrammes comme le diagramme d'état transition. Il y a une relation intuitive directe entre les diagrammes de séquences et les réseaux de Petri, cependant la traduction (sémantique) entre un modèle semi formel et un modèle formel exige des choix et demeure un point critique dans la transformation des modèles. Nous avons proposé quatre sémantiques formelles selon l'interprétation donnée à l'exécution d'un message et à sa terminaison, laissant le choix à l'utilisateur. En fait, il ne faut pas oublier que les diagrammes de séquences sont utilisés dans une phase d'expression de besoins où tout n'est pas bien précisé. Il faut donner la possibilité à ces expressions d'évoluer en fournissant un outil qui puisse permettre une analyse le plus tôt possible, tout en laissant à l'utilisateur la possibilité d'être... *flou*, dans le sens de ne pas être obligé de tout préciser.

Chapitre 9

Vérification et mise en oeuvre de systèmes à événements discrets

9.1 Introduction

Selon le Petit Robert, la *coordination* est l'*agencement des parties d'un tout selon un plan logique et en vue d'une fin déterminée.*

Le problème de la planification et de la commande de systèmes à événements discrets est un problème très complexe étant données les caractéristiques combinatoires de l'espace de solution. Une solution très employée est la décomposition en niveaux hiérarchiques pour permettre de réduire la complexité. De façon classique nous rencontrons les niveaux suivants : planification, ordonnancement, *coordination* globale et pilotage, coordination de sous-systèmes et commande locale. Chaque niveau opère sur un certain horizon de temps, les contraintes temps-réel sont introduites progressivement et sont plus restrictives à mesure que l'on descend vers les niveaux les plus bas.

Dans le cas des systèmes industriels, tels que les ateliers flexibles et les stations de pompage de pétrole, le niveau *coordination* a comme fonction principale de contrôler que le fonctionnement se déroule conformément au plan détaillé calculé au préalable par les niveaux supérieurs. Mes travaux dans ce domaine se sont déroulés dans le cadre du projet *Méthodologies et outils pour la commande de procédés et l'automatisation industrielle*, financé par le pouvoir public brésilien (Projet CNPQ/AI n^o 501360/91, août 1991 à juillet 1993) qui a donné lieu à deux collaborations industrielles liées à deux encadrements de *Mestrado* : celui de M. Braga [Bra93a], *Un gestionnaire de cellule flexible de production*, et celui de F. Soares [Soa94b], *Un environnement intégré d'outils pour la conception de systèmes à événements discrets*. Ces travaux ont conduit aux publications suivantes : [Cur92, Far92, Far93, Car92, Bra93b, Soa94a].

La notion de *coordination* est aussi utilisée dans le cadre de systèmes multi-agents (SMA), métaphore sociale utilisée en IA distribuée. Dans un SMA, le but n'est plus de contrôler des modules mais de donner à des agents les moyens de s'organiser. Ces agents doivent donc interagir pour coopérer (contrôle), collaborer (allocation de tâches), négocier (résolution de conflits) et se *coordonner* (synchronisation).

Les travaux dans le cadre des SMA se sont déroulés principalement au sein d'un projet de coopération international Capes-Cofecub, entre l'équipe Systèmes à Objets Coopératifs de l'IRIT-UT1 (Institut de Recherche en Informatique de Toulouse/Université Toulouse 1) et le Departamento de

Automação e Sistemas de l'UFSC (Université Fédérale de Santa Catarina, Brésil), que j'ai animé avec G. Bittencourt (UFSC). Ce projet, *Interaction entre modèles formels pour les systèmes de supervision et les systèmes d'information*, a permis l'encadrement d'un M2R brésilien, celui de A. Postal [Pos04], et de deux thèses en cotutelle, celle de L. Frigo [Fri07] *Un outil de conception de cours adaptatifs pour les systèmes tuteurs intelligents* et celle de E. Pozzebon [Poz07] *Une Architecture multiagent pour supporter lapprentissage en groupe dans un Système Tuteur Intelligent*. Ces travaux ont donné lieu aux publications suivantes : [Fri05b, Car04a, Car04b, Fri05a, Poz06]. Une partie de ces travaux s'est inspirée de la notion de protocole de communication considéré comme un composant à part entière dans les systèmes coopératifs [Sib00, Sib01c, Sib01a].

Suivant le degré de complexité du système, différents modèles de réseaux de Petri (RdP) peuvent être utilisés. Ainsi, dans le cadre du projet *Méthodologies et outils pour la commande de procédés et l'automatisation industrielle* nous avons utilisé les RdP ordinaires. Dans le cadre de Systèmes Tuteurs Intelligents, où le système interagi avec différents étudiants au même temps, selon un même processus, nous avons utilisé les RdP à Objets [zSib85]. Ces deux volets seront détaillés par la suite.

9.2 Réseaux de Petri ordinaires pour la coordination

La révolution technologique provoquée par la banalisation (baisse des prix) des microprocesseurs a provoqué une réorganisation de l'industrie mécanique sous le nom de mécatronique, qui, en plus de la partie opérative à dominante mécanique et électromécanique, intègre dans sa partie commande l'électronique et l'informatique temps réel, prenant en compte aussi l'interface Homme/Machine. La concurrence a poussé non seulement à une plus grande flexibilité et réactivité des commandes (temps de production court, diminution de la taille des lots,...) mais surtout à un besoin en fiabilité et modularité. L'industrie s'est alors appuyée sur les des centres de recherche pour développer des méthodes de conception plus formelles de tels systèmes. Les collaborations entre industrie et université[1] ont bouleversé aussi l'enseignement avec la création en 1988 à l'UFSC de la première formation, au Brésil, en Génie en Automatique et Informatique (en Brésilien, engenharia de controle e automação, synonyme de mécatronique au Brésil). C'est dans ce contexte que mes travaux, de nature essentiellement appliquée et fondés sur des opérations de transfert vers le secteur industriel, se sont effectués. Ils ont donné lieu à deux encadrements de *Mestrado*, celui de M. Braga et de F. Soares.

M. Braga était étudiant du *Mestrado en Génie Mécanique* de l'UFSC et son stage était lié à une collaboration avec l'industrie de fabrication de moteurs Weg Motores S.A.. Le travail consistait à implémenter un logiciel pour réaliser la coordination d'une cellule de production constituée d'un robot IPSO V-15, un tour CSEPEL avec CNC Dynapath S-20 et un micromètre Laser Olsen, intégrés par un bus selon la norme DIN Profibus. L'entrée des pièces dans le magasin d'entrée est faite par un opérateur. Le robot doit enlever la pièce du magasin d'entrée, la déposer dans le tour (s'il est libre) ; après l'usinage, la pièce est amenée par le robot au micromètre (s'il est libre) pour inspection. Après cette opération le robot dépose la pièce dans le magasin de sortie d'où elle est retirée par l'opérateur. Le logiciel de coordination, appelé *Gestionnaire* devait s'exécuter dans un ordinateur compatible IBM-PC. Les fonctionnalités principales de ce Gestionnaire étaient de :

[1] Au Brésil, les centres de recherche sont peu nombreux, et pratiquement toute la recherche est faite dans des laboratoires appartenant intégralement aux universités et composés donc d'enseignants chercheurs.

– réaliser la gestion d'une base de données avec les programmes et paramètres responsables pour le transport, la manipulation, l'usinage et inspection ainsi que les informations sur le carnet de commande ;
– indiquer la prochaine pièce qui doit entrer dans la cellule selon une politique d'ordonnancement ;
– interagir avec l'opérateur de la cellule (arrivée d'une nouvelle pièce dans le magasin d'entrée, demande de retrait d'une pièce du magasin de sortie, indication des anomalies, etc) ;
– coordonner l'opération de la cellule à partir de la connaissance de l'état de chaque élément ;
– accéder aux services du réseau de façon à envoyer les programmes et paramètres pour chaque élément, demander l'exécution de ces programmes et vérifier leur fin d'exécution.

Les tâches du système correspondent à cette décomposition fonctionnelle et l'interaction entre les tâches ainsi que le parallélisme des activités ont conduit à l'utilisation des réseaux de Petri pour la spécification et vérification du Gestionnaire [Bra93b]. En fait, une première version de ce Gestionnaire avait été faite directement en C et avait beaucoup de problèmes. La validation a été réalisée par ARP, un logiciel développé au sein de notre laboratoire dans un autre travail de mestrado, et a permis de corriger beaucoup d'erreurs de cette première version (réalisée directement en C). Nous avons utilisé pour le prototypage le logiciel SPP (développé aussi dans un mestrado dans notre laboratoire) qui générait un code C exécutable à partir d'un système de règles, en faisant une traduction (manuelle) des réseaux de Petri déjà vérifiés en systèmes de règles. Le code ainsi obtenu a été réutilisé à environ 70%.

Dans le cadre d'une convention avec la société Petrobrás (Pétrole du Brésil) un ingénieur (F. Assis) a été détaché dans notre laboratoire pour améliorer les techniques utilisées dans cette entreprise pour l'automatisation des stations de pompage de pétrole dans le Nord-est du Brésil. Le choix de la modélisation de la coordination de ces stations par réseaux de Petri étant fait, plus l'existence des outils de vérification de ces réseaux d'un coté, et de génération de code (à partir d'un système de règles) de l'autre, mais de façon séparée, cela nous a conduit à la conception et mise-en-œuvre d'un environnement intégré pour la spécification de systèmes à base de réseaux de Petri [Soa94a, Soa94b], IET, Integrated Environnement Tool. Bien que ses travaux aient débuté avant ceux de M. Braga, l'environnement à été développé plus tard car le travail de F. Assis s'est étalé sur plusieurs périodes disjointes.

Cet environnement permet, à partir d'un réseau de Petri interprété – où l'interaction avec l'environnement est représenté par des conditions et des actions (capteurs, actionneurs, messages envoi/réception) associées aux transitons et où le temps est pris en compte implicitement – de valider (en utilisant ARP), simuler et émuler la coordination d'un système. Le système physique à contrôler (ou procédé) peut être directement contrôlé par un joueur de réseau de Petri ou par un code C généré automatiquement à partir du réseau. Une validation préliminaire du niveau coordination est faite à partir du réseau de Petri sous-jacent (en enlevant toutes les conditions et actions). Même si les évolutions du réseau interprété sont des restrictions des évolutions du réseau sous-jacent, il faut aussi une étape de simulation pour valider le réseau interprété ; il est ainsi nécessaire de modéliser et simuler le procédé pour pouvoir prendre en compte l'interaction avec ce système.

L'architecture de l'IET est composée de quatre modules : module Édition, module Analyse, Module Simulation et Module Traduction. Le module Édition permet de décrire le réseau de Petri interprété (RPI) du contrôle et du système à contrôler, en utilisant un éditeur de texte ou lire un fichier avec la description du réseau, avec analyse syntaxique et sémantique ; en plus du fichier représentant le RPI, un fichier représentant le réseau sous-jacent est aussi généré. Ce réseau sous-jacent est analysé par le module Analyse, implementé par le logiciel ARP qui réalise l'analyse à partir du graphe de marquages et de l'analyse d'invariants. Le module Simulation permet la simulation du

contrôle et l'émulation de procédé. La simulation peut être interrompue par l'utilisateur à tout moment ; l'état et le temps (virtuel) de la simulation sont sauvés et peuvent être restaurés pour reprendre la simulation s'il le souhaite. Une fois la validation par analyse et simulation faite, le module Traduction permet de générer le code source en C correspondant au réseau interprété du contrôle, en utilisant le logiciel SPP. Comme ce logiciel accepte en entrée un système de règles, le module Traduction traduit le réseau de Petri interprété en un système de règles.

L'environnement IET a été appliqué à l'automatisation d'une station de pompage de pétrole similaire aux champs de pétrole terrestres exploités par la Petrobrás dans le Nord-est du Brésil. La fonction d'une station de pompage est d'obtenir le pétrole cru provenant des puits de pétrole environnants, vérifier le flux de sortie (outflow) des puits, et transférer au moyen de pompes de transfert le pétrole ainsi produit aux stations de plus grande capacité par des canalisations (pipeline). Le procédé est composé d'un réservoir R recevant continuellement le pétrole à partir des canalisations venant des puits ; quand le pétrole atteint un niveau maximum H une pompe P_1 est mise en marche et démarre le transfert du pétrole vers la station collectrice centrale. Si cette pompe ne démarre pas dans 2 secondes, une deuxième pompe P_2 doit être mise en marche. Si le pétrole dans le réservoir descend en dessous d'un niveau minimum L, la pompe doit être arrêtée. Elle doit aussi être arrêtée si la pression augmente au delà d'un niveau de sécurité, qui peut correspondre au blocage de la canalisation descendante. Ce signal est ignoré s'il a eu lieu 30 secondes avant l'opération de pompage. L'automatisation de la station nécessite l'implementation des services suivants : logique d'interblocage des commutateurs de niveau et de pression ; mesure du volume de pétrole produit ; acquisition et transmission de donnés à la station centrale responsable pour la supervision. À l'époque, ces tâches étaient partiellement implementées dans un automate programmable dans chaque station de pompage. La programmation de ces automates étant faite à l'aide de diagrammes à échelles, aucune validation sérieuse ne pouvait être effectuée.

Le travail qui a été réalisé dans ce cadre est un travail typique de transfert. Il a mis en évidence l'applicabilité de l'utilisation des réseaux de Petri ordinaire pour la supervision et la coordination de systèmes automatisés. Il est intéressant de remarquer que la première mise en oeuvre des réseaux s'est faite sous la forme de règles, tout simplement parce que l'outil était disponible. Ce travail a également permis de constater les limites des réseaux de Petri ordinaires dès qu'il était nécessaire de manipuler des données de façon plus explicites et de s'intégrer à des systèmes de règles plus sophistiqués. Cela a motivé les études sur les Systèmes Multi-Agents quelques années plus tard. Par ailleurs, dans le cadre du projet *Méthodologies et outils pour la commande de procédés et l'automatisation industrielle*, mené en collaboration avec les collègues du département de Génie en Automatique et Informatique, je me suis impliquée dans la coordination des Systèmes Flexibles de Manufacture en utilisant les réseaux de Petri de Haut Niveau [Cur92, Far92, Far93, Car92].

9.3 Conception de Systèmes Coopératifs et Systèmes Tuteurs Intelligents (STI)

J'ai commencé à travailler sur les SMA en 1999, l'année où j'ai rejoint l'équipe *Systèmes à Objets Coopératifs* à l'UT1/IRIT. Les agents[2] d'un SMA doivent interagir pour coopérer (contrôle), col-

[2]Selon Ferber [zFer99], un agent est une entité réelle ou virtuelle, évoluant dans un environnement, capable de le percevoir et d'agir dessus, qui peut communiquer avec d'autres agents, qui exhibe un comportement autonome, lequel peut être vu comme la conséquence de ses connaissances, de ses interactions avec d'autres agents et des buts qu'il poursuit.

laborer (partage de tâches), négocier (résolution de conflits) et se *coordonner* (synchronisation). Ces différentes interactions sont basées sur le concept de *protocole*, qui doit être souvent *ouvert* – le nombre de participants n'est pas limité a priori et certaines étapes du protocole admettent le départ et/ou l'arrivée de participants – ou bien admettre un grand nombre de participants.

Dans un travail conjoint initié par C. Sibertin-Blanc et C. Hanachi nous avons proposé une architecture de SMA qui considère les protocoles comme des entités à part entière [Sib00, Sib01c, Sib01a]. Une *conversation* n'est qu'une exécution particulière d'un protocole ; elle met en jeu des *interventions* effectuées par les entités et le *contrôle* vérifiant que ces interventions respectent les règles du protocole. Nous avons choisi de confier le contrôle des conversations à des entités spécialisées, appelées *Modérateurs*, en le dissociant ainsi des interventions. Le Modérateur est chargé, pour chaque conversation, de gérer le déroulement de la conversation et de garantir qu'elle respecte les règles du protocole. Il est une instance d'un type de Protocole, crée dynamiquement par un Agent selon ses besoins de communication et disparaît à la fin de la Conversation. Les Agents adressent leurs interventions au Modérateur, et non pas directement aux autres Agents ; ils n'ont besoin de connaître que les primitives du protocole, allégeant ainsi leur code de tout ce qui concerne la synchronisation. Les modérateurs sont mis en oeuvre par des *Objets CoOpératifs* [zSib99], un langage à objet totalement concurrent basé sur les réseaux de Petri. En plus du Modérateur, un nouvel élément a été ajouté à l'architecture SMA : l'*Entremetteur*, ou Facilitateur de Conversations, qui permet aux Agents de récupérer des informations sur les Conversations en cours et d'être informés des nouvelles Conversations.

Par la suite j'ai proposé, avec G. Bittencourt, un projet Capes/Cofecub, *Interaction entre modèles formels pour les systèmes de supervision et les systèmes d'information*, dont l'objectif est la formation. Il a débuté en janvier 2002 et a été renouvellé à 2 reprises jusqu'à décembre 2005.

Le thème central de ce projet était l'étude de différents formalismes pour traiter les problèmes de modélisation, et plus particulièrement ceux liés aux interactions, dans les systèmes complexes - systèmes de contrôle d'automatismes ou systèmes d'information distribués. Le projet est basé sur les systèmes multiagents - qui intègrent les technologies du génie de logiciel, des systèmes distribués et de l'intelligence artificielle. Les deux laboratoires étaient complémentaires vis-à-vis de ces technologies. Nous avons travaillé plus précisément sur les Systèmes Tuteurs Intelligents (STI).

9.3.1 Architecture classique d'un STI

Les systèmes d'Enseignement Assisté par Ordinateur (CAI – Computer Aided Instruction) sont apparus dans les années 50. L'interaction entre ces systèmes et les utilisateurs était fixe et donc il n'y avait pas de possibilité d'adaptation. L'utilisateur peut seulement *tourner les pages*. À partir des années 70, avec les techniques d'Intelligence Artificielle et de la Psychologie Cognitive, ces systèmes ont évolué vers l'Enseignement Assisté par Ordinateur Intelligent (ICAI – Intelligent Computer-Aided Instruction). Or, le point principal des systèmes éducatifs impliquant l'intelligence artificielle est le passage de la programmation des décisions à celle des connaissances. De là un changement méthodologique radical entre l'EAO et le STI. La notion centrale devient celle de modèle [zWen87].

L'architecture classique d'un STI est composée de trois modèles, ou modules, comme représentée figure 9.1 :

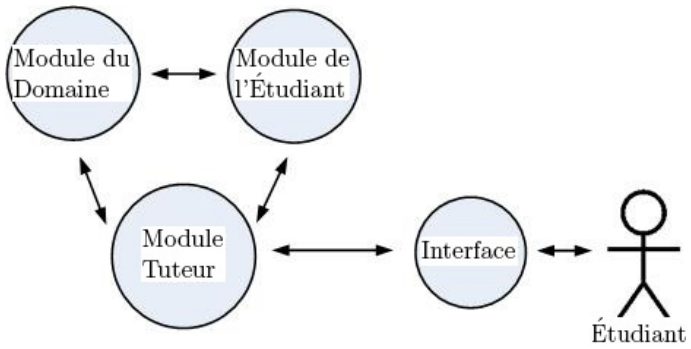


FIG. 9.1 – Architecture classique d'un STI.

Modèle du Domaine : il contient les informations sur un domaine donné, organisé de façon à représenter la connaissance d'un spécialiste ou enseignant.

Modèle de l'Étudiant : il décrit les informations sur un étudiant, ses préférences et l'historique de ses activités ce qui est fondamentale pour le mécanisme d'adaptation. Sa construction est une tâche très complexe.

Modèle Pédagogique : il utilise les informations du modèle de l'Étudiant et du modèle du Domaine pour déterminer quelle information présenter à l'étudiant et comment la présenter.

La recherche dans ce domaine est très active, avec plusieurs systèmes : ADIS [zWar97], CALAT [zNak97] I-Help [zVas01], et, parmi ceux en cours de développement : REDEEM [zAin06], CTAT [zAle06] et XTA [zNuz05]. Une comparaison entre les trois derniers outils permet de vérifier qu'aucun des trois ne permet une navigation adaptative (selon le profil de l'étudiant). Le CTAT et le XTA utilisent les techniques d'IA comme les règles de production et les graphes de prérequis pour la conception de cours. Les systèmes CTAT et REDEEM offrent différentes stratégies. L'Hypermédia adaptatif est aussi un domaine qui s'intéresse aux systèmes éducatifs impliquant l'intelligence artificielle [zBru03] comme Trellis [zNa01], Morena [zBot05], LAOS [zCri04] et HTSPN [zWil02].

Les limitations majeures des STI actuels restent encore essentiellement le manque de modèles et d'outils pour la conception de cours avec de tels systèmes, la difficulté du partage et de la réutilisation de leurs composants logiciels ainsi que la distance entre la planification des cours (faite par le professeur) et la stratégie du système tuteur pour l'adaptation dynamique du comportement de l'ITS. La principale contribution des travaux décrits section 9.3.2 est justement de diminuer cette distance.

9.3.2 Conception de cours d'un STI

L'objectif de nos travaux dans le domaine de STI était celui de définir un cadre méthodologique assorti d'outils informatisés pour proposer une solution au problème de l'adaptation dans les STI à partir de la modélisation des interactions entre le système tuteur et l'étudiant.

Nos travaux se sont situés d'abord dans un cadre où plusieurs étudiants interagissent *individuellement* avec un STI, localement ou à distance. Nous avons proposé un cadre méthodologique assorti d'outils informatisés, pour le développement de STI s'adaptant aux interactions avec l'élève, permettant au professeur (l'auteur d'un cours) de spécifier le contenu d'un cours sans avoir à se préoccuper des mécanismes d'adaptation de ces interactions. Plus précisément, ce cadre méthodologique de création de cours doit permettre une intégration automatique du contenu fourni par le professeur

avec un mécanisme d'adaptation à l'élève que prend en compte ses caractéristiques individuelles et les résultats de ses performances durant son interaction avec le système. Ce cadre méthodologique a été mis en oeuvre par un outil informatisé, nommé FAST (Ferramenta de Autoria para Sistemas Tutores en portugais, ou *Outil pour la Conception de Cours dans un Système Tuteur Intelligent*). FAST permet de générer la description d'un STI spécifique à un domaine avec l'utilisation des réseaux de Petri à Objets. Néanmoins, l'utilisation de ce formalisme est transparente pour le professeur et l'étudiant. Au final, l'auteur d'un cours doit uniquement fournir le modèle du domaine, accompagné d'un graphe des prérequis entre les unités du cours à partir duquel le réseau de Petri est automatiquement généré. Enfin, un outil informatisé, nommé ASTI (Arcabouço para Sistemas Tutores Inteligentes en portugais, *Shell pour Systèmes Tuteurs Intelligents*), a été partiellement réalisé pour permettre le développement de STI en mettant en oeuvre le cadre méthodologique FAST. Nous avons privilégié l'approche orientée agents et une exploitation d'ontologies de domaine dans la définition du cadre méthodologique FAST ainsi que dans la conception de l'outil ASTI.

Nous avons utilisé le modèle Mathema [zCos95] comme support pour structurer la représentation du domaine de connaissance en un STI. En prenant en compte des critères tels que point de vue, niveau de complexité et connaissances complémentaires, le domaine est décomposé en *sous-domaines*. Ensuite un sous-domaine est organisé en un ensemble de curricula. Chaque curriculum est progressivement raffiné selon trois niveaux de détail : les Unités Pédagogiques, les Problèmes et les Unités d'Interaction (exemples, exercices et explications). Les unités pédagogiques et les problèmes sont partiellement ordonnés. Chaque problème est associé à un ensemble d'Unités d'Interaction. Mathema propose aussi une architecture fonctionnelle des STI avec la définition d'une Société des Agents Tuteurs Artificiels (SATA). La SATA est un système multi-agents dont le nombre d'agents est fonction de la décomposition à partir de la vue externe ; chaque agent est spécialisé dans un sous-domaine du domaine cible et a lui même une architecture classique comme représentée figure 9.1.

Cependant, le coeur de ces travaux est la méthodologie qui a donné lieu à l'Outil pour la Conception de Cours dans un STI (FAST). La FAST permet, à partir de la connaissance d'un sous-domaine de générer le contenu des modèles d'un agent de la SATA. Les principaux choix que nous avons effectués sont les suivants [Fri05a, Fri05b] :
- représentation de la connaissance décrivant les modèles du domaine et de l'étudiant, ainsi que leurs relations, en utilisant les ontologies ;
- l'utilisation du formalisme des Réseaux de Petri à Objets (RPO) [zSib85] pour spécifier le modèle pédagogique du système (les transitions contrôlent l'enchaînement des interactions selon des conditions portant sur le modèle de l'étudiant, et les choix faits par l'étudiant durant l'interaction) ;
- l'utilisation d'un contrôle multi-niveaux pour augmenter la flexibilité du système.
- l'utilisation des règles de production pour la mise en oeuvre du comportement de l'agent.

Le modèle conceptuel de FAST est composé de :
- une interface interactive où le professeur-auteur saisit la structure du (sous-)domaine sous la forme de graphes et son contenu (celui-ci va constituer le *modèle du domaine*),
- un compilateur qui transforme automatiquement la structure du (sous-)domaine en un réseau de Petri à objet qui contrôle le *modèle Pédagogique* de l'Agent Tuteur et qui intègre les informations du *modèle de l'Étudiant*.

Pour augmenter la flexibilité des comportements possibles du *modèle Pédagogique*, nous avons utilisé un contrôle en deux niveaux : le niveau du *Curriculum* et le niveau des *Stratégies*. Le niveau du Curriculum, décrit par le réseau RPO-CV, spécifie les parcours possibles du cours selon les Unités Pédagogiques et les Problèmes, en conformité avec les prérequis définis par l'auteur sous

la forme d'un graphe. L'auteur doit définir, pour chaque Curriculum le graphe de prérequis des Unités Pédagogiques (UP), et pour chaque UP le graphe de prérequis des Problèmes la composant. Le niveau des Stratégies, décrit par le réseau RPO-E, spécifie l'interaction entre l'étudiant et le STI durant la résolution d'un problème spécifique. Il présente à l'étudiant les Unités d'Interactions associées à un problème. Les niveaux Curriculum et Stratégies interagissent par le mécanisme de passage de message utilisant les places de communication, comme représentés sur la figure 9.2 (les autres places Begin_{out} et End_{in} de chaque transition du RPO-CV ne sont pas représentées pour la clarté de la figure).

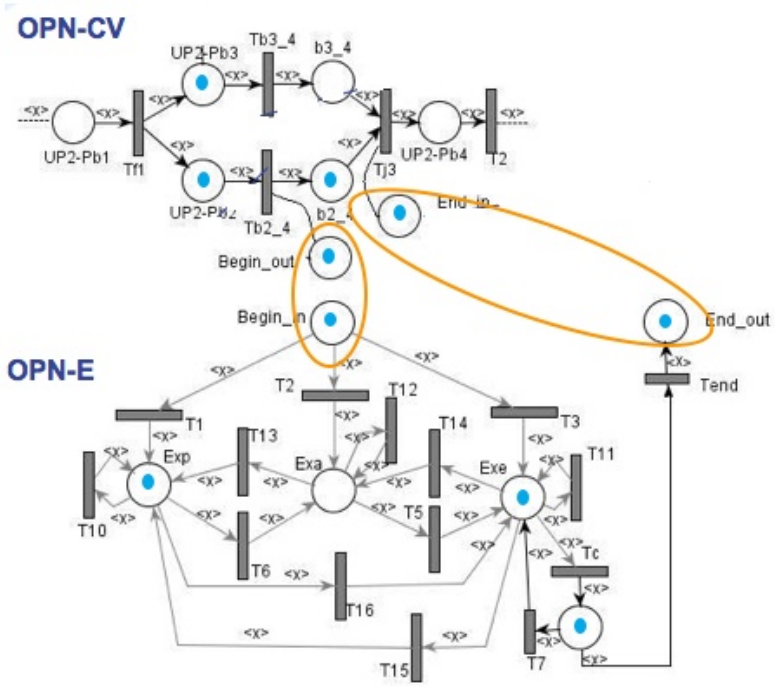


Fig. 9.2 – Modèle fonctionnel de FAST

La figure 9.3 représente le modèle fonctionnel de FAST : le graphe de prérequis fourni par l'auteur est automatiquement traduit en un réseau RPO-CV [Pos04, Car04a]. La structure de contrôle du réseau est donnée par ce graphe ; la structure de données associée aux jetons est celle de la classe *Étudiant*, obtenue à partir de la structure du Modèle de l'Étudiant, lui aussi proposé par le concepteur (figure 9.2). Les variables associées aux arcs sont typées par la classe *Étudiant*. Les pré-conditions et les actions des transitions portent sur les attributs des objets instantiés. Cela permet de modéliser plusieurs étudiants interagissant simultanément avec le STI et suivant différents parcours, en prenant en compte leurs caractéristiques et en faisant la mise à jour de leur interactions avec le système.

Si c'est l'auteur qui fournit le graphe représentant les prérequis du domaine qui seront traduits par le réseau RPO-CV, c'est le concepteur qui fournit le réseau RPO-E du niveau Stratégies ainsi que la structure du Modèle de l'Étudiant. En fait, le concepteur fournit une bibliothèque de tels réseaux, permettant de modéliser différentes stratégies qui seront proposées à l'étudiant, soit directement, soit par analyse de son modèle, permettant la définition de différents scénarios d'apprentissage.

Du point de vue de l'implémentation des Agents Tuteurs, le choix a été porté sur la plate-forme Jade et le comportement des Agents est exécuté par JESS (Java Expert System Shell) [zFrie03]. Il a donc fallu traduire les deux réseaux de Petri RPO-CV et RPO-E en deux bases de règles JESS,

JESS-CV et JESS-E (figure 9.3), qui sont exécutées par deux instances du moteur d'inférence JESS. Le mécanisme de communication par passage de message est traduit par l'addition de règles aux deux bases.

L'outil FAST (figure 9.3) génère, à partir des donnés d'un sous-domaine (le graphe de prérequis des problèmes et des unités pédagogiques d'un curriculum), le Modèle Pédagogique de l'Agent Tuteur de la SATA responsable pour ce sous-domaine, sous la forme de deux bases de règles qui vont déterminer son comportement lors de l'interaction avec l'étudiant. Le modèle du Domaine contient simplement le contenu de chaque problème (exercices, exemples, explications), donné par l'auteur et qui sera présenté à l'étudiant sous la forme de pages WEB. Le modèle de l'Étudiant correspond à la structure proposée par le concepteur et qui sera remplie lors des interactions de l'étudiant avec le système.

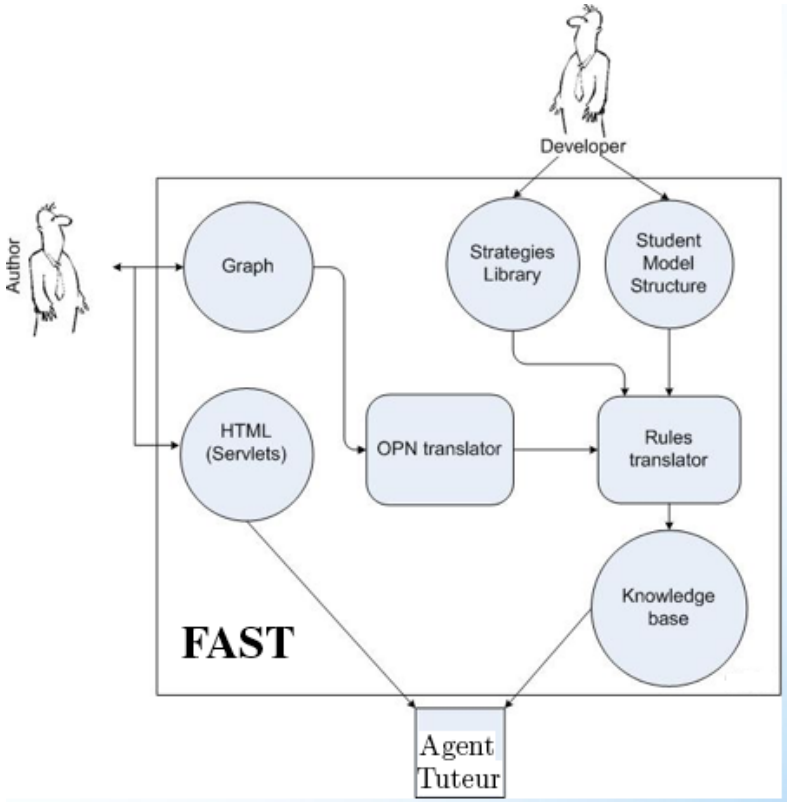


Fig. 9.3 – Modèle fonctionnel de FAST

Comme nous avons dit section 9.3.1, une des limitations des STI est la difficulté de partage et de réutilisation de leurs composants logiciels. C'est pourquoi nous avons proposé un *Shell* pour Systèmes Tuteurs Intelligents, ASTI (pour Arcabouço para Sistemas Tutores Inteligentes en portugais). Ce shell, représenté sur la figure 9.4, s'inspire et étend l'architecture proposée par Mathema. Ses éléments sont construits à l'aide de l'outil de conception de cours FAST qui fournit la structure du modèle du Domaine, du modèle Pédagogique et de celui de l'Étudiant de chaque Agent Tuteur de la SATA (les autres éléments de FAST sont représentés sur la figure 9.3) ; il est encore en cours de développement. Nous avons apporté des modifications à l'architecture Mathema pour l'adapter à notre démarche. Par exemple, le motivateur externe, appelé maintenant professeur-moniteur, interagit aussi avec la SATA ; la Société des Spécialistes Humains a été divisée en deux entités qui ont des rôles différents : i) le concepteur qui crée la bibliothèque des stratégies et la structure du modèle de l'étudiant (fig. 9.3) ; ii) le professeur-auteur qui fourni le contenu du modèle du Domaine

et contribue à la construction du modèle Pédagogique (fig. 9.3).

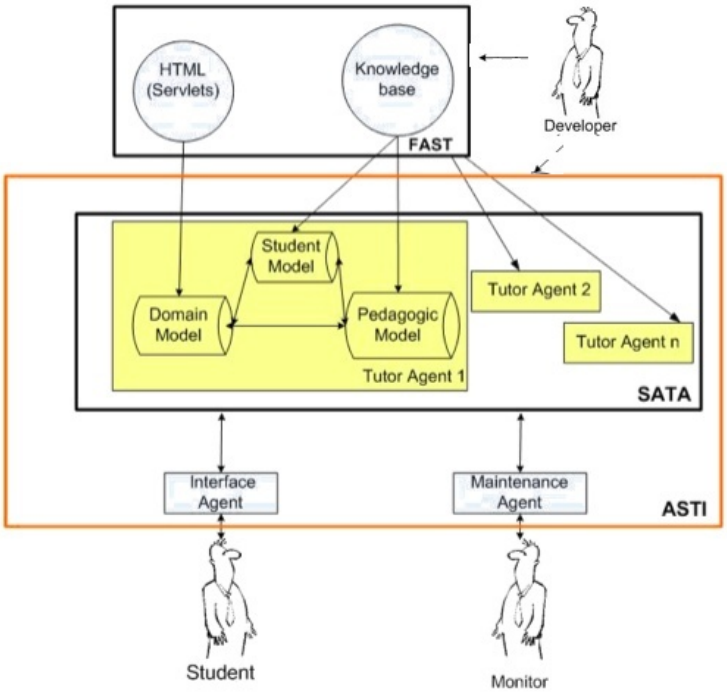


FIG. 9.4 – ASTI : *Shell* pour Systèmes Tuteurs Intelligents

Le fonctionnement d'un STI construit avec l'ASTI peut être décrit de la façon suivante. D'abord un curriculum est sélectionné parmi l'ensemble de curricula associés au sous-domaine, soit automatiquement à partir des informations du modèle de l'Étudiant soit par choix direct de l'étudiant. Ensuite, tant qu'il y a des problèmes associés à ce curriculum à proposer à l'étudiant :

- la machine d'inférence indique les problèmes qui peuvent être présentés à l'étudiant à cette étape ;
- à partir du choix effectué par l'étudiant, l'agent d'interface (qui représente l'étudiant) extrait les informations associées au problème choisi et envoie à l'interface sous la forme d'une page HTML ;
- s'il n'y a pas d'interactions externes, par exemple, un environnement de programmation, le modèle de l'Étudiant est mis à jour avec le résultat de l'interaction.

Les principales contributions du cadre méthodologique FAST sont :

1. du point de vue du concepteur : l'utilisation de RPO pour coupler les modèles de l'architecture de STI, facilitant la visualisation des trajectoires possibles que l'étudiant peut effectuer, afin d'éviter des chemins inconsistentes ; l'utilisation des ontologies pour la représentation des modèles de domaine et de l'étudiant, facilitant l'inspection, la manutention et la réutilisation.
2. du point de vue du professeur-auteur : la réduction du travail exhaustif et improductif, atténuant la complexité inhérente au processus enseignement-apprentissage, lui permettant de se préoccuper seulement de l'organisation du contenu.
3. du point de vue de l'étudiant : l'utilisation d'un environnement complexe capable de s'adapter à ses besoins.

9.3.3 Apprentissage en groupe dans un STI

Le *Shell* pour Systèmes Tuteurs Intelligents (ASTI) décrit section 9.3.2 construit à partir de l'outil de conception de cours FAST permet l'interaction individuelle avec un étudiant. Nous avons voulu aussi permettre l'interaction des étudiants sous la forme des groupes, un étudiant pouvant participer à plusieurs groupes différents. Dans ce cas le STI doit non seulement présenter le (sous-)domaine à chaque étudiant mais aussi traiter de l'interaction de l'étudiant avec le groupe. L'objectif est celui de transposer la réalité de la salle de cours aux STI. En effet, l'expérience montre que les personnes travaillant en groupe apprennent d'avantage. Le groupe peut produire un effet de stimulation sur le comportement individuel. En général, on peut distinguer au moins deux types de groupe dans une salle de cours : les groupes *spontanés* où les étudiants se rencontrent pour étudier une discipline (pour l'entre-aide), et les groupes formés par le professeur pour accomplir une tâche. Plusieurs questions sont donc apparues, en considérant que l'ASTI permet aussi l'interaction des étudiants distants géographiquement (interaction par le réseau) :

– le degré de contrôle de l'interaction : le système peut fournir des outils de travail coopératif (chat, mail, forum, etc.) et laisser la coordination sous la responsabilité du professeur-moniteur ou bien il peut contrôler complètement l'interaction en suivant des protocoles bien définis ;
– type de groupe : les membres d'un groupe doivent avoir le même niveau (groupe par intérêt), ou avoir un animateur qui puisse aider dans la résolution des problèmes ?
– la création d'un groupe : proposition automatique du STI à partir d'interactions individuelles (un étudiant peut créer un groupe et demander au STI de "chercher" des membres).

Nous avons choisi de ne développer une approche que pour traiter de la première question, et de proposer une méthode pour la spécification et l'exécution de la gestion des groupes dans un STI. C'est un compromis entre un contrôle complètement défini par des protocoles et un contrôle par le professeur. Ces travaux s'effectuent dans le cadre de la thèse en cotutelle de E. Pozzebon [Poz07]. Nous avons proposé une méthode qui utilise un langage de spécification formelle permettant la définition de protocoles d'interaction complexe, que nous appelons *scénarios* dans le but d'avoir des interactions à la fois flexibles et fiables [Poz06]. Ces scénarios sont spécifiés par le concepteur de l'outil de conception de cours pour permettre d'offrir une éventail d'activités de groupe. Le professeur-auteur doit fournir le contenu et adapter le scénario choisi.

Cela implique de créer un nouveau shell pour STI permettant l'apprentissage en groupe, que nous nommons ASTI-G, bâti sur ASTI. Comme le comportement de chaque Agent Tuteur appartenant à l'ASTI est généré à partir de l'outil FAST, il faut d'une part étendre cet outil pour permettre à l'auteur et au concepteur de rentrer les informations nécessaires aux activités de groupe, et d'autre part créer une structure pour gérer ces activités au sein de l'ASTI-G.

Comme ASTI-G permet toujours l'interaction individuelle, le comportement de l'agent responsable pour l'interaction avec un étudiant est celui présenté section 9.3.2. Cependant, il faut étendre la structure du modèle de l'Étudiant pour prendre en compte les activités de groupe (préférences, enregistrement des interactions, etc.) et surtout créer un nouveau modèle pour prendre en compte l'interaction par groupe, le *Modèle d'Activité de Groupe*. FAST devient maintenant un outil de conception de cours et d'activités de groupe, appelé FAST-G. Le Modèle d'Activité de Groupe et le Modèle de l'Étudiant (étendu) définissent le comportement des agents responsables pour les activités de groupe.

L'Activité de Groupe implique le *concepteur* (qui spécifie la blibliothèque de scénarios), l'*auteur* (qui choisit et adapte un scénario) et le *moniteur* qui supervise l'activité du groupe, déterminant le début et la fin de l'activité, ainsi que l'interaction avec les étudiants. Chaque activité de groupe est

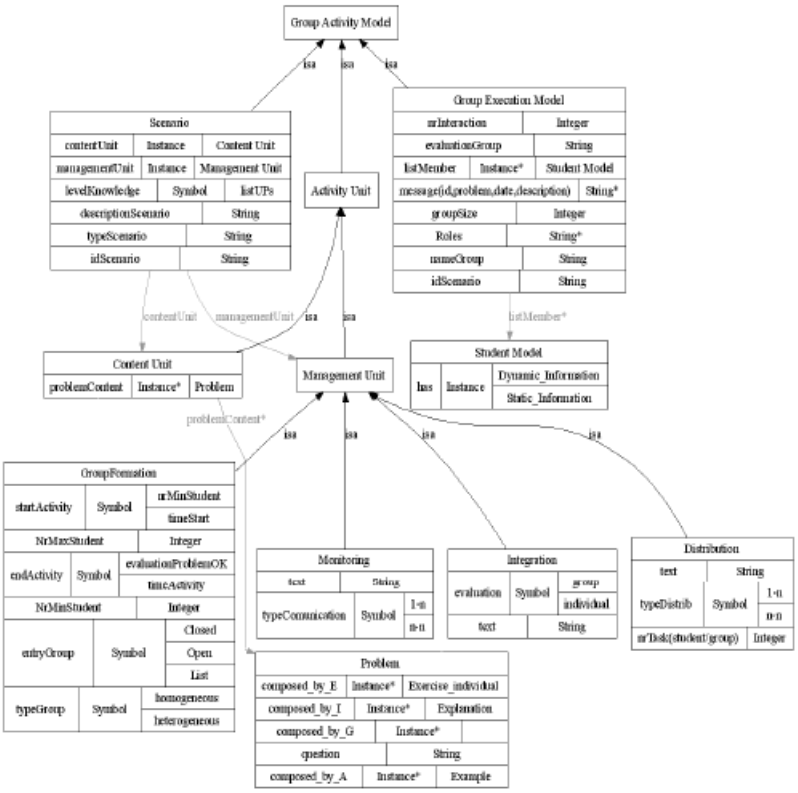


FIG. 9.5 – Modèle d'Activité de Groupe

associée à un ITS sous-jacent, généré pas FAST (section 9.3.2). La figure 9.5 représente les concepts impliqués dans l'Activité de Groupe : Groupe, Scénario, Role, Prérequis, Unités d'Activité (unités de Gestion et de Contenu) et Protocole d'Interaction. Le concept de Groupe est lié à un ensemble d'étudiants déjà inscrits à un ITS sous-jacent et éventuellement un moniteur. Le concept de Scénario est la définition opérationnelle d'une activité de groupe et inclut les prérequis, les unités d'activités et un protocole d'interaction. L'unité de Gestion définit les activités typiques d'interaction de groupe : formation du groupe, distribution de problèmes, attente d'une solution, attente de toutes les solutions, etc. L'unité de Contenu définit les tâches de solution de problèmes associées à un scénario donné. Elles sont basées sur la notion de Problème (et ses Unités d'Interaction) du modèle de Domaine du STI sous-jacent, qui sert donc aussi bien à l'interaction individuelle qu'à l'interaction par groupe. Le Protocole d'Interaction spécifie l'ordre dans lequel l'activité sera exécutée dans un scénario donné et, de façon similaire au Modèle Pédagogique, il est aussi représenté par un réseau de Petri à Objets (RPO) à deux niveaux. À partir d'un scénario choisi (spécifié par RPO), un compilateur intègre le modèle du domaine du STI sous-jacent ainsi que le modèle (étendu) de l'étudiant, générant ainsi un protocole adapté à l'étudiant.

Le niveau exécution du modèle d'Activité de Groupe est défini par une architecture multi-agent inspirée par Ferber [zFer03]. Il est représenté figure 9.6. Ces agents sont associés à trois rôles : Agent Étudiant (AE), qui représente l'étudiant ; Agent Superviseur de Groupe (ASG) supervise l'activité de groupe selon le RPO associé au protocole d'interaction défini par l'instance de scénario ; Agent Coordinateur, responsable pour la création et destruction des Agents Superviseurs de Groupe durant l'exécution. Il serait intéressant aussi qu'il puisse suivre l'apprentissage individuelle pour détecter et proposer des activités d'apprentissage par groupe. Le moniteur peut suivre les activités de groupe par l'intermédiaire de cet agent.

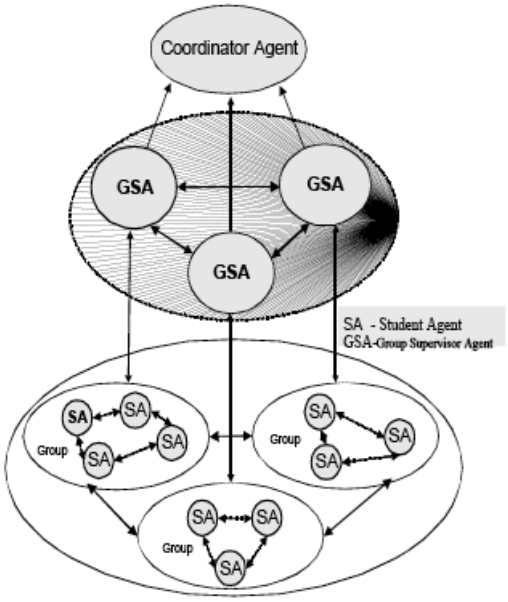


FIG. 9.6 – Architecture du Modèle d'Activité de Groupe

Les deux rôles, Agent Superviseur de Groupe et Agent Coordinateur sont très liés, respectivement, aux concepts de Modérateur et Entremetteur présentés en [Sib00, Sib01c, Sib01a]. En fait, l'interaction au sein d'un groupe est vue comme une conversation, considérée comme une exécution particulière d'un protocole. Dans ce travail le protocole n'est pas complètement spécifié à priori, mais est adapté au groupe. Ces travaux sont encore en phase de conception et de début d'implementation, la soutenance de E. Pozzebon étant prévue pour 2008.

La leçon que l'on peut tirer de ces travaux sur les STI du point de vue d'un chercheur s'intéressant d'abord aux applications possibles des réseaux de Petri, c'est que cet outil peut être utilisé sans problème dans un environnement *Intelligence Artificielle*. Dans ce cadre il permet au niveau conceptuel d'exprimer des scénarios et des politiques d'interaction entre des agents. En fait il permet d'isoler une vision *Systèmes à Événements Discrets*, c'est-à-dire comportementale, des autres connaissances. Quand on arrive à la mise en oeuvre, cette séparation entre les connaissances comportementales et les autres connaissances peut s'effacer et les transitions du réseau de Petri deviennent alors des règles au milieu d'autres règles.

Chapitre 10

Spécification et vérification de contraintes floues

10.1 Introduction

Nous avons vu au chapitre 9 que le problème de la planification et de la commande de systèmes à événements discrets est très souvent décomposée en niveaux hiérarchiques pour pouvoir réduire la complexité. Les travaux décrits dans ce chapitre ont trait au niveau coordination globale et pilotage, du point de vue de la *commande*. Nous nous sommes intéressés aussi à la coordination globale du point de vue de la *surveillance* des Ateliers de Fabrication Automatisés. Dans ces systèmes, une surveillance efficace doit être capable de tolérer les interventions humaines et nécessite de pouvoir décrire des situations où l'état du système n'est pas connu avec certitude. Or, au delà d'une décomposition en niveaux, une autre façon de simplifier l'analyse d'un système complexe est de permettre un certain degré d'incertitude dans sa description [zKli88, zDub89]. Cela peut être formellement pris en compte en utilisant la théorie des Possibilités, avec les concepts de distribution de possibilités et de mesure de possibilités [zDub88]. Il faut pouvoir se ramener, du point de vue de la surveillance, à une description globale d'un ensemble d'alternatives sans effectuer un choix entre elles. L'idée est celle de raisonner sur cet ensemble d'alternatives tant que de nouvelles informations plus précises n'arrivent pas. Il faut pouvoir aider au diagnostic, avec d'un coté les informations disponibles et de l'autre, la structure du système. Nous avons choisi de réaliser cette combinaison en utilisant la théorie des possibilités – qui permet de raisonner sur des informations mal-connues (imprécises, floues ou incertaines) – et les réseaux de Petri – qui permettent de représenter la dynamique d'un système, en particulier les contraintes de séquencement découlant, par exemple, des gammes de fabrication et des allocations des machines. L'intérêt d'utiliser conjointement ces deux théories dans le domaine des systèmes manufacturiers a été discutée d'abord dans [Val94] et ensuite de façon plus approfondie dans [Car96] pour les systèmes à événements discrets en général. Notre approche alliant les réseaux de Petri à objets et la théorie des Possibilités est décrite en [Car98, Car99]. Dans le cas de réseaux de Petri saufs, on peut travailler avec la notion d'information mal-connue sans avoir besoin d'utiliser les réseaux de Petri à objets. En [San93, San97, San98, San99] nous montrons comment utiliser les informations du graphe de marquages d'un réseau possibiliste temporel sauf pour établir des préférences entre les états possibles d'un système.

Une partie des travaux dans ce domaine s'est déroulée dans le cadre du projet brésilien *Un environnement pour la spécification formelle de bases de connaissances représentant les systèmes dy-*

namiques, dans lequel s'est développé le travail de *Mestrado* de L. Caimi [Cai98] (en collaboration avec G. Bittencourt) [Bit98, Cai98a, Cai98b]. Ces travaux sont détaillés section 10.2.

Nous avons déjà dit qu'une autre façon de simplifier l'analyse d'un système complexe est de permettre un certain degré d'incertitude dans sa description. Cette incertitude peut venir non pas d'un mauvais fonctionnement, mais de la volonté de repousser des choix à plus tard lors des phases amont de la conception, par exemple en choisissant un domaine de valeurs possibles pour certains paramètres ou contraintes, ou bien de fixer une valeur arbitraire. Si l'on prend en compte, en particulier, les contraintes temporelles, quand on s'intéresse à la vérification, on peut utiliser la notion de graphe de classes ou celle de scénario. Dans ce dernier cas, je me suis intéressée à construire des raisonnements au-dessus des modèles du système physique, en utilisant la logique linéaire [Car93, Car95a, Car96a, Car95b, Kun96b, Val96, Car97]. Nous détaillerons cette approche section 10.3, et l'approche graphe de classes au chapitre 11.

Pour mieux maîtriser cette théorie, indépendamment de la notion de séquence, j'ai effectué quelques travaux sur les applications des ensembles flous lors de la réalisation d'un contrôleur flou pour un hélicoptère radio-commandé dans le cadre d'une collaboration industrielle avec l'entreprise Gyron Technologie à Florianópolis, Brésil. Cet hélicoptère avait un rotor de 2m et devrait être utilisé pour l'inspection de lignes électriques. Le contrôle flou utilise des règles du type *si X est A alors Y est B*, où A et B sont des ensembles qui peuvent être flous. Son principe est assez simple et utilise seulement un fragment de la théorie des ensembles flous. Le contrôle flou peut être vu comme un type de raisonnement par analogie ou comme un méchanisme d'interpolation. Il peut être très utile dans les systèmes complexes où il n'y a pas de modèle mathématique précis, comme c'est le cas des hélicoptères. Le *Mestrado* de C. Cavalcante [Cav94a] s'est déroulé au sein de cette collaboration et a porté sur la conception et la simulation du système de navigation de cet hélicoptère [Cav94b, Cav95].

10.2 Les réseaux de Petri et l'imperfection de l'information

Il y a deux aspects fondamentaux dans les systèmes dynamiques à événements discrets : l'évolution dans le temps et l'incertitude des informations traitées. Ces deux aspects ont été pendant un moment traités dans la littérature de façon disjointe : les travaux sur l'aspect évolutif rarement prenaient en compte le caractère incertain de la description du monde, et l'aspect dynamique n'a pas toujours été considéré dans les travaux traitant l'aspect incertain des informations. Le premier papier prenant en compte les réseaux de Petri et la logique floue est celui de Looney [zLoo88]. Plusieurs approches ont été proposées pour les réseaux de Petri flous, mais sous ce même nom, elles sont basées sur des notions différentes, et plus ou moins cohérentes avec la théorie des réseaux de Petri. Par exemple, dans [zLoo88], les RPF s'approchent des graphes et/ou ou des réseaux de neurones sans les aspects dynamiques des réseaux de Petri ; dans [zSca93], les RPF sont des modèles de raisonnements sur des propositions ; dans [Car90b] ils sont des modèles de raisonnements sur le temps et les états. Deux voies principales se dégagent dans tous ces travaux : soit les RPF montrent l'enchaînement d'un ensemble de règles et un marquage correspond à une étape dans le raisonnement, soit ils représentent un système dynamique et le marquage dénote la connaissance sur son état à un instant donné.

Mes travaux concernant l'association des réseaux de Petri avec la théorie des possiblités ont débuté

à l'occasion de mon doctorat au LAAS [Car90a]. Je les ai poursuivi lors de mon retour au Brésil (LCMI Florianópolis) puis lors de mon retour en France et jusqu'à aujourd'hui.

J'ai proposé deux approches pour combiner les réseaux de Petri et la théorie des possibilités selon que la façon de prendre en compte le temps est implicite ou explicite.

Dans le modèle du réseau de Petri avec marquage flou [Car98], [Car99] la prise en compte du temps est implicite. Le marquage d'un réseau de Petri à marquages flous représente une connaissance imprécise (ou floue) de l'état d'un système. Un tel marquage ne permet pas d'affirmer qu'un jeton est dans une place donnée, mais seulement qu'il est dans l'une des places d'un ensemble des places donné. À chaque jeton est associé une *distribution de possibilité* sur l'ensemble de places. La définition de ce modèle a été possible grâce à l'utilisation des réseaux de Petri à Objets comme modèle de base [zSib85]. Dans ce modèle, un jeton est un n-uplet individualisé d'instances de classes. C'est donc impossible d'avoir deux jetons identiques dans un marquage *connu* (précis) ; l'existence de deux jetons identiques dans un marquage le caractérise comme imprécis ou flou (selon les valeurs des la fonction de distribution de possibilité pour ce marquage). Le temps est pris en compte sous la forme d'une interprétation en associant à chaque transition une fonction d'autorisation. La notion de pseudo-tir est introduite, permettant de réaliser des inférences sur l'état du système quand la valeur de cette fonction n'est pas certaine. Le pseudo-tir construit un marquage flou comme un *nuage* : à partir d'un marquage certain, autres marquages sont additionnés. Ce nuage augmente jusqu'à la réception d'une information certaine. Dans ce cas, la taille du nuage diminue ou se réduit à un seul marquage (marquage précis).

Dans le modèle du réseau de Petri temporel flou [Car98] la prise en compte du temps est explicite. Les modèles de base sont le réseau de Petri à objets et le réseau de Petri temporel [zMer74] où à chaque transition est associée une *distribution de possibilité* sous la forme d'un intervalle de temps flou au lieu d'un intervalle (imprécis). L'intérêt du formalisme du réseau de Petri se trouve dans sa représentation d'un ordre partiel entre les événements et donc la possibilité d'élaborer un raisonnement complexe sur les informations temporelles. La propagation des contraintes temporelles et le calcul du marquage sont basés sur les méthodes de raisonnement concernant la connaissance temporelle floue [zDub89].

En collaboration avec S. Sandri, du INPE (Instituto National de Pesquisas Espaciais, São Paulo)[1] nous avons proposé un modèle plus simple, les réseaux de Petri possibilistes saufs temporisés, où une durée floue est associée à chaque transition, et chaque place place peut contenir un seul jeton. Nous avons ajouté un degré de possibilité à chaque transition pour exprimer la préférence d'une transition en relation aux autres [San97, San98, San99]. Le marquage initial est connu, mais à un instant θ donné (précis ou imprécis) il y a une information incomplète sur l'état du système, permettant de connaître seulement le marquage de certaines places et donc d'avoir une représentation partielle du système. À partir du graphe des marquages du réseau sous-jacent (sans le temps) un arbre avec les informations temporelles est généré jusqu'à l'instant θ. Cette information temporelle, associée aux préférences entre les transitions, permet d'ordonner l'ensemble des marquages candidats (compatible avec l'état partiel) à représenter l'état du système et donc d'établir des préférences entre les états possibles. Cette approche est différente de celle présenté en [Car99], d'abord parce que les réseaux de Petri ici sont ordinaires et saufs. Ensuite, si la modélisation du système est faite en utilisant le réseau de Petri, l'étape de supervision est faite en utilisant le graphe de marquage, contrairement à [Car99] où il y a une propagation en avant avec la définition du pseudo-tir.

[1] Actuellement au IIIA – Instituto de Investigation en Inteligencia Artificial à Barcelone, Espagne, rataché au CSIC – Consejo Superior de Investigaciones Cientificas.

À Florianópolis un certain nombre de mes collègues travaillaient dans le domaine de l'I.A. et comme je l'ai exposé au paragraphe 9.2, j'avais dès mon retour encadré des étudiants qui avaient traduit des réseaux de Petri ordinaires sous la forme de règles pour la supervision de systèmes industriels. Lorsque la technique de modélisation fondée sur les réseaux de Petri avec marquage flou m'a paru mature, je me suis posé le problème de sa mise en oeuvre. C'est tout naturellement que je me suis à nouveau orientée vers l'I.A. Cette mise en oeuvre a donné lieu à un travail de *Mestrado*, celui de L. Caimi [Cai98]. Ces travaux se sont déroulés dans le cadre du projet brésilien[2] *Un environnement pour la spécification formelle de bases de connaissances représentant les systèmes dynamiques*. Les liens entre les réseaux de Petri et les systèmes de règles ont rendu possible la collaboration étroite avec G. Bittencourt, un chercheur en IA de mon laboratoire.

L'outil de simulation construit par L. Caimi [Bit98, Cai98a, Cai98b] est formé de trois modules indépendants :
- un traducteur du langage de description des réseaux de Petri à marquage flou en faits et règles associés à la base de connaissance du système expert ;
- un simulateur de réseau de Petri qui exécute le système expert généré par le traducteur ;
- un module de gestion avec d'autres programmes qui organisent et explorent les informations déclaratives obtenues par la simulation du réseau de Petri.

Cet outil est basé sur FASE [zBit93], un *shell* de systèmes experts en Common Lisp qui contient trois langages de représentation de la connaissance (logique, frames et réseaux sémantiques), les stratégies chaînage arrière et chaînage avant et différentes méthodes de résolution de conflits. Nous avons utilisé les frames pour représenter les réseaux de Petri à marquage flou. Ce formalisme, crée par Minsky, est formé d'une hiérarchie de structures de données appelées *frames*. Chaque frame est composé d'un ensemble de slots avec des valeurs associées. Ces valeurs peuvent être des primitives d'un concept représenté par le frame, ou un autre frame. Ce formalisme possède de mécanisme d'inférence efficace : héritage, *facet* qui contrôle le type de valeur adéquate pour chaque slot, valeurs par défaut et possibilité d'associer des procédures à ses attributs pour permettre l'exécution de fonctions externes, appelées *démons*.

La représentation des réseaux de Petri par frames est basée sur les places. Chaque place est un frame avec plusieurs slots. Les slots *identification*, *transitions d'entrée* et *transitions de sortie* contiennent des symboles qui identifient la places et ses transitions d'entrée et de sortie. Le démon attaché au slot *interface* permet de représenter le comportement du réseau. Chaque place a un slot buffer de sortie associés à chacune de ses sorties, $contenu_i$, qui contien un seul objet à chaque moment. Chaque place a un seul slot buffer d'entrée (*contenu*) qui contient la liste des objets. Ces slots d'entrée et de sortie ne sont pas modifiés directement, mais sont traités par le démon du slot *interface*. Le principal problème a été de d'implémenter le mécanisme de *pseudo-tir*. Il nous a fallu aussi traiter le problème du contenu de la place : comment mettre à jour le marquage quand, dans un même cycle, il y a des règles qui rajoutent des objets et d'autres qui enlèvent des objets. La base de règles a été partagé donc en trois, de façon à traiter : le choix de la règle à exécuter, le tir normal et le pseudo-tir. Le marquage certain est représenté par un objet *réel* et le marquage flou par un objet *ghost*, qui hérite de tous les attributs des objets réels par le slot localisation. Ce slot va contenir un ensemble de noms de place dans le cas du marquage flou. Seulement le tir normal est affecté par les règles de conflit ; si plusieurs règles de pseudo-tir sont habilitées elles sont toutes franchies. Au début de chaque cycle, après avoir reçu une information externe, l'algorithme de recouvrement est exécuté s'il y a des objets ghost. Cet algorithme utilise la fonctionnalité de requêtes de la représentation

[2]Les projets CNPQ/AI sont proposés par un chercheur et ont une durée des 2 ans, avec une évaluation à mi-parcours (AI veut dire "auxílio individual" (*aide individuelle*) ; le nombre de bourses est réduit et la concurrence est très grande.

par frames. Il doit annuler ou finir les pseudo-tir à partir de l'information reçue. Si le marquage résultant est précis, l'objet réel est mis à jour et les ghosts sont effacés.

Le modèle de réseaux de Petri avec marquage imprécis permet de spécifier et d'implémenter la fonction surveillance au niveau de la coordination d'un système, et peut être validé en utilisant le simulateur implémenté par L. Caimi. Du point de vue de l'analyse, un marquage imprécis est simplement un sous-ensemble connexe de l'ensemble des marquages accessibles. Si l'algorithme de restauration permet de retrouver un état certain M, celui-ci doit être un élément appartenant l'ensemble des marquages accessibles.

10.3 Caractérisation d'une séquence de transitions par la logique linéaire

Les travaux que j'avais menés concernant la mise en oeuvre des réseaux de Petri dans des environnements I.A., que ce soit directement sous la forme de règles (en assimilant une transition à une règle cf section 9.2) ou sous une forme plus sophistiquée exploitant la structure graphique du réseau de Petri pour optimiser la recherche des règles applicables (cf section 10.2), m'avaient convaincu de l'existence de liens forts (bien que cachés) entre I.A. et réseaux de Petri. La spécificité des réseaux de Petri vus comme des ensembles de règles est cependant forte : la condition d'application de la règle (les jetons sensibilisant une transition) devient systématiquement fausse (ces jetons sont consommés lors du franchissement) lors de l'application de la règle. Si ce n'est pas interdit dans un système de règle, ce n'est par contre pas obligatoire [Car97]. Cela est même contradictoire avec le mécanisme de déduction de la logique classique qui impose qu'une proposition, une fois prouvée *vraie*, ne peut devenir fausse.

J'ai donc été amenée à me poser la question suivante : peut-on réconcilier les réseaux de Petri et la logique et cela permet-il d'avoir une autre vision de l'association que j'avais faite entre les réseaux de Petri et la logique possibiliste sous la forme de la définition des réseaux à marquage flou (cf section 10.2).

B. Pradin-Chézalviel [zChe00] et R. Valette (LAAS) se sont intéressés à la traduction des réseaux de Petri en logique linéaire dans le but de développer des méthodes d'analyse des réseaux de Petri sans construire le graphe des états accessibles. Cette logique est appelée "logique des ressources" de par ses capacités à raisonner sur les notions d'exemplaires, de ressources que l'on peut produire et consommer. Certaines règles de la logique classique ne sont plus valides (contraction et weakening) dans la logique linéaire. La négation exprime la dualité consommation-production et non pas les valeurs vrai-faux. La traduction des réseaux de Petri en logique linéaire est donc *naturelle* ou du moins directe puisque l'exécution d'une action aussi bien dans un réseau de Petri que dans un système de règles linéaires correspond à une consommation de ressources.

L'analyse est faite sur un *scénario* du réseau de Petri, défini comme une séquence de transitions écrite sur la forme d'un ordre partiel, en utilisant la structure de l'arbre de preuve de la logique linéaire. Ceci permet une caractérisation claire des scénarios de franchissements, ce que ne fait pas la théorie des réseaux de Petri (le vecteur caractéristique d'une séquence ne prend pas en compte l'ordre des transitions et les séquences imposant une sémantique d'entrelacement). Était-t-il possible de représenter aussi les marquages imprécis et les séquences de tir imprécises d'un réseau de Petri à Objets comme définis dans [Car90a, Car90b] ? En particulier, étant donné l'exécution

d'un scénario, quels sont les interactions entre la connaissance incomplète de son comportement (séquences imprécises) et la connaissance incomplète de son état actuel ?

Nous avons montré en [Car93, Car95a] que les connecteurs additifs $\oplus$ (plus) et & (avec) de la logique linéaire permettent de construire des ensembles disjonctifs de séquences de tir ou de marquages[3]. Ces connecteurs représentent, respectivement, des choix *interne* et *externe*. Dans le premier cas le choix de la séquence à être franchie est interne au système, indépendamment de son environnement externe tandis que dans le deuxième cas ce choix est libre et peut être fait par l'environnement. Un marquage imprécis n'étant qu'un ensemble disjonctif de marquages, les connecteurs additifs permettent bien d'introduire l'imprécision. Nous avons utilisé le choix *externe* pour représenter l'imprécision telle que définie en [Car90a, Car90b] aussi bien pour le marquage imprécis que pour les séquences de tir imprécises.

Un marquage précis M indique une connaissance du monde réel précise ou *bien connue*. Lors de la réception d'un événement arrivant du système, le tir d'une transition à partir d'un tel marquage correspond à une *mise-à-jour* de l'état du système. Cependant, si l'on considère que le système a pu changer sans que l'événement soit reçu (dû à une défaillance), la connaissance sur le système – représentée par le marquage – doit être *révisée*. Cela est réalisé par le *pseudo-tir* de la transition correspondante, qui amène à un marquage imprécis $\mathcal{M}$.

Ce marquage $\mathcal{M}$ atteint par le pseudo-tir de t à partir de M, est représenté par la formule $\mathcal{M} : M\&M'$ où $M \xrightarrow{t} M'$ et est vu comme un choix *externe* (connecteur additif &) fait par l'environnement (la défaillance). Soit M_i l'état qui serait atteint par le franchissement de la séquence s_i (ayant au moins une transition t qui admet un pseudo-tir) à partir du marquage initial M_0, $M_0 \xrightarrow{s_i} M_i$. La propagation de l'imprécision peut être telle que si plusieurs transitions sont pseudo-tirées, la représentation de l'état du système correspond à un ensemble disjonctif de marquages, $\mathcal{M} : M_0\&M_1\& \ldots \&M_k$, comme on peut voir sur la figure **??**. La réception d'information certaine, associé au tir d'une transitions t qui a été pseudo-tirée, permet de faire une révision concernant cet événement. L'imprécision peut être alors partiellement éliminée, ou totalement éliminée, au quel cas le marquage redevient certain. Dans le cadre de la logique linéaire, cela correspond à l'utilisation du connecteur additif & dans les séquents entre $\mathcal{M}$ et chaque formule $M_0 \overset{s_i}{\multimap} M_i$, avec $t \in s_i$. Un exemple concernant le mécanisme de propagation et de révision en logique linéaire est développé en [Car96a].

La propagation de l'imprécision peut être telle que le système atteint un état correspondant au tir de plusieurs séquences de tir disjointes. Le marquage atteint par un ensemble disjonctif de k séquences est aussi un ensemble disjonctif de marquages : $\mathcal{M} : M_0\&M_1\& \ldots \&M_k$.

Un aspect important dans la théorie des réseaux de Petri est la représentation du vraie parallélisme. Or, lors de la vérification par le graphe de marquages, est utilisé la notion de parallélisme par entrelacement et la structure du réseau de Petri est perdue. En plus, si l'on s'intéresse au raisonnement temporel, la vérification des propriétés est faite utilisant la logique temporelle sur ce graphe de marquages. Par contre, la logique linéaire permet de construire des raisonnements basés sur la structure du réseau, indépendamment de son marquage initial, en utilisant le raisonnement sur des scénarios [Val96]. Elle permet donc d'avoir un point de vue orienté événements alors que la logique temporelle présente un point de vue orienté vers les états. Cela reste vraie aussi pour le réseau de Petri temporel flou, qui permet de décrire à la fois le comportement normal et le comportement en

[3]Une partie de ces travaux a été réalisée lors de mon année sabatique à Toulouse (94/95), quand j'ai eu l'occasion de participer au *Programme de Recherches Coordonnées (PRC-GDR) du Centre National de la Recherche Scientifique - CNRS Gestion de l'Evolutif et de l'Incertain dans une Base de Connaissance.*

cas de panne. Une application de ces travaux a été faite dans la supervision pour améliorer le suivi et amorcer le diagnostique [Car95b, Kun96b].

Ces travaux illustrent bien l'intersection entre les différents champs disciplinaires comme évoqué dans la figure 7.1.b au chapitre 7. D'une part il y a l'utilisation conjointe des objets (modélisés par des diagrammes de classe dans la communauté UML) et des réseaux de Petri. D'autre part, nous avons utlisé des techniques I.A. pour augmenter l'efficacité des systèmes de règles dans les travaux de L. Caimi. Les travaux sur la logique linéaire ont permis une meilleure compréhension des relations entre logique et réseaux de Petri ainsi qu'une compréhension des liens entre ces relations et les sémantiques d'entrelacement et de parallélisme vrai. En effet, la sémantique de parallélisme vrai nécessite de restreindre les relations d'ordre entre les franchissements aux seuls relations de causalité (produire un jeton avant de pouvoir le consommer). Ces travaux m'ont permis d'être mieux armée pour aborder l'analyse des réseaux de Petri avec des contraintes temporelles au chapitre 11.

Chapitre 11

Spécification et vérification de contraintes temporelles quantitatives imprécises et floues

11.1 Introduction

Le caractère graphique de réseaux de Petri est la première avantage qui voient les utilisateurs néophytes de ce modèle, notamment ceux d'IA qui trouvent commode d'utiliser un modèle qui leur permet de structurer la connaissance. Le fait d'être un outil d'analyse est vu souvent comme un bonus supplémentaire. Ce n'est pas mon cas car j'ai toujours considéré que la preuve formelle ou la vérification formelle, même si elle n'était qu'un élément d'une démarche plus générale de validation plus ou moins formelle, était nécessaire pour les systèmes embarqués.

Cette préoccupation était d'ailleurs au premier plan des travaux sur la sémantique des diagrammes UML présentés au chapitre 8. Elle était également sous-jacente aux travaux présentés en section 9.2 puisque la motivation principale pour spécifier la supervision des stations de pompage était la nécessité de pouvoir la vérifier avant la mise en oeuvre. Ce n'est que dans le cadre des travaux sur la conception de systèmes tuteurs intelligents que la possibilité d'analyse formelle se restreignait à la vérification des réseaux de Petri sous-jacents et le pouvoir descriptif de comportements et de scénarios est passé au premier plan.

Tant que les contraintes temporelles quantitatives n'interviennent pas, et si le réseau de Petri est borné, il est toujours possible de vérifier les propriétés en énumérant tous les marquages accessibles. La tâche devient beaucoup plus complexe lorsque des contraintes temporelles quantitatives sont introduites car alors le nombre d'états (marquage + horloge) devient infini. Il faut trouver le moyen de regrouper ces états en un nombre fini de classes pour pouvoir vérifier les propriétés.

Comme nous l'avons vu au chapitre 10, j'ai introduit dès ma thèse de doctorat un outil de modélisation, les réseaux de Petri temporels flous, rajoutant l'aspect flou aux réseaux de Petri temporels. Le problème de leur analyse formelle s'est posé dès le départ. À l'époque l'analyse formelle des réseaux temporels était en cours de développement et il fallait sans doute que j'éclaircisse les relations entre logique et réseaux de Petri avant de me lancer dans l'étude de l'impact du flou sur cette analyse.

Dans le cas du réseau de Petri temporel flou, il est possible d'effectuer une première analyse à partir

du réseau sous-jacent, sans considérer les intervalles temporels. Une deuxième analyse peut être faite en considérant seulement le support des intervalles temporels flous, ce qui revient à prendre le réseau de Petri temporel le moins contraignant. Il est possible alors de construire le graphe de classe comme proposé, par exemple, par Bertomieu et all [zBer04]. Cependant, toute la richesse d'information exprimée par les l'intervalles flous est perdue du point de vue de la vérification des propriétés. Il était donc naturel d'étendre le graphe de classes d'états d'un réseau de Petri temporel pour le réseau de Petri temporel flou. Dans le DEA de S. Cousy [Cou04, Car05a] nous avons défini une méthodologie de calcul du *graphe de classes d'états flou* qui étend le graphe des classes classique et qui permet de vérifier les propriétés LTL (Linear Time Logic) comme défini dans [zBer04]. Cependant, il n'est pas toujours possible d'associer à un chemin dans le graphe une séquence effectivement franchissable dans le systèmes. Dans le cas où cela est possible, les contraintes que doivent vérifier les dates de franchissement des transitions ne sont pas directement données. Avec R. Valette, nous nous sommes posé la question de construire un graphe de classes tel qu'à tout chemin de ce graphe il soit possible d'associer un ensemble de contraintes temporelles délimitant exactement les domaines des dates des franchissements de transition de la séquence correspondante dans le réseau de Petri correspondant. Cette information est effectivement nécessaire pour connaître le degré de possibilité de franchissement de la transition (il faut le min de la possibilité de l'état de départ avec celui de l'intervalle *possible* (réel) de tir à partir de cet état). La seule information donnée pour un intervalle de tir potentiel ne suffit pas.

Avant de s'attaquer aux réseaux de Petri temporels flous, nous avons d'abord décidé de traiter le cas imprécis en définissant un nouveau graphe de classes pour les réseaux de Petri t-temporels. Ces travaux concernent le M2R de X. Mao [Mao05, Car05b, Car05c, Mao05a] et A. Hamdani [Ham06]. Nous avons ensuite étendu ce graphe de classes aux réseaux de Petri temporels flous [Car06b].

11.2 Graphe de classes flou préservant les propriétés LTL

Une classe d'états d'un réseau de Petri temporel est un ensemble d'états qui ont le même marquage et dont le domaine de tir est l'union du domaine de tir de ses états. Ce réseau utilise la sémantique forte (si plusieurs transitions sont sensibilisées, une d'entre elles doit être franchie avant la limite supérieure des autres). Le parallélisme est traité par l'entrelacement (les transitions sont franchies séquentiellement, définissant un ordre total).

Pour rappel, un réseau de Petri temporel RdPT est un triplet $< N, M_0, I >$ où : $N =< P, T, Pre, Post >$ est un Réseau de Petri, M_0 : est le marquage initial, $I^0 : T \rightarrow (Q^+ \cup 0) * (Q^+ \cup 0)$ est la fonction intervalle statique.

La construction du graphe de classes d'un réseau de Petri temporel est basée sur deux notions : l'*intervalle de tir* d'une transition, et le *domaine temporel* des transitions qui restent sensibilisées après le tir d'une transition. Une classe est caractérisée par un marquage et un domaine temporel ; l'arc reliant deux classes C et C' du graphe est étiqueté par l'*intervalle de tir* $D(t_i)$ de la transition t_i qui provoque le changement d'état. Le *domaine temporel* de cette nouvelle classe d'état C' est donné par :

- l'*intervalle de tir* $I'(t_k)$ de toutes les transitions t_k sensibilisées par le marquage de C',
- les *contraintes de temps* $[t_k, t_i]$ entre tous les couples de transitions (t_k, t_i) qui restent sensibilisées. Pour que t_i puisse être franchie avant t_k, cette contrainte doit être un intervalle temporel positif. Elle doit prendre en compte aussi les contraintes dans les classes précédentes (mémoire du passé).

Nous avons exprimé le calcul du domaine temporel d'une classe ainsi que le calcul de l'intervalle de tir en utilisant la théorie des intervalles [Car05a], au lieu d'utiliser le graphe de régions comme [zBer04].

Soit $I_0(t_i) = [a_i, b_i]$ l'intervalle temporel statique associé à une transition t_i, $I^j(t_i)$ son intervalle temporel dynamique dans la classe C_j et soit

$$[t_k, t_i]_0 = (I^0(t_k) - I^0(t_i)) \cap [0, \infty) = [a_i - b_i, b_k - a_i] \cap [0, \infty) \tag{11.1}$$

la contrainte temporel dans la classe initial C_0. On considère qu'il n'y a pas de contrainte avant le temps 0, $[t_k, t_i]_{-\infty} \subseteq (-\infty, \infty)$. La *contrainte de temps* entre deux transitions t_i et t_k dans une classe C_j, $j \geq 0$ est donnée par

$$[t_k, t_i]_j = (I^j(t_k) - I^j(t_i)) \cap [t_k, t_i]_{j-1} \cap [0, \infty) \tag{11.2}$$

Si $[t_k, t_i]_j \neq \phi$ alors t_i peut être franchie avant t_k. Si $[t_k, t_i]_{j-1} \subseteq I^j(t_k) - I^j(t_i)$, alors la restriction dans le passé est redondante.

Comme on utilise la sémantique forte, une transition t_i peut être franchie à partir de C_j pendant son *intervalle de tir*, mais avant la plus petite borne supérieur de toutes les transitions sensibilisées dans C_j :

$$D^j(t_i) = [a_i, b_i] \bigcap_k [0, b_k] \tag{11.3}$$

Si l'on note $sUB = min_k(b_k)$ la plus petite borne supérieur de toutes les transitions sensibilisées dans la classe C_j, l'équation 11.3 correspond à $D^j(t_i) = [a_i, sUB]$.

Après le tir d'une transition t_i à partir d'une classe C_j, l'*intervalle de temps* des autres transitions t_k doit être mis à jour dans la classe C_{j+1} :

$$I^{j+1}(t_k) = [t_k, t_i]_j \cap (I^j(t_k) - D^j(t_i)) \cap [0, \infty) \tag{11.4}$$

Soit par exemple le cas d'un réseau de Petri temporel avec trois transitions parallèles t_1, t_2 et t_3 sensibilisées dans la classe initial C_0, avec intervalle statique $I_0(t_1) = [5, 6]$, $I_0(t_2) = [7, 8]$ et $I_0(t_3) = [0, 4]$. En appliquant les équations 11.1, 11.3 et 11.4 on calcule la nouvelle classe C_1, puis en appliquant les équations 11.2, 11.3 et 11.4, la classe C_2. Le graphe de classes d'état résultant est représenté figure 11.1.

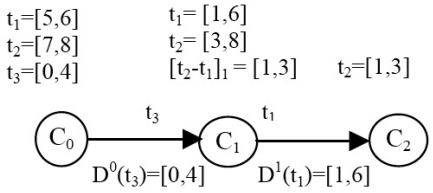


FIG. 11.1 – Graphe de classes d'un réseau temporel.

Les opérations de base de ce calcul sont la différence d'intervalles et l'intersection d'intervalles. Cela nous a permis d'étendre ce calcul d'intervalle (imprécis) au calcul d'intervalle flou, en utilisant la différence floue $\ominus$ et l'intersection floue [zDub88].

Pour rappel, un réseau de Petri temporel flou RdPTF [Car98] est un triplet $< N, M_0, I_f >$ où : $N =< P, T, Pre, Post >$ est un Réseau de Petri, M_0 est le marquage initial, $I_f^0 : T \rightarrow (Q^+ \cup 0) *$

$(Q^+ \cup \infty) * (Q^+ \cup \infty) * (Q^+ \cup \infty)$ est la fonction intervalle flou statique. La fonction I_f^0 associe à chaque transition t_i un intervalle temporel flou $[a_i, b_i, c_i, d_i]$ qui représente l'ensemble de toutes les dates de tir possibles à partir de sa date de sensibilisation. Si $a_i = b_i$ et $c_i = d_i$ pour toutes les transitions $t_i \in T$, alors le RdPTF est équivalent à un RdPT ayant $I^0 = [a_i, c_i]$.

Les équations 11.1 à 11.4 pour le calcul du graphe de classes d'un RdPT sont étendues au cas flou par, respectivement, les équations :

$$[t_k \ominus t_i]_0 = (I_f^0(t_k) \ominus I_f^0(t_i)) \cap [0, \infty) = [a_k - d_i, b_k - c_i, c_k - b_i, d_k - a_i] \cap [0, \infty) \tag{11.5}$$

$$[t_k \ominus t_i]_j = (I_f^j(t_k) \ominus I_f^j(t_i)) \cap [t_k, t_i]_{j-1} \cap [0, \infty) \tag{11.6}$$

$$D_f^j(t_i) = I_f^j(t_i) \bigcap_k [0, 0, c_k, d_k] = h * [a, b, c, d] \tag{11.7}$$

$$I_f^{j+1}(t_k) = [t_k, t_i]_j \cap (I_f^j(t_k) - D^j(t_i)) \cap [0, \infty) \tag{11.8}$$

Si $h = 1$ dans l'équation 11.7, l'ensemble flou $D_f^j(t_i)$ est normalisé ; sinon, il est dit sous-normalisé.

I_f est l'ensemble flou qui délimite la distribution de possibilité de la date de tir de t_i, $\pi_{I_f(t_i)} : X \to [0, 1]$, où $X = [0, \infty)$ c'est l'ensemble des instants de temps. Nous allons utiliser la notation $\pi_{t_i} : T \to [X \to [0, 1]]$, pour $t_i \in T$.

Dans le cas du réseau de Petri temporel flou, la construction du graphe de classes consiste à savoir si une transition t_i, avec une distribution de possibilité π_{t_i}, peut être franchie avant les autres transitions t_k avec π_{t_k}, sensibilisées (par le marquage) à partir d'une classe C_j. L'association d'une distribution de possibilité aux transitions permet de calculer les mesures de possibilité et de nécessité du fait que le tir de t_i ait lieu avant les autres transitions t_k.

À partir de l'équation 11.7 on peut calculer la mesure de possibilité $\Pi^j(t_i \leq t_k)$ que t_i soit franchie avant les autres transitions t_k à partir de la classe C_j :

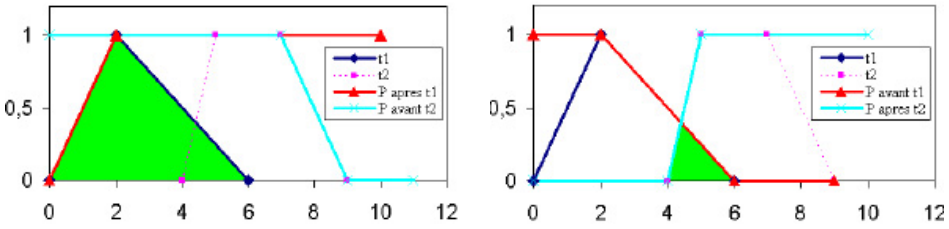
$$\Pi^j(t_i \leq t_k) = sup\ D_f^j(t_i) = h \tag{11.9}$$

L'équation 11.9 correspond à la formule $\Pi^j(t_i \leq t_k) = min_{k \neq i} sup_{x \leq y}(min(\pi_{t_i}(x), \pi_{t_k}(y))$ [zDub89]. Si $D_f^j(t_i)$ est normalisé ($h = 1$) la mesure de nécessité $N^j(t_i \leq t_k)$ peut être calculée ; sinon $N^j = 0$. La mesure de nécessité est donnée par $N^j(t_i \leq t_k) = 1 - \overline{\Pi^j(t_i < t_k)} = 1 - \Pi^j(t_k \leq t_i)$. Pour simplifier la notation, quand il y a plusieurs transitions t_k on note directement $\Pi^j(t_i)$ et $N^j(t_i)$.

La figure 11.2 indique l'intervalle de tir (en gris) et les valeurs de mesure de possibilité dans le cas d'un réseau de Petri temporel flou ayant deux transitions parallèles t_1 et t_2 avec $I_f^0(t_1) = [0, 2, 2, 6]$ et $I_f^0(t_2) = [4, 5, 7, 9]$. En appliquant l'équation 11.7 on obtient les intervalles $D_f^0(t_1) = [0, 2, 2, 6]$ et $D_f^0(t_2) = 0.4 * [4, 4.4, 4.4, 6]$ et en appliquant l'équation 11.9 on obtient $\Pi^0(t_1 \leq t_2) = 1$ et $\Pi^0(t_2 \leq t_1) = 0.4$. Il est donc possible de franchir les deux transitions, mais la possibilité de franchir t_1 est plus grande que celle de franchir t_2. Ce classement est impossible dans le cas de réseaux de Petri temporels. Si l'on prend en compte seulement le support des distributions de possibilités associés aux transitions d'un RdPF, on obtient un RdPT avec $I^0(t_1) = [0, 6]$ et $I^0(t_2) = [4, 9]$, en

appliquant l'équation 11.3 on obtient $D^0(t_1) = [0, 6]$ et $D^0(t_2) = [4, 6]$: les deux transitions peuvent être également franchies.

La mesure de nécessité indique la certitude avec laquelle l'événement va avoir lieu. Pour cet exemple, $N^0(t_1 \leq t_2) = 1 - \Pi^0(t_2 < t_1) = 0.6$ et $N^0(t_2 \leq t_1) = 1 - \Pi^0(t_1 < t_2) = 0$.



a) $D_f^0(t_1)$ et $\Pi^0(t_1 \leq t_2) = 1$ b) $D_f^0(t_2)$ et $\Pi^0(t_2 \leq t_1) = 0.4$

FIG. 11.2 – Intervalle de tir et calcul de la mesure de possibilité.

Pour pouvoir faire le lien entre le graphe de classes d'un réseau de Petri temporel et celui d'un réseau de Petri temporel flou, nous avons défini en [Car05a] la notion de classe support-équivalente, ou *s-equiv*. Deux classes C et C' sont *s-equiv* si elles ont :
– le même marquage,
– le support de leurs intervalles temporels flous sont les mêmes (même si le noyau sont différents),
– pour chaque arc sortant de C, étiqueté par $D_f^j(t_i)$ et C', le support des intervalles de tir flous sont les mêmes

Si le graphe de classe du RdPTF est plié en respectant les classes *s-equiv*, le graphe obtenu correspond au graphe de classe du RdPT généré par l'outil Tina [zBer04]. Ce résultat n'est pas surprenant vu que les RdPT est un cas particulier d'un RdPTF quand les intervalles associés aux transitions sont imprécis au lieu d'être flous.

Considérons l'exemple traité en [Car05a] représenté figure 11.3 : la production est modélisée par les places p_1 et p_2, la consommation par t_5 et p_6 ; la transmission par t_2 et la stockage de données par le sous-réseau en pointillé. On souhaite connaître si un écrasement de données (tir de t_4) peut avoir lieu dans le système suite à l'arrivée d'un nouveau message avant que le précédent ne soit pas encore consommé. La structure du graphe de classes flou est donné figure 11.4.a ; la figure 11.4.b montre le graphe de classes pour le RdPT, dont les intervalles associés aux transitions t_i correspondent aux supports de $I_f^0(t_i)$. Si on applique la définition de classes *s-equiv*, on obtient les ensembles suivants : (C_2, C_{12}, C_{14}), $(C_3, C_9, C_{13}, C_{15})$, (C_4, C_{10}), (C_5, C_8), (C_6, C_{11}), (C_7, C_{16}, C_{17}). Les classes ce ces ensembles correspondent, respectivement, aux classes C_2, C_3, C_4, C_5 et C_6 du graphe de classes du RdPT de la figure 11.4.b. La table 11.1 montre le détail des classes et des arcs du graphe de classes flou de la figure 11.4.a. D'après le formules $max(\Pi(B), \Pi(\overline{B})) = 1$, $min(N(B), N(\overline{B})) = 0$ et $\Pi(B) = 1 - N(\overline{B})$ [zDub88], $\Pi < 1$ implique que $N = 0$ et $N > 0$ implique que $\Pi = 1$ c'est pourquoi ces valeurs impliquées ne sont pas montrées sur le tableau.

Dans le graphe de classes du RdPT, tous les tirs de transitions sont également possibles, tandis que dans le cas du graphe de classes flou (du RdPTF), les mesures de possibilité et de nécessité permettent d'ordonner les possibilités de tir. C'est le cas par exemple des transitions t_2 et t_5 sensibilisées dans la classe C_4 (figure 11.4.a), avec $\Pi^4(t_2) = 0.5/N^4(t_2) = 0$ pour $D_f^4(t_2) = 0.5 * [2, 2, 2, 3]$ et $\Pi^4(t_5) = 1/N^4(t_5) = 0.5$ pour $D_f^4(t_5) = 1 * [0, 0, 1, 3]$. Vu que $N^4(t_5) > N^4(t_2)$, le tir de t_5 est plus *nécessaire* que celui de t_2, compte tenu des intervalles de temps associés aux transitions pour cet exemple. Cependant, le tir de t_2 n'est pas impossible, et amène à la classe C_5

où l'écrasement de données (tir de t_4) ou la consommation (tir de t_5) peuvent également avoir lieu : $D_f^5(t_4) = D_f^5(t_5) = 1 * [0, 0, 0, 0]$, et donc $\Pi^4(t_5) = \Pi^4(t_4) = 1$.

Associer une sémantique dite floue est donc une manière d'exprimer une connaissance améliorée sur les instants de l'intervalle, ce qui permet de définir des mesures qui sont utiles pour un utilisateur. Dans le cas de cet exemple, il est possible de voir, aussi bien dans le graphe (flou) de la figure 11.4.a comme dans le graphe de la figure 11.4.b que c'est le tir de t_2 à partir de la classe C_4 qui rend possible le tir de t_4 (à partir de la classe C_5). Cependant, c'est que dans le graphe de classes flou qu'il est possible de mesurer que le tir de t_5 (consommation) est plus *nécessaire* que celui de t_2 (transmission) puisque $N_4(t_5) = 0.5$ et $N_4(t_2) = 0$. Si t_2 est franchie, alors l'écrasement et la consommation sont équi-possibles dans la classe C_5 : $\Pi_5(t_4) = \Pi_5(t_5) = 1$. Il est possible aussi d'affiner l'information sur les séquences : après un premier écrasement et une nouvelle production (classe C_{10} avec $D_f^{10}(t_5) = 1 * [0, 0, 0, 3]$ et $D_f^{10}(t_2) = 0.33 * [2, 2, 2, 3]$), la *nécessité* d'une consommation ($N^{10}(t_5) = 0.67$) est toujours plus grande que celle de la transmission ($N^{10}(t_2) = 0$). Par ailleurs, la nécessité de tir de t_5 (par rapport à t_2) est plus grande dans la classe C_{10} que dans la classe C_4 ; ces deux classes sont, comme nous avons vu, *s-equiv*. Il est vraie aussi que cette richesse d'information amène à une explosion combinatoire comme le montre la figure 11.4.

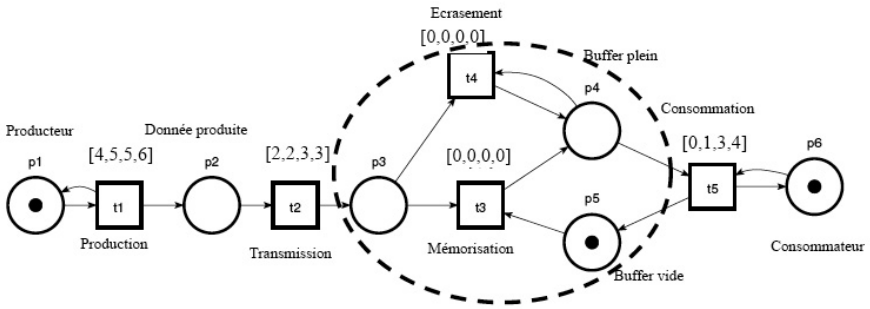


FIG. 11.3 – Réseau de Petri temporel flou du protocole de transfert de données uniderectionnel.

Les classes C_j avec le *domaine temporel* des transitions sensibilisées

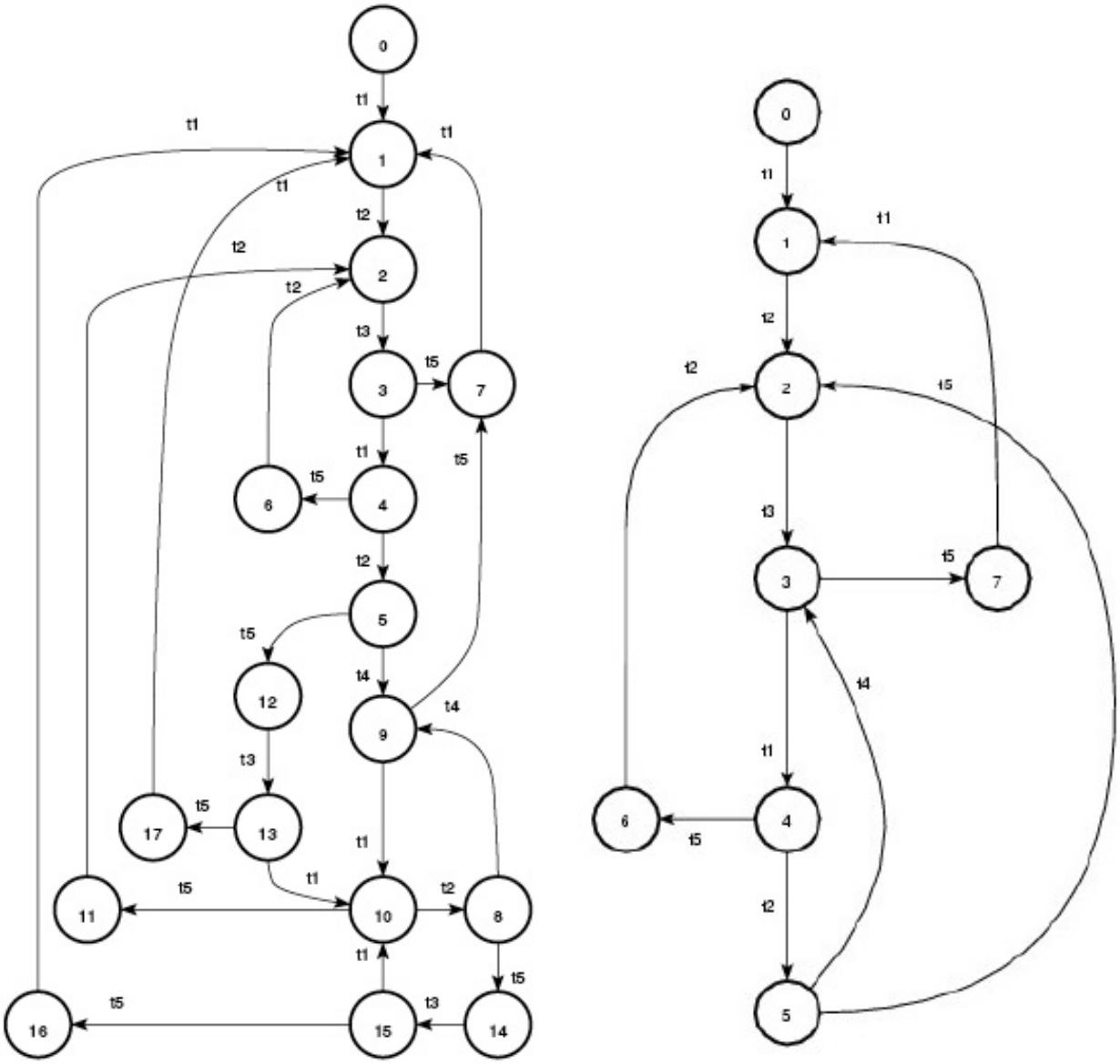
C0, p1 p5 p6, t_1=[4 5 5 6]
C1, p1 p2 p5 p6, t_1=[4 5 5 6], t_2=[2 2 3 3]
C2, p1 p3 p5 p6, t_1=[1 2 3 4], t_3=[0 0 0 0]
C3, p1 p4 p6, t_1=[1 2 3 4], t_3=[0 1 3 4]
C4, p1 p2 p4 p6,t_1=[4 5 5 6], t_2=[2 2 3 3], t_3=[0 0 1 3]
C5, p1 p3 p4 p6, t_1=[1 3 3 4], t_4=[0 0 0 0], t_3=[0 0 0 1] $[t_1 \ominus t_4]_5$ =[1 4 5 6]
C6, p1 p2 p5 p6, t_1=[1 4 5 6], t_2=[0 1 3 3] $[t_1 - t_2]_6$=[1 2 3 4]
C7, p1 p5 p6, t_1=[0 0 2 4]
C8, p1 p3 p4 p6, t_1=[1 3 3 4], t_4=[0 0 0 0],t_3=[0 0 0 1]
$[t_1 \ominus t_3]_6$=[4 5 5 6] – [0 0 0 3]=[1 5 5 6]
C9, p1 p4 p6 ,t_1=[1 3 3 4],t_3=[0 1 3 4]
C10, p1 p2 p4 p6, t_1=[4 5 5 6],t_2=[2 2 3 3],t_3=[0 0 0 3]
C11, p1 p2 p5 p6, t_1=[1 5 5 6], t_2=[0 2 3 3] $[t_1 \ominus t_2]_{11}$=[1 2 3 4]
C12, p1 p3 p5 p6, t_1=[1 3.25 3.25 4], t_3=[0 0 0 0]
C13, p1 p4 p6, t_1=[1 3.25 3.25 4], t_3=[0 1 3 4]
C14, p1 p3 p5 p6, t_1=[1 3.4 3.4 4], t_3=[0 0 0 0]
C15, p1 p4 p6, t_1=[1 3.4 3.4 4], t_3=[0 1 3 4]
C16, p1 p5 p6, t_1=[0 0.4 2.4 4]
C17, p1 p5 p6, t_1=[0 0.25 2.25 4]

Intervalle de tir $D_f^j(t_i)$ et mesures de possibilité/nécessité sur les arcs sortant de C_j

$D^0(t_1)$ = [4 5 5 6], N = 1
$D^1(t_2)$ = [2 2 3 3], N = 1
$D^2(t_3)$ = [0 0 0 0], N = 1
$D^3(t_1)$ = [1 2 3 4], Π = 1, N = 0
$D^3(t_3)$ = [0 1 3 4], Π = 1, N = 0
$D^4(t_2)$ = [2 2 2 3], Π = 0.5
$D^4(t_3)$ = [0 0 1 3], N = 0.5
$D^5(t_4)$ = [0 0 0 0], Π = 1, N = 0
$D^5(t_5)$ = [0 0 0 0], Π = 1
$D^6(t_2)$ = [0 1 3 3],N = 1
$D^7(t_1)$ = [0 0 2 4], N = 1
$D^8(t_4)$ = [0 0 0 0], Π = 1, N = 0
$D^8(t_3)$ = [0 0 0 0], Π = 1, N = 0
$D^9(t_1)$ = [1 3 3 4]· Π = 1
$D^9(t_3)$ = [0 1 3 4]· Π = 1
$D^{10}(t_2)$ = [2 2 2 3]· Π = 1/3
$D^{10}(t_3)$ = [0 0 0 3]· N = 2/3
$D^{11}(t_2)$ = [0 2 3 3]· N = 1
$D^{12}(t_3)$ = [0 0 0 0]· N = 1
$D^{13}(t_1)$ = [0 3.07 3.07 4]· Π = 0.923
$D^{13}(t_3)$ = [0 1 3 4]· N = 0.077
$D^{14}(t_3)$ = [0 0 0 0]· N = 1
$D^{15}(t_1)$ = [0 3.12 3.12 4]· Π = 0.961
$D^{15}(t_3)$ = [0 1 3 4]· N = 0.039
$D^{16}(t_1)$ = [0 0.4 2.4 4]· N = 1
$D^{17}(t_1)$ = [0 0.25 2.25 4]N = 1

TAB. 11.1 – Les détails des classes et des arcs du graphe de la figure 11.4.a.

Dans le DEA de S. Cousy, nous avons aussi associé une sémantique stochastique à l'intervalle en utilisant les Réseaux de Petri Temporisés Stochastiques (RdPTS) [zGal97], défini comme le n-uplet $< RdPT, Fo >$, où RdPT est un réseau de Petri temporel et Fo est la fonction densité de probabilité définie sur les intervalles de temps associés aux transitions. Dans ce modèle, les densités de probabilité peuvent être de trois types : continue (exponentielle ou uniforme), discrète



a) Graphe de classe du RdPTF b) Graphe de classes du RdPT

FIG. 11.4 – Graphes de classes du réseau de la figure 11.3.

(impulsion de Dirac) et mixte (une partie uniforme et l'autre discrète). La densité mixte permet de privilégier certaines dates d'un intervalle temporel pour l'occurrence de l'événement associé à la transition.

Dans un premier temps, nous avons utilisé le graphe d'états probabilisé. Dans ce graphe, un état est un triplet (marquage, intervalle temporel des transitions sensibilisées, densité de probabilité sur cet intervalle). L'arc entre deux états est un triplet (transition t_i, l'instant de tir θ_i, la probabilité P_i de tir de t_i). Nous avons fait une analyse de scénario où l'instant de tir est déterminé à l'avance. Trois types de scénario ont été utilisés, proposés en [zGal97], où toutes les transitions sont franchies à une date qui correspond : à la moyenne de l'intervalle, à la borne minimale de l'intervalle ou à la borne maximale de l'intervalle. Soit le réseau de Petri Temporisés Stochastiques ayant comme structure le même réseau de la figure 11.3, dont les intervalles associés aux transitions t_i correspondent aux supports de $I_f^0(t_i)$ (figure .11.5).

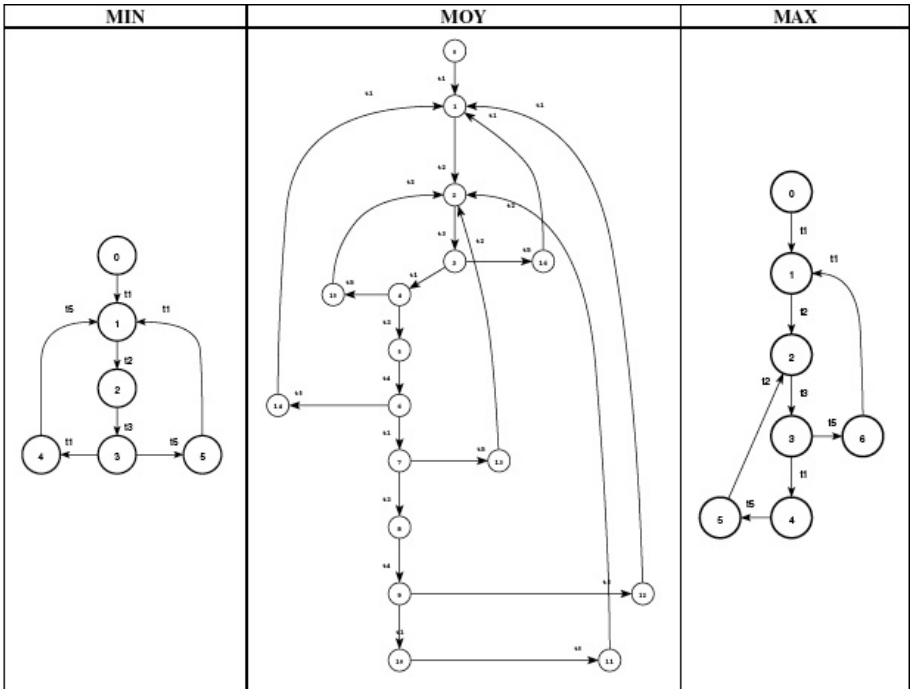


FIG. 11.5 – Graphe d'états probabilisé du réseau de Petri Temporisé Stochastique.

Cependant, il est intéressant de faire une analyse *exhaustive* du comportement dynamique, en considérant, comme ça a été le cas du graphe proposé pour les réseaux de Petri temporels flous, que toute transition sensibilisée peut être tirée à n'importe quel instant de son intervalle temporel. Cette analyse exhaustive nécessite une mise à jour complexe (produit de convolution) des intervalles temporels et des distributions de probabilité des transitions restées sensibilisées suite au tir d'une autre transition. Dans ce cas, il faut construire un graphe de classes d'états probabilisé, où une classe est un triplet (marquage, intervalle temporel des transitions sensibilisées, densité de probabilité sur cet intervalle). L'arc entre deux classes est maintenant un quadruplet (transition t_i, intervalle de tir $[a_i, sUB]$ (sUB est la plus petite des bornes max des intervalles associés aux transitions sensibilisées), la probabilité P_i de tir de t_i, θ, calcul de l'instant moyen du tir).

Nous avons calculé le graphe de classes d'états probabilisé (avec analyse *exhaustive*) pour l'exemple de la figure 11.3. Il apparaît que les formes de distributions, lors de la mise à jour, sont de plus en plus déformées. Il arrive donc un moment où les calculs deviennent très complexes. Cela résulte du fait que dans le RdP étudié, il y a dans la plupart des classes plusieurs transitions sensibilisées en

parallèles. De ce fait, à chaque mise à jour, avec l'utilisation du produit de convolution, le degré de la variable temporelle des distributions ne cesse d'augmenter et donc les déformations sont de plus en plus importantes. Cette méthode d'analyse, si nous voulons limiter sa complexité, et donc permettre son utilisation, nécessite des réseaux de Petri dans lesquels le parallélisme est limité et où nous avons des transitions de synchronisation. Il nous apparaît également que cette méthode fonctionne pour tout RdP ayant les bonnes propriétés (borné, vivant, réinitialisable) mais ces points restent à approfondir.

S'il est vraie que ce graphe ne permet pas une analyse *exhaustive*, il est vraie aussi qu'aujourd'hui, souvent, dans le domaine aéronautique et aussi nucléaire les changements d'états ne sont pas pris en compte : le raisonnement est fait pour un état, en utilisant par exemple les arbres de défaillances. Ces changements d'états sont prix en compte par les réseaux de Petri dit stochastiques en utilisant : i) les processus markoviens (taux transitions exponentiels), ii) les processus non-markoviens iii) la simulation de Monte-Carlo (aussi bien pour les processus markoviens que pour les non-markoviens). Dans le cas des processus non-markoviens il est utilisé en général la simulation de Monte-Carlo. Les analyses analytiques sont faites souvent dans le cas de processus markoviens, c'est pourquoi une analyse par scénario comme en [zJua97] a aussi son intérêt.

En fait, la théorie de possibilités permet de traiter de façon assez simple les informations incomplètes, en particulier si des données statistiques précises ne sont pas disponibles. Dans ce cas, il est plus intéressant de travailler avec les données disponibles (même si mal-connues) que d'essayer d'utiliser une information qui n'existe pas. Si des vérifications peuvent être faites avec des valeurs mal-connues dans des étapes amont de la conception, cela permet que les contraintes temporelles soient affinées par la suite.

La construction du graphe de classes d'un réseau de Petri temporel flou, permet, par rapport au réseau de Petri temporel, de quantifier l'occurrence des événements, par exemple, le phénomène d'écrasement dans l'exemple du protocole unidirectionnel (figure 11.3). Les mesures de possibilité et de nécessité permettent non seulement de répondre *Oui* ou *Non* à la vérification d'une propriété, mais aussi de dire qu'il *possible avec une valeur de possibilité/nécessité*. Par rapport au réseau de Petri stochastique, il est possible de faire une analyse *exhaustive* en utilisant les RdPTF, ce qui semble difficile dans le cas général des réseaux de Petri stochastique.

11.3 Graphe de classes pour les RdPT exprimant des contraintes quantitatives

Comme nous l'avons dit en introduction, notre but était de définir un graphe des classes pour les réseaux de Petri t-temporels avec sémantique forte qui permette d'associer à tout chemin de ce graphe l'ensemble des séquences de franchissements effectivement franchissables. Cela veut dire que l'on définit exactement (de façon quantitative) les contraintes temporelles que doivent vérifier les dates de franchissement.

La motivation de ces travaux vient de deux problèmes, que l'on peut appeler le problème du *branchement* et le problème du *chemin*. Considérons le RdPT de la figure 11.6.a, dont le graphe de classes en mode linéaire, préservant la propriété LTL [zBer04], est représenté figure 11.6.b. Le problème du branchement correspond au fait qu'une séquence du graphe peut ne pas être effectivement franchissable pour tous les états d'une classe. C'est le cas de la séquence t_2t_3 sur le graphe : elle n'est pas toujours franchissable, pour tous les états de la classe 2, dans le réseau

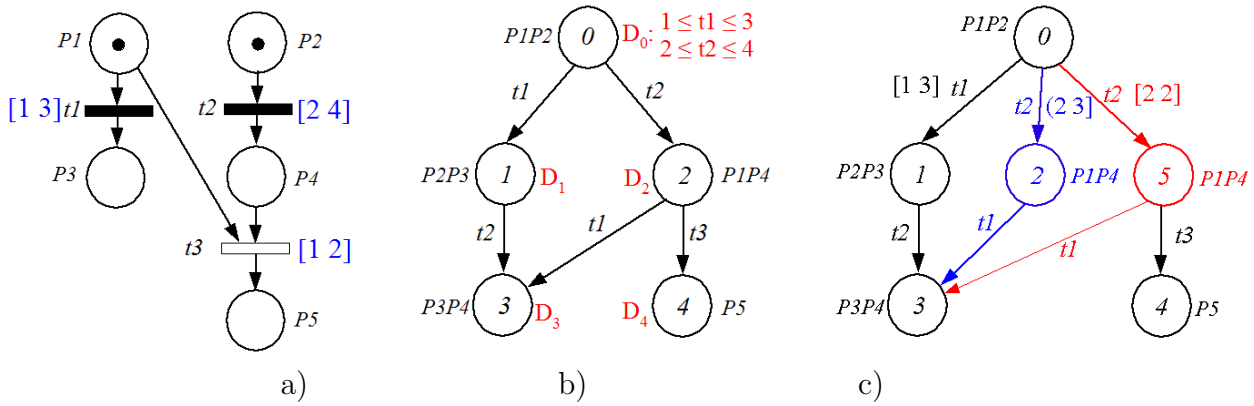


FIG. 11.6 – a) Réseau de Petri Temporel, b) Graphe de classes LTL, c) Graphe de classes CTL.

de Petri. En effet, si t_2 est franchie à l'instant 3, t_3 ne pourra être franchie que dans l'intervalle $3+[1,2]=[4,5]$. Or, d'après la sémantique forte, c'est le temps qui commande et t_1 avec $[1,3]$ doit être franchie avant t_3. Par contre, si t_2 est franchie à l'instant 2, t_3 sera en conflit avec t_1 et pourra être franchie. Le graphe de classes en mode arborescent, préservant les propriétés CTL, [zBer04], est représenté figure 11.6.c. Dans ce graphe, le problème du branchement est résolu : la classe 2 de la figure 11.6.b est partitionnée en deux classes 2 et 5. Si t_2 est franchie à l'instant 2 t_3 peut être franchie (elle est en conflit avec t_1), mais elle ne sera jamais franchie si t_2 est franchie entre $(2,3]$. Cependant, les états ne peuvent pas être distingués par leurs passé : la classe 3 présente des états atteints par le tir de t_1 et par le tir de t_2. Ce graphe présente, a nos yeux, deux inconvénients :

1. l'arc correspondant au tir d'une transition t à partir d'une classe $\mathcal{C}$ n'a pas suffisamment d'informations temporelles entre les différents événements (tir des transitions précédentes par exemple) ;
2. il ne garde pas les informations dans le passé (une classe peut être atteinte par plusieurs tir de transitions).

Pour pouvoir exprimer dans un graphe, des séquences de franchissement effectivement franchissables avec les contraintes temporelles minimales que doivent vérifier les dates de franchissement, il faudrait pouvoir partitionner la classe 3. En effet, suivant le chemin suivi les états atteints et regroupés dans la classe 3 ne sont pas les mêmes.

Nous avons exprimé dans la section 11.2 le calcul du graphe de classes d'un RdPT sous la forme d'un calcul d'intervalles. Ces intervalles peuvent être vus comme des contraintes binaires entre deux variables. C'est le cas clairement des équations 11.1 et 11.2 entre les variables (t_k, t_i). Mais c'est aussi le cas des équations 11.3 et 11.4 entre les variables (t_j, t_i), où t_j est la variable associée au tir de la transition qui amené à la classe C_j. Pour exprimer les dates de franchissements de transitions et les contraintes entre ces franchissements, nous avons choisi le formalisme des réseaux de contraintes temporelles (STN - Simple Temporal Network). Ce formalisme permet de représenter des contraintes binaires (entre deux variables) définies sous la forme d'intervalles convexes $[a_i, b_i]$. Le réseau de contraintes est complet si un intervalle est associé à chaque couple de variables ; il est minimal s'il existe une affectation de valeurs à toutes les variables. Le réseau de contraintes temporelles peut être exprimé sous la forme d'un graphe des coûts : chaque arc $(x_i, x_j) = [a_i, b_i]$ est remplacé par deux arcs : $(x_i, x_j) = b_i$ et $(x_j, x_i) = -a_i$. L'algorithme de Floyd-Warshall permet de calculer le plus court chemin $[d_{mij}, d_{Mij}]$ entre deux noeuds (x_i, x_j) et de construire ainsi un réseau de contraintes complet et minimal tel que d_{mij} est la meilleure borne inférieur possible et que d_{Mij} est la meilleure borne supérieure possible.

Le premier point à aborder est la détermination des événements à considérer dans un RdPT, et ainsi définir les variables temporelles associées aux dates de ces événements. Ces événements sont les suivants :

- *la sensibilisation d'une transition*
- *le début de l'intervalle de tir*
- *la fin de l'intervalle de tir*
- *le franchissement effectif de la transition.*

Les contraintes suivantes doivent être vérifiées entre ces événements : i) la date de sensibilisation d'une transition correspond à la date du dernier franchissement ayant contribué à la sensibilisation, ii) la date de franchissement doit appartenir à l'intervalle de sensibilisation, iii) la date de franchissement d'une transition doit avoir lieu avant la borne max des autres transitions sensibilisées (contrainte imposée par la *sémantique forte*, voir exemple représenté figure 11.1).

Pour pouvoir calculer directement sur le graphe les contraintes quantitatives, il faut que les classes puissent avoir les contraintes temporelles clairement définies. Une classe doit donner suffisamment d'information pour permettre le choix des dates de franchissement pour toutes les transitions suivantes (dans le futur). Elle doit donc donner exactement les contraintes temporelles que doivent vérifier la variable (associée au franchissement d'une transition) vis-à-vis des variables associées aux franchissements passés. Bien sur, il faut *oublier* une partie de ce passé, et conserver seulement parmi les variables associées aux franchissements passés et les contraintes temporelles qui les lient, celles qui sont non redondantes et délimitent effectivement les dates des transitions franchissables dans cette classe.

Le graphe de classes que nous avons proposé, est composé de noeuds représentant les classes $\mathcal{C}$ – données par une marquage M et un réseaux de contraintes temporelles exprimant les événements ayant sensibilisé les transitions franchissables à partir de $\mathcal{C}$ – et des arcs $(\mathcal{C}, \mathcal{C}')$ étiquetés par la transition franchie t_j et un un réseaux de contraintes temporelles décrivant les contraintes que t_j doit vérifier (figure 11.7). Nous avons d'abord traité le cas de transitions bornées [Car05c, Mao05a] et nous l'avons ensuite étendu au cas des transitions non bornées [Car05b].

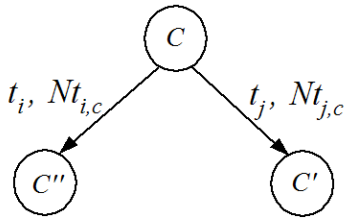


FIG. 11.7 – Graphe de classes

La classe initiale $\mathcal{C}_0$ est définie par un couple (M_0, Nc_0) où : M_0 est le marquage initial et Nc_0 est le réseau de contraintes temporelles composé par la variable x_0 représentant l'origine de temps.

Soit la séquence de franchissement $\sigma = t_1 ; \cdots ; t_i$; d'un réseau de Petri t-temporel, t_i est le dernier franchissement dans cette séquence. Soit $t_{s(k)}$ la transition qui a sensibilisé une transition t_k et o_i l'ordre de tir de la transition t_i dans la séquence.

Définition 1 *La classe d'état $\mathcal{C}$, obtenue après le franchissement de la transition t_i est définie par un couple $\{M, Nc\}$ où :*

- *M est le marquage atteint après le tir de t_i, nous supposons que pour ce marquage n transitions sont sensibilisées (y compris t_j),*

– *Nc est le réseau de contraintes temporelles simple comprenant les variables et les contraintes suivantes :*
  1. *la variable associée à la date du dernier franchissement de transition x_i^{oi},*
  2. *pour chaque transition t_k sensibilisée par M, la variable associée au franchissement de la transition ayant provoqué cette sensibilisation, soit $x_{s(k)}^{osk}$ $(k = 1 \ldots n)$.*
  3. *toutes les contraintes temporelles liant ces variables sous la forme d'un réseau complet et minimal.*

Soit t_j une transition parmi les n transitions franchissables dans la classe $\mathcal{C}$ (atteinte par le franchissement de t_i). Soit t_l, l ≠ j, les autres $n-1$ transitions sensibilisées dans la classe $\mathcal{C}$. $Nt_{j,c}$ est le réseau de contraintes temporelles délimitant le franchissement de t_j à partir de la classe $\mathcal{C}$.

La procédure de construction du réseau de contraintes $Nt_{j,c}$ consiste à partir du réseau de contraintes temporelles de la classe origine $\mathcal{C}$, et à :

1. ajouter la variable x_j^{oj} (associée au franchissement de t_j) et comme contrainte entre x_j^{oj} et x_{sj}^{osj} son intervalle statique $I(t_j)$,
2. ajouter les variables y_l^{ol} correspondant aux dates maximales de franchissement des autres transitions sensibilisées dans la classe
3. ajouter la contrainte $[0, \infty[$ entre les variables x_i^{oi} et x_j^{oj} pour exprimer le fait que t_j doit être franchie *après* t_i,
4. ajouter les contraintes $[0, \infty[$ entre les couples de variables x_j^{oj} et y_l^{ol} $(l \neq j)$, pour exprimer le fait que t_j doit être franchie *avant* la borne maximal de t_l,
5. appliquer l'algorithme de Floyd-Warshall. Si le réseau n'est pas cohérent, la transition t_j n'est pas franchissable,
6. effacer les variables y_l^{ol} et les contraintes auxquelles elles sont directement reliées

Lors du calcul du réseau de contraintes $Nt_{j,c}$, après l'application de l'algorithme de Floyd-Warshall, certaines des contraintes $C_{k,l}$ entre deux nœuds x_k et x_l peuvent se trouver modifiées, donc plus restreintes que leurs valeurs initiales dans le réseau Nc de la classe $\mathcal{C}$ (il faut souligner que, par construction, $Nt_{j,c}$ part de Nc). Cela implique que la transition t_j n'est pas franchissable à partir de tous les états de la classe, mais seulement à partir de certains d'entre eux. Cela définit une sous-classe d'une classe $\mathcal{C}$, notée $\mathcal{C}r$, qui est restreinte au franchissement d'une transition déterminée : $Nc_r = Nt_{j,c} \cap Nc$. soit $\mathcal{C}_p$ une classe précédente à $\mathcal{C}$ (figure 11.8). Le graphe de classes est modifié alors de la façon suivante : ajouter la nouvelle classe $\mathcal{C}r$; effacer l'arc $(\mathcal{C}, \mathcal{C}')$; ajouter un arc $(\mathcal{C}r, \mathcal{C}')$ étiquetté par Ntr_{j,c_r} ; ajouter un arc $(\mathcal{C}_p, \mathcal{C}r)$ ettiquété Nt'_{i,c_p} où les arcs (x_k, x_l) de Nt_{i,c_p} existant aussi en Nc_r sont remplacés par $(x_k, x_l)_{Nt_{i,c_p}} \cap (x_k, x_l)_{Nc_r}$. Si $Nt'_{i,c_p} \cap Nc_p \neq Nc_p$, cela implique que la transition t_i peut elle même induire une restriction de la classe $\mathcal{C}_p$, notée $\mathcal{C}_{pr}$, à partir de laquelle t_i est franchie et ainsi de suite.

Nous avons défini aussi la notion de classes équivalentes, en faisant la différence entre l'équivalence entre deux classes quelconques et l'équivalence avec la classe initiale. Dans ce dernier cas, l'ensemble des variables X de Nc doit être un singleton, X = $\{x_k\}$. En effet, il ne faut pas avoir une mémoire du passé. Quant au cas plus général des deux classes quelconques, il faut s'assurer que les variables sont les dates de franchissement d'une même transition, en vérifiant s'il existe une bijection entre les variables des deux classes. Dans le cas de deux classes restreintes, il faut, de plus, que les séquences franchissables définissant les restrictions soient les mêmes.

L'algorithme de construction du graphe des classes est composé de trois étapes : 1) construire le graphe de classes sans considérer les classes restreintes ; 2) vérifier pour chaque arc s'il y a

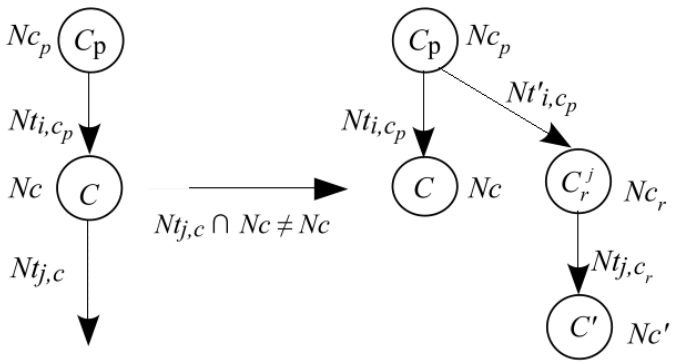


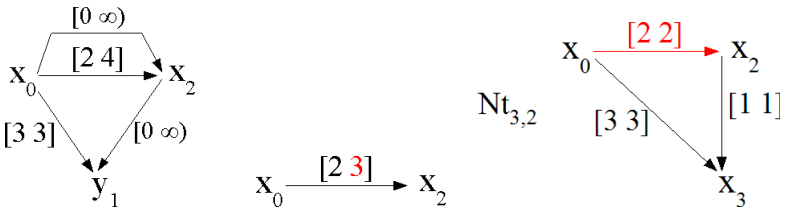
FIG. 11.8 – Classe restreinte $\mathcal{C}r$ de la classe $\mathcal{C}$

une restriction et créer récursivement les classes restreintes le cas échéant ; 3) vérifier s'il y a des relations de restriction entre les noeuds du graphe. À partir de cet algorithme l'outil GraphC a été implémenté en Java en utilisant l'environnement Eclipse, initialement par X. Mao [Mao05] et ensuite par A. Hamdani [Ham06]. La première version prenait en compte seulement des transitions bornées. Ensuite il a été pris en compte le cas général des transitions non bornées ; cette extension de l'outil GraphC a été par ailleurs facilité par l'utilisation du langage UML lors de la phase de conception de l'outil. La mise au point de l'outil GraphC a permis de tester des exemples plus compliqués que ceux utilisés pendant la phase de mise au point de l'algorithme. En particulier, cela a permis de comparer le graphe que nous avons proposé avec des graphes de classes déjà existants. Cet outil est disponible sur `http ://graphc.sourceforge.net`.

Le réseau de contraintes temporelles d'une séquence de transitions est donné par l'union des réseaux de contraintes de chaque transition de cette séquence. Lors de cette union, certaines contraintes peuvent être redondantes, mais des contraintes supplémentaires peuvent apparaître réduisant l'espace de solutions. Le réseau de contraintes temporelles de la séquence indique globalement les contraintes concernant les dates de franchissement de toutes les transitions.

Considérons le RdPT de la figure 11.6.a. La classe initiale $\mathcal{C}_0$ est donné par $(M_0 : P1P2, Nc_0 : x_0)$, où x_0 est l'origine du temps. A partir de $\mathcal{C}_0$, les transitions sensibilisées sont t_1 et t_2. Si l'on considère le tir de $t2$, le réseau de contraintes $Nt_{2,0}$ délimitant ce tir est celui de la figure 11.9.a, avec au départ $Nt_2 = Nc_0$, donné parl nœud x_0 auquel ont été ajoutés :
– le noeud x_2 (correspondant au tir de $t2$), et la contrainte $(x_0, x_2) = I(t2) = [2\ 4]$, correspondant à l'intervalle statique de t_2,
– le noeud y_1 (correspondant à borne max. de $t1$), $(x_0, y_1) = [d_M\ d_M] = [3\ 3]$,
– les arcs $[0\ \infty)$ pour exprimer l'ordre des tirs : entre (x_0, x_2), et entre (x_2, y_1) : $t2$ avant $t1$.
Après appliquer l'algorithme de Floyd-Warshall et effacer le nœud y_1 et les contraintes associées le réseau de contraintes obtenu est celui de la figure 11.9.b. On peut observer que l'intervalle de tir de t_2, initialement à [2 4] a été diminué à [2 2] à cause de l'intervalle de tir de t_1 (sémantique forte).

Le tir de t_2 amène à la classe $\mathcal{C}_2$, avec marquage $M_2 = P1\ P4$ et réseau de contraintes Nc_2 ayant au départ la variable x_2 correspondant au tir de t_2. Les transitions sensibilisées dans cette classe sont t_1 et t_3. Il faut donc ajouter :
– le noeud x_0 (correspondant à l'événement qui a sensibilisé t1 : $s(t_1) = 0$), déjà présent,
– le noeud x_2 (correspondant à l'événement qui a sensibilisé t3 : $s(t_3) = 2$),
– les contraintes entre x_0 et x_2 existantes dans le réseau $Nt_{2,0}$: $(x_0, x_2) = [2\ 3]$.
Le réseau de contraintes Nc_2 est donné par $(x_0, x_2) = [2, 3]$ (il est aussi représenté par la fi-



a) réseau initial $Nt_{2,0}$ b) $Nt_{2,0}$ c)

FIG. 11.9 – Réseaux de contraintes associés aux tirs de transitions

gure 11.9.b). Les réseau de contrainte $Nt_{3,2}$ lié au tir de t_3 à partir de la classe $\mathcal{C}_2$ est représenté figure 11.9.c.

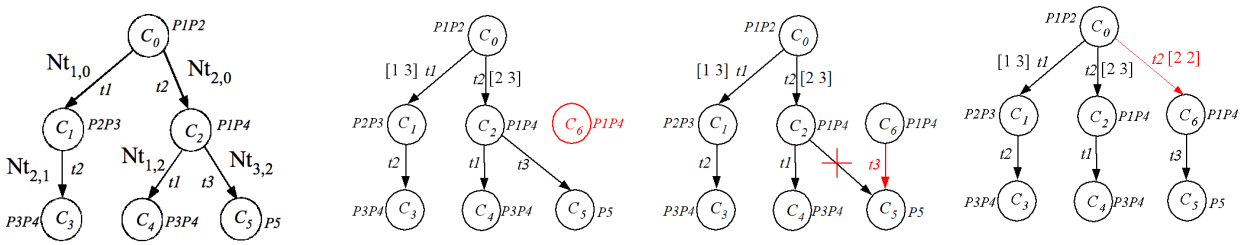


FIG. 11.10 – Construction du graphe de classe

Une fois le graphe de classes construit (figure 11.10.a) il faut comparer les réseaux de contraintes $Nt_{i,c}$ des arcs sortant d'une classe $\mathcal{C}$ avec le réseau de contraintes de cette classe. S'il est plus restreint (il existe un arc dont l'intervalle est plus restreint que celui de la classe), il implique que la transition t_i peut être franchie à partir de seulement certains états de cette classe. Dans ce cas, il faut créér une classe restreinte où t_i peut être franchie. C'est le cas de l'arc $Nt_{3,2}$ sortant de $\mathcal{C}_2$, où $(x_0, x_2)_{Nt_{3,2}} = [2, 2] \subset (x_0, x_2)_{Nc_2} = [2, 3]$. La classe $\mathcal{C}_6$, restreinte de $\mathcal{C}_2$, est créée (figure 11.10.b), avec $Nc_6 : (x_0, x_2) = [2, 2]$. Ensuite il faut effacer l'arc ($\mathcal{C}_2$,$\mathcal{C}_5$) et créer l'arc ($\mathcal{C}_2$,$\mathcal{C}_6$) (figure 11.10.c). Il faut connecter la classe restreinte avec les prédécesseurs de la classe $\mathcal{C}_2$, propager ces contraintes vers l'arrière, et vérifier s'il y a d'autres classes restreintes. Dans cette exemple, il faut créer l'arc ($\mathcal{C}_0$,$\mathcal{C}_6$). Le nouveau réseau de contrainte $Nt'_{2,0}$ est donné par $(x_0, x_2) = [2, 2]$. Comme les réseaux $Nt'_{2,0}$ et Nc_0 n'ont pas d'arcs en commun, l'algorithme s'arrête et le graphe de la figure gure 11.10.d est le graphe de classes obtenu pour le RdPT de la figure 11.6.a. Un exemple complet avec des boucles élémentaires a été traité dans le rapport de M2R de A. Hambdani [Ham06].

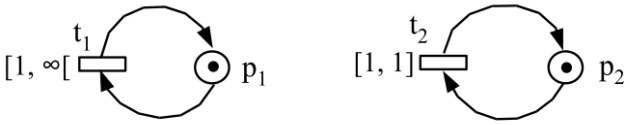
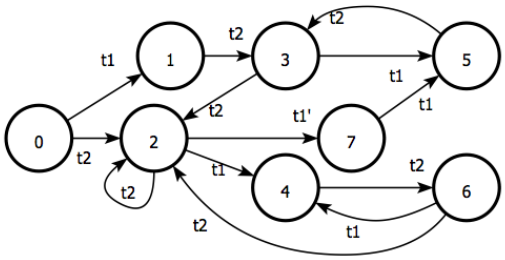


FIG. 11.11 – RdP t-temporel non bornée

Dans le cas de RdPT avec des transitions non bornées, avec $I(t_i) = [a_i, \infty[$, $a_i \in Q^+$, le nombre de classes mode $\mathcal{C}$ peut être infini.

Considérons par exemple la séquence $\sigma = t_2; t_2; (t_2)^*$ dans le RdPT de la figure 11.11, avec $I_s(t_1) = [1, \infty)$ et $I_s(t_2) = [1, 1]$. Si la définition de classe faite précédemment est utilisée, nous avons, à partir de la classe $\mathcal{C}_0 = (p_1p_2, x_0)$, les classes suivantes (toutes avec marquage p_1p_2) avec les réseaux



$C0, Nc_0 : x_0, T_\infty = \emptyset$
$C1, Nc_1 : C_{0,1} = [11], T_\infty = \emptyset$
$C2, Nc_2 : x_2, T_\infty = \{t_1\}$
$C3, Nc_3 : C_{1,2} = [00], T_\infty = \emptyset$
$C4, Nc_4 : C_{2,1} = [01], T_\infty = \emptyset$
$C5, Nc_5 : C_{2,1} = [11], T_\infty = \emptyset$
$C6, Nc_6 : C_{1,2} = [01], T_\infty = \emptyset$
$C7, Nc_7 : C_{2,1} = [00], T_\infty = \emptyset$

FIG. 11.12 – Graphe de classes du réseau de Petri de la fig. 11.11.a

de contraintes : $Nc_1 : (x_0, x_2) = [1\ 1]$, $Nc_2 : (x_0, x_2^2) = [2\ 2]$, ..., $Nc_n : (x_0, x_2^n) = [n\ n]$, où x_0 correspond à l'origine du temps et x_2^i est l'événement correspondant au dernier tir de t_2 amenant à $\mathcal{C}_i$. Ces deux événement doivent être conservés puisque x_0 est la date de sensibilisation de t_1 et x_2^i celle de t_2^i. En fait, si l'on considère l'intervalle dynamique $I^i(t_1)$ de t_1 (l'intervalle statique retranché de la date de tir dans $\mathcal{C}_i$), il est toujours égale à $(I_s(t_1) - C_{0,2}(x_0, x_2^i)) \cap [0\ \infty) = [0, \infty)$. Donc, le tir de t_2 ne contraint pas le tir de t_1. Cela veut dire en fait que dans les classe $\mathcal{C}_1$ à $\mathcal{C}_n$, il n'est pas nécessaire de garder la contrainte $C_{0,2^i}(x_0, x_2^i)$ et donc le nœud x_0 peut être enlevé de $Nc_i, i \geq 1$. Ce n'est pas le cas de $\mathcal{C}_0$, où $I(t_1) = [1\ \infty)$. Il faut garder l'information concernant le fait que t_1 est franchissable sous contraintes ou non.

Nous avons donc introduit dans le réseau de contraintes Nt délimitant le franchissement de t_j à partir de $\mathcal{C}$ deux nouvelles informations :
– une nouvelle variable z_l^{ol} correspondant à la borne minimale de l'intervalle statique pour chaque transition non-bornée sensibilisée dans la classe $\mathcal{C}$,
– l'ensemble T_∞ de transitions non-bornées t non contraintes par le tir de t_j amenant dans $\mathcal{C}$.
L'ensemble T_∞ est aussi ajouté aux informations associées à la classe. Les détails de la construction du graphe pour les RdPT avec des transitions non bornées sont présentés dans [Car05b]. Le graphe de classes pour l'exemple de la figure 11.11 est représenté sur la figure 11.12 avec le détail des classes. Toutes les classes ont le marquage p_1p_2. La classe $\mathcal{C}_7$ est la classe restreinte de la classe $\mathcal{C}_4$.

Le graphe de classes en mode C permet d'obtenir les contraintes temporelles qui doivent être vérifiée par chaque tir de transition dans une séquence de tir. Si l'on compare avec les graphes existants tels que [zYon98, zBer04], une première différence est que nous utilisons les réseaux de contraintes (STN) en lieu de graphe de régions géométriques pour traiter les informations temporelles. [zYon98] est plus proche de notre approche, avec cependant des différences. Dans notre cas, quand une classe restreinte $\mathcal{C}r$ est crée, la classe initiale $\mathcal{C}$ est conservée au lieu d'être remplacée par une classe qui est complémentaire à $\mathcal{C}r$. Nous ne gardont pas toutes les contraintes du passé et surtout nous avons traité les transitions non-bornées. Par rapport à [zBer04], nous avons comparé notre approche avec deux les graphes de classes en mode linéaire (W) et en mode atomique (A), qui permettent de vérifier, respectivement, les propriétés LTL et CTL : Dans le mode W, une classe est donné par son marquage, le domaine temporel des transitions sensibilisés et les contraintes (non-redondantes) existant entre ces transitions dans le passé. Dans le mode A, une classe est donné par son marquage et un horloge pour chaque transition sensibilisée (si la transition vient d'être sensibilisée, l'horloge(t)=0 sinon elle prend la valeur précédente). Le mode A permet de différencier les états pour lesquels il y a un conflit entre deux transitions de ceux pour lesquels

une seule transitions est franchissable (cas des classes 2 et 5 dans la figure 11.6.c). Dans notre cas (figure 11.10.d), considérons les classes $\mathcal{C}_2$ et $\mathcal{C}_6$ (classe restreinte de $\mathcal{C}_2$) : la transition t_1 peut être franchie à partir de tous les états de $\mathcal{C}_2$ (indépendamment que t_3 soit ou pas franchie). Mais $\mathcal{C}_6$ a été définie pour caractériser les contraintes temporelles que t_3 doit vérifier et donc t_1 n'apparaît pas comme sortie de ce noeud. Une autre différence apparaît dans la façon dont le passé est mémorisé. Notre approche permet d'obtenir directement[1] l'ensemble de contraintes temporelles d'une séquence. Bien que l'obtention des contraintes quantitatives liées à une séquence de tir ne soit pas nécessaire pour la vérification des propriétés, elle est fondamentale pour aider le concepteur à ajuster les paramètres qui permettent de vérifier ces propriétés.

Ces travaux constituent une première étape avant l'introduction de contraintes temporelles floues ; à ce titre ils constituent une phase préliminaire du projet Féria *Les réseaux de Petri temporels flous : étude de propriétés et application aux SMA* entre l'IRIT et le LAAS. J'ai animé ce projet entre 2004 et 2006 avec G. Juanole (LAAS). Le cadre général de ce projet est l'analyse d'un système dynamique où les informations temporelles sont connues de façon imparfaite.

11.4 Graphe de classes exprimant des contraintes quantitatives pour les RdPT flous

Nous avons montré dans la section 11.2 que la théorie de possibilités permet d'allier l'expressivité et la simplicité et traiter de façon efficace des informations dont l'incertitude apparaît sous la forme d'une ignorance partielle (plutôt que de nature aléatoire). On se situe dans une phase de conception en amont, où toutes les valeurs des paramètres ne sont pas connues avec certitude, mais on souhaite connaître les contraintes quantitatives entre les événements qui ont lieu dans le système. De la même façon où nous avons étendu le graphe de classes en mode linéaire, proposé par [zBer04], au cas flou (section 11.2), nous avons aussi étendu le graphe de classes en mode C (section 11.3) au cas flou, que nous avons appelé GraphC flou. Une première différence entre le GraphC flou et le graphe flou présenté en [Car05a] est donc le type de classe. La deuxième est par rapport à la génération du graphe. En [Car05a], lors du calcul des mesures de possibilité et de nécessité du tir des transitions, nous avons utilisé directement les intersections et soustractions entre les intervalles flous. Or, les intervalles ainsi obtenus ne sont pas toujours trapézoïdaux (figure 11.13) mais il faut les approximer par un trapèze le plus *proche* possible, c'est-à-dire, avec la distribution la plus similaire pour introduire le moins d'erreur possible. D'une part nous avons le problème de trouver la distribution la plus similaire. D'autre part, nous avions déjà un outil pour générer le graphe pour le cas imprécis. La théorie de possibilités permet de traiter l'ensemble des cas de façon ordonnée au lieu de considérer les cas individuellement : les fenêtres temporelles floues représentent des ensembles emboîtées d'intervalles temporels. En effet, un ensemble flou peut être considéré comme une collection des ensembles (classiques) emboîtés appelés α-coupes, comme représenté figure 11.14. Et si l'on connaît toutes les α-coupes (emboîtées) d'un ensemble flou non-connu, on peut construire la fonction d'appartenance de cet ensemble.

Nous avons donc choisi de transformer un réseau de Petri temporel flou en plusieurs réseaux de Petri temporels en utilisant le concept de α-coupe [zDub88]. Pour chaque réseau de Petri temporel, un graphe de classe est généré avec l'outil GraphC. Ensuite, il faut composer tous les graphes ainsi

[1] L'outil Tina propose depuis 2006, le calcul des contraintes temporels entre les tirs de transition d'une séquence à la volé.

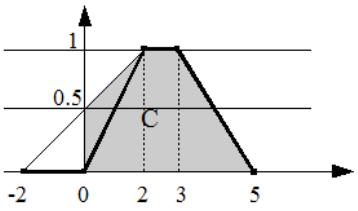


FIG. 11.13 – Un ensemble flou non-trapezoïdale.

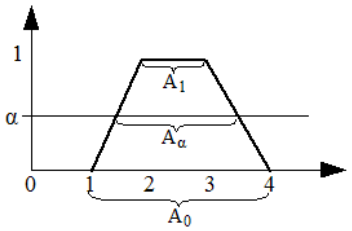


FIG. 11.14 – Exemple d'une α-coupe

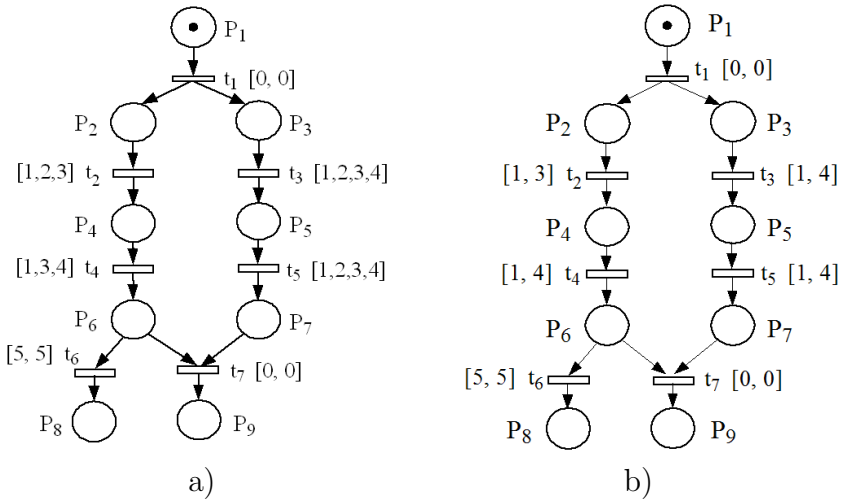


FIG. 11.15 – a) Réseau de Petri temporel flou, b) 1-RdPT

générés dans un graphC flou, en utilisant l'algorithme présenté en [Car06b]. Nous présentons ici un exemple d'application.

Le GraphC flou $G_F = (\mathcal{N}_F, \mathcal{A}_F)$ est une extension du GraphC au cas des réseaux de Petri temporels flous. Cela veut dire que les réseaux de contraintes temporelles associés aux classes et aux arcs deviennent des réseaux de contraintes temporelles floues. Les problèmes d'approximation des ensembles flous non-trapezoïdales sont évités en travaillant sur les graphes générés par chaque α-coupe séparément. Le GraphC flou est obtenu de la façon suivante :

- décomposition d'un RdPTF en k RdPT : Il faut décomposer les intervalles flous $I_f^0(t_i)$ associés à chaque transition t_i d'un RdPTF en k α-coupes : on obtient, pour chaque intervalle flou, k intervalles (classiques) $I_\alpha^0(t_i)$. Ensuite, on construit un α-RdPT composé de tous les intervalles $I_\alpha^0(t_i)$, pour un niveau d'α-coupe ;
- génération des k graphC G^α pour chaque α-RdPT ;
- composition des k graphes G^α en un GraphC flou : il faut *emboîter* les réseaux de contraintes des classes et des arcs des différents G^α. Bien que le problème de *matching* entre deux graphes soit un problème NP complet, dans ce cas le problème est plus simple parce que les séquences de tir sont préservés. Puisque $G^{\alpha_1} \subseteq G^\alpha$ for $\alpha_1 \geq \alpha$, le seul problème est trouver les classes de G^{α_1} sont contenues en celles de G^α. En effet, la structure des graphes (nombre de noeuds et d'arcs) peut être différente dû aux classes restreintes qui peuvent apparaître ou disparaître quand α change.

La figure 11.15.a représente un réseau de Petri temporel flou, modélisant un processus principal qui appelle un processus distant. Le processus principal est décrit par les transitions t_1, t_2, t_4, t_6 et t_7. Le processus distant est décrit par les transitions t_3 et t_7. La transition t_7 correspond à la réception de la réponse (jeton en p_9) après que l'appel a été fait (jeton en p_3). Une des question que l'on

peut se poser sur le comportement de ce système est si la réponse arrive ou pas à temps. Le but est d'avoir un compromis entre une longue attente inutile en cas de défaillance ou la perte d'une réponse tardive. Il est donc intéresant d'évaluer les différents compromis selon la valeur du chien de garde associé à t_6.

Nous avons utilisé 10 α-coupes obtenant ainsi dix α-RdPT. Le réseau de Petri temporel 0-RdPT de la figure 11.15.b est la représentation graphique pour $\alpha = 0$ (le support). Les intervalles de temps pour les α-RdPT avec $\alpha = 1, 0.9, 0.5$ et 0.1 sont :
- pour $\alpha = 1$ (le noyau) : $I_1(t_1) = I_1(t_7) = [0\ 0], I_1(t_2) = [2\ 2], I_1(t_3) = I_1(t_5) = [2\ 3], I_1(t_4) = [3\ 3]$, $I_1(t_6) = [5.\ 5.]$.
- pour $\alpha = 0.9$: $I_{.9}(t_1) = I_{.9}(t_7) = [0\ 0], I_{.9}(t_2) = [1.9\ 2.1], I_{.9}(t_3) = I_{.9}(t_5) = [1.9\ 3.1], I_{.9}(t_4) = [2.8\ 3.1]$ and $I_{.9}(t_6) = [5.\ 5.]$.
- pour $\alpha = 0.5$: $I_{.5}(t_1) = I_{.5}(t_7) = [0\ 0], I_{.5}(t_2) = [1.5\ 2.5], I_{.5}(t_3) = I_{.5}(t_5) = [1.5\ 3.5], I_{.5}(t_4) = [2.0\ 3.5]$ and $I_{.5}(t_6) = [5.\ 5.]$.
- pour $\alpha = 0.1$: $I_{.1}(t_1) = I_{.1}(t_7) = [0\ 0], I_{.1}(t_2) = [1.1\ 2.9], I_{.1}(t_3) = I_{.1}(t_5) = [1.1\ 3.9], I_{.1}(t_4) = [1.2\ 3.9]$ and $I_{.1}(t_6) = [5.\ 5.]$.

La figure 11.16 montre les graphes de classes obtenus lors de l'étape de génération des graphes des α-RdPT. On peut voir que certains graphes ont exactement la même structure, par exemple les graphes $G^{0.5}$ et $G^{0.4}$ de la figure 11.16.c.

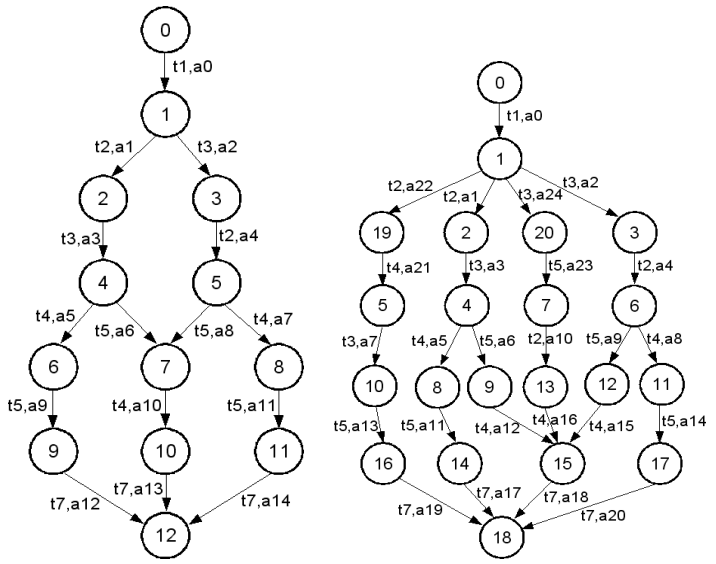
La figure 11.17 montre un tableau avec les valeurs des contraintes pour chaque classe, disposées de façon à montrer les inclusions de classes trouvées durant l'étape de composition du GraphC flou. Les deux premières colonnes du tableau (à gauche) indiquent le marquage et les contraintes $C_{i,j}$ entre les noeuds (x_i, x_j) du réseau de contraintes pour toutes les classes. Les colonnes suivantes, sous la forme de cinq (sous)-tableaux, indiquent pour chaque classe (la première colonne du sous-tableau) la valeur de la contrainte $C_{i,j}$ sous forme d'un intervalle temporel pour chaque α-coupe. En regardant le tableau de gauche à droite, chaque ligne indique : le marquage M_i, le STN Nc_i, et pour chaque ensemble de colonnes, le nom des classes en G^α et la valeur de la contrainte. Par exemple, pour $\alpha = 0.9$, le graphe de classes $G^{0.9}$ possède 14 classes ; la classe $\mathcal{C}_2 = \{M, Nc_2\}$ de ce graphe est caractérisée par le marquage $M = p_3p_4$ et réseau de contraintes Nc_2 ayant les noeuds $\{x_1, x_2\}$ et $C_{1,2} = [1.9\ 2.1]$. Les classes $\mathcal{C}_0$, $\mathcal{C}_1$, $\mathcal{C}_{12}$ et $\mathcal{C}_{13}$ sont caractérisées par des réseaux de contraintes ayant un seul noeud, c'est pourquoi la cellule correspondante est vide, par exemple, $Nc_1 : x_1$.

Les sous-tableaux ne sont pas de même taille ; par exemple, $G^{0.9}$ possède 14 classes et $G^{0.5}$ en possède 18. Lors de la construction de l'ensemble flou représentant la contrainte d'une classe flou $\mathcal{C}_{F_i}$, la hauteur de cet ensemble est donné par la valeur de la dernière α-coupe. Par exemple, pour $\mathcal{C}_{F_{14}}$ la hauteur de $C_{1,4}$ est de 0.5.

La liste des classes restreintes est la suivante : $[\mathcal{C}_{17}]^{0.5} = [\mathcal{C}_2]^{0.5}_r$, $[\mathcal{C}_{19}]^{0.3} = [\mathcal{C}_2]^{0.3}_r$, $[\mathcal{C}_{20}]^{0.3} = [\mathcal{C}_3]^{0.3}_r$, $[\mathcal{C}_{22}]^{0.2} = [\mathcal{C}_2]^{0.2}_r$, $[\mathcal{C}_{23}]^{0.2} = [\mathcal{C}_3]^{0.2}_r$, $[\mathcal{C}_{24}]^{0.2} = [\mathcal{C}_{10}]^{0.2}_r$, $[\mathcal{C}_{25}]^{0.2} = [\mathcal{C}_5]^{0.2}_r$, $[\mathcal{C}_{26}]^{0.2} = [\mathcal{C}_{22}]^{0.2}_r$, $[\mathcal{C}_{27}]^{0.2} = [\mathcal{C}_{16}]^{0.2}_r$, $[\mathcal{C}_{25}]^0 = [\mathcal{C}_2]^0_r$, $[\mathcal{C}_{22}]^0 = [\mathcal{C}_3]^0_r$, $[\mathcal{C}_{24}]^0 = [\mathcal{C}_5]^0_r$ and $[\mathcal{C}_{26}]^0 = [\mathcal{C}_{16}]^0_r$.

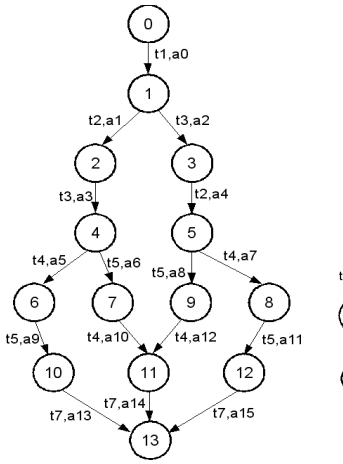
La structure du GraphC flou G_F est la même que le graphe de la figure 11.16.e. Les noms des classes floues sont indiquées dans la dernière colonne de la figure 11.17. L'intervalle temporel flou associé à une contrainte est obtenu en emboîtant les intervalles en partant de la droite vers la gauche. La figure 11.18 représente les contraintes associées aux classes $\mathcal{C}_{F_3}$ et $\mathcal{C}_{F_{14}}$. Les réseaux de contraintes $Nt_{i,c}$ associés aux transitions t_i sortant de la classe $\mathcal{C}$ ne sont représentés.

Lors de la vérification d'inclusion de classes $[\mathcal{C}]^{\alpha_1} \subseteq [\mathcal{C}]^\alpha$, cette vérification est directe si G^{α_1} et G^α ont exactement la même structure. C'est le cas pour : 1) $G^{0.5}$ et $G^{0.4}$, 2) $G^{0.9}$, $G^{0.8}$, $G^{0.7}$ et $G^{0.6}$ et

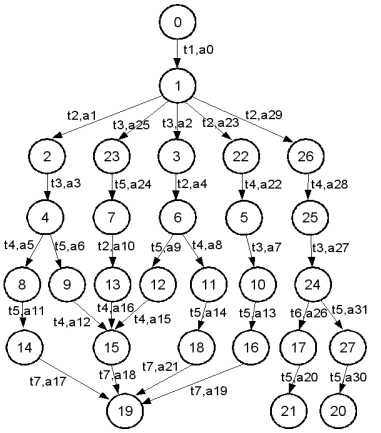


a) G^1

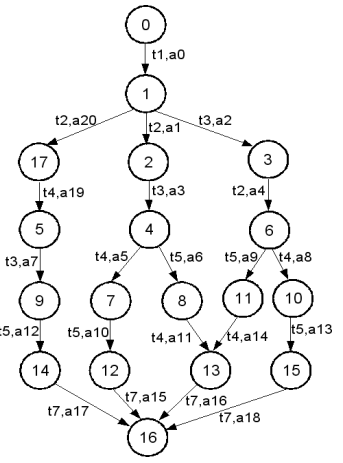
d) $G^{0.3}$



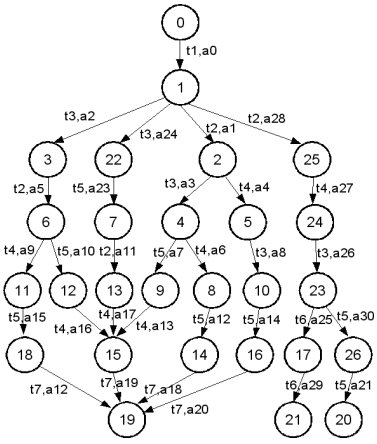
b) $G^{0.9}$, $G^{0.8}$, $G^{0.7}$ et $G^{0.6}$



e) $G^{0.2}$ et $G^{0.1}$



c) $G^{0.5}$ et $G^{0.4}$



f) G^0

FIG. 11.16 – Graphes de classes G^α

M_i	Nc_i	$\mathcal{C}_i$	$\alpha = 1$
p_1	x_0	$\mathcal{C}_0$	
p_2p_3	x_1	$\mathcal{C}_1$	
p_3p_4	$C_{1,2}$	$\mathcal{C}_2$	[2 2]
p_2p_5	$C_{1,3}$	$\mathcal{C}_3$	[2 2]
p_4p_5	$C_{2,3}$	$\mathcal{C}_4$	[0 1]
p_4p_5	$C_{3,2}$	$\mathcal{C}_5$	[0 0]
p_5p_6	$C_{3,4}$	$\mathcal{C}_6$	[2 3]
p_4p_7	$C_{2,5}$	$\mathcal{C}_7$	[2 3]
p_5p_6	$C_{3,4}$	$\mathcal{C}_8$	[3 3]
p_4p_7	$C_{2,5}$	$\mathcal{C}_9$	[2 3]
p_6p_7	$C_{4,5}$	$\mathcal{C}_{10}$	[0 1]
p_6p_7	x_4	$\mathcal{C}_{11}$	
p_6p_7	$C_{4,5}$	$\mathcal{C}_{12}$	[0 0]
p_9	x_7		

$\mathcal{C}_i$	$\alpha = 0.9$	$\alpha = 0.8$	$\alpha = 0.7$	$\alpha = 0.6$
$\mathcal{C}_0$				
$\mathcal{C}_1$	[1.9 2.1]	[1.8 2.2]	[1.7 2.3]	[1.6 2.4]
$\mathcal{C}_2$	[1.9 2.1]	[1.8 2.2]	[1.7 2.3]	[1.6 2.4]
$\mathcal{C}_3$	[0 1.2]	[0 1.4]	[0 1.6]	[0 1.8]
$\mathcal{C}_4$	[0 0.2]	[0 0.4]	[0 0.6]	[0 0.8]
$\mathcal{C}_5$	[1.6 3.1]	[1.2 3.2]	[0.8 3.3]	[0.4 3.4]
$\mathcal{C}_6$	[1.9 3.1]	[1.8 3.2]	[1.7 3.3]	[1.6 3.4]
$\mathcal{C}_7$	[2.8 3.1]	[2.6 3.2]	[2.4 3.3]	[2.2 3.4]
$\mathcal{C}_8$	[1.7 3.1]	[1.4 3.2]	[1.1 3.3]	[0.8 3.4]
$\mathcal{C}_9$	[0 1.5]	[0 2]	[0 2.5]	[0 3]
$\mathcal{C}_{10}$				
$\mathcal{C}_{11}$	[0 0.3]	[0 0.6]	[0 0.9]	[0 1.2]
$\mathcal{C}_{12}$				
$\mathcal{C}_{13}$				

$\mathcal{C}_i$	$\alpha = 0.5$	$\alpha = 0.4$
$\mathcal{C}_0$		
$\mathcal{C}_1$	[1.5 2.5]	[1.4 2.6]
$\mathcal{C}_2$	[1.5 2.5]	[1.4 2.6]
$\mathcal{C}_3$	[0 2]	[0 2.2]
$\mathcal{C}_4$	[0 1]	[0 1.2]
$\mathcal{C}_6$	[0 3.5]	[0 3.6]
$\mathcal{C}_7$	[1.5 3.5]	[1.4 3.6]
$\mathcal{C}_8$	[2 3.5]	[1.8 3.6]
$\mathcal{C}_{10}$	[0.5 3.5]	[0.2 3.6]
$\mathcal{C}_{11}$	[0 3.5]	[0 3.6]
$\mathcal{C}_{12}$		
$\mathcal{C}_{13}$		
$\mathcal{C}_{15}$	[0 1.5]	[0 1.8]
$\mathcal{C}_{16}$		

$\mathcal{C}_i$	$\alpha = 0.3$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.3 2.7]
$\mathcal{C}_2$	[1.3 2.7]
$\mathcal{C}_3$	[0 2.4]
$\mathcal{C}_4$	[0 1.4]
$\mathcal{C}_6$	[0 3.7]
$\mathcal{C}_8$	[1.3 3.7]
$\mathcal{C}_9$	[1.6 3.7]
$\mathcal{C}_{11}$	[0 3.7]
$\mathcal{C}_{12}$	[0 3.7]
$\mathcal{C}_{14}$	
$\mathcal{C}_{15}$	
$\mathcal{C}_{17}$	[0 2.1]
$\mathcal{C}_{18}$	

$\mathcal{C}_i$	$\alpha = 0.2$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.2 2.8]
$\mathcal{C}_2$	[1.2 2.8]
$\mathcal{C}_3$	[0 2.6]
$\mathcal{C}_4$	[0 1.6]
$\mathcal{C}_6$	[0 3.8]
$\mathcal{C}_8$	[1.2 3.8]
$\mathcal{C}_9$	[1.4 3.8]
$\mathcal{C}_{11}$	[0 3.8]
$\mathcal{C}_{12}$	[0 3.8]
$\mathcal{C}_{14}$	
$\mathcal{C}_{15}$	
$\mathcal{C}_{17}$	[0 2.4]
$\mathcal{C}_{18}$	
$\mathcal{C}_{19}$	

$\mathcal{C}_i$	$\alpha = 0.1$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.1 2.9]
$\mathcal{C}_2$	[1.1 2.9]
$\mathcal{C}_3$	[0 2.8]
$\mathcal{C}_4$	[0 1.8]
$\mathcal{C}_6$	[0 3.9]
$\mathcal{C}_8$	[1.1 3.9]
$\mathcal{C}_9$	[1.2 3.9]
$\mathcal{C}_{11}$	[0 3.9]
$\mathcal{C}_{12}$	[0 3.9]
$\mathcal{C}_{14}$	
$\mathcal{C}_{15}$	
$\mathcal{C}_{17}$	[0 2.7]
$\mathcal{C}_{18}$	
$\mathcal{C}_{19}$	

$\mathcal{C}_i$	α=0
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1 3]
$\mathcal{C}_2$*	[1 3]
$\mathcal{C}_3$	[0 3]
$\mathcal{C}_4$	[0 2]
$\mathcal{C}_6$	[0 4]
$\mathcal{C}_8$	[0 4]
$\mathcal{C}_9$	[1 4]
$\mathcal{C}_{11}$	[0 4]
$\mathcal{C}_{12}$	[0 4]
$\mathcal{C}_{14}$	
$\mathcal{C}_{15}$	
$\mathcal{C}_{17}$	[0 3]
$\mathcal{C}_{18}$	
$\mathcal{C}_{19}$	

$\mathcal{C}_i$	$\alpha = 0.6$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.6 2.4]
$\mathcal{C}_2$	[1.6 2.4]
$\mathcal{C}_3$	[0 1.8]
$\mathcal{C}_4$	[0 0.8]
$\mathcal{C}_5$	[0.4 3.4]
$\mathcal{C}_6$	[1.6 3.4]
$\mathcal{C}_7$	[2.2 3.4]
$\mathcal{C}_8$	[0.8 3.4]
$\mathcal{C}_9$	[0 3]
$\mathcal{C}_{10}$	
$\mathcal{C}_{11}$	[0 1.2]
$\mathcal{C}_{12}$	
$\mathcal{C}_{13}$	

$\mathcal{C}_i$	$\alpha = 0.7$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.7 2.3]
$\mathcal{C}_2$	[1.7 2.3]
$\mathcal{C}_3$	[0 1.6]
$\mathcal{C}_4$	[0 0.6]
$\mathcal{C}_5$	[0.8 3.3]
$\mathcal{C}_6$	[1.7 3.3]
$\mathcal{C}_7$	[2.4 3.3]
$\mathcal{C}_8$	[1.1 3.3]
$\mathcal{C}_9$	[0 2.5]
$\mathcal{C}_{10}$	
$\mathcal{C}_{11}$	[0 0.9]
$\mathcal{C}_{12}$	
$\mathcal{C}_{13}$	

$\mathcal{C}_i$	$\alpha = 0.8$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.8 2.2]
$\mathcal{C}_2$	[1.8 2.2]
$\mathcal{C}_3$	[0 1.4]
$\mathcal{C}_4$	[0 0.4]
$\mathcal{C}_5$	[1.2 3.2]
$\mathcal{C}_6$	[1.8 3.2]
$\mathcal{C}_7$	[2.6 3.2]
$\mathcal{C}_8$	[1.4 3.2]
$\mathcal{C}_9$	[0 2]
$\mathcal{C}_{10}$	
$\mathcal{C}_{11}$	[0 0.6]
$\mathcal{C}_{12}$	
$\mathcal{C}_{13}$	

$\mathcal{C}_i$	$\alpha = 0.9$
$\mathcal{C}_0$	
$\mathcal{C}_1$	[1.9 2.1]
$\mathcal{C}_2$	[1.9 2.1]
$\mathcal{C}_3$	[0 1.2]
$\mathcal{C}_4$	[0 0.2]
$\mathcal{C}_5$	[1.6 3.1]
$\mathcal{C}_6$	[1.9 3.1]
$\mathcal{C}_7$	[2.8 3.1]
$\mathcal{C}_8$	[1.7 3.1]
$\mathcal{C}_9$	[0 1.5]
$\mathcal{C}_{10}$	
$\mathcal{C}_{11}$	[0 0.3]
$\mathcal{C}_{12}$	
$\mathcal{C}_{13}$	

$\mathcal{C}_i$	$\alpha = 0.5$	$\alpha = 0.4$
$\mathcal{C}_5$	[3.5 3.5]	[3.2 3.6]
$\mathcal{C}_9$	[0 0]	[0 0.4]
$\mathcal{C}_{14}$	[1.5 3.5]	[1.4 4]
$\mathcal{C}_{17}$	[1.5 1.5]	[1.4 1.8]

$\mathcal{C}_i$	$\alpha = 0.3$
$\mathcal{C}_5$	[2.9 3.7]
$\mathcal{C}_{10}$	[0 0.8]
$\mathcal{C}_{16}$	[1.3 4.5]
$\mathcal{C}_{19}$	[1.3 2.1]

$\mathcal{C}_i$	$\alpha = 0.2$
$\mathcal{C}_5$	[2.6 3.8]
$\mathcal{C}_{10}$	[0 1.2]
$\mathcal{C}_{16}$	[1.2 5]
$\mathcal{C}_{22}$	[1.2 2.4]

$\mathcal{C}_i$	$\alpha = 0.1$
$\mathcal{C}_5$	[2.3 3.9]
$\mathcal{C}_{10}$	[0 1.6]
$\mathcal{C}_{16}$	[1.1 5]
$\mathcal{C}_2$*	[1.1 2.7]

$\mathcal{C}_i$	α=0
$\mathcal{C}_5$	[2 4]
$\mathcal{C}_{10}$	[0 2]
$\mathcal{C}_{16}$	[1 5]
$\mathcal{C}_2$*	[1 3]

$\mathcal{C}_i$	$\alpha = 0.3$
$\mathcal{C}_{20}$	[1.3 1.4]
$\mathcal{C}_{13}$	
$\mathcal{C}_7$	[2.6 2.7]

$\mathcal{C}_i$	$\alpha = 0.2$
$\mathcal{C}_{23}$	[1.2 1.6]
$\mathcal{C}_{13}$	
$\mathcal{C}_7$	[2.4 2.8]
$\mathcal{C}_{17}$	[3.8 3.8]
$\mathcal{C}_{20}$	
$\mathcal{C}_{21}$	
$\mathcal{C}_{24}$	[1.2 1.2]
$\mathcal{C}_{25}$	[2.6 2.6]
$\mathcal{C}_{26}$	[1.2 1.2]
$\mathcal{C}_{27}$	[5 5]

$\mathcal{C}_i$	$\alpha = 0.1$
$\mathcal{C}_{22}$	[1.1 1.8]
$\mathcal{C}_{13}$	
$\mathcal{C}_7$	[2.2 2.9]
$\mathcal{C}_{17}$	[3.4 3.9]
$\mathcal{C}_{20}$	
$\mathcal{C}_{21}$	
$\mathcal{C}_{24}$	[1.1 1.6]
$\mathcal{C}_{25}$	[2.3 2.8]
$\mathcal{C}_{26}$	[1.1 1.6]
$\mathcal{C}_{27}$	[5 5]

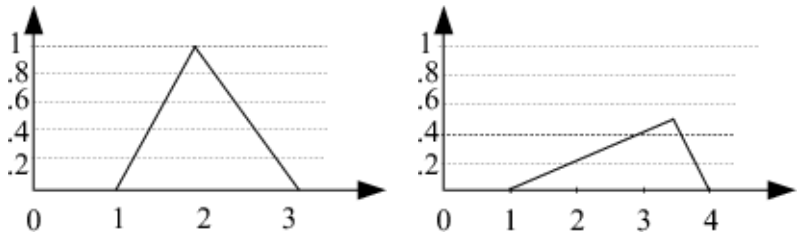
$\mathcal{C}_i$	α=0
$\mathcal{C}_{22}$	[1 2]
$\mathcal{C}_{13}$	
$\mathcal{C}_7$	[2 3]
$\mathcal{C}_{17}$	[3 4]
$\mathcal{C}_{20}$	
$\mathcal{C}_{21}$	
$\mathcal{C}_{23}$	[1 2]
$\mathcal{C}_{24}$	[2 3]
$\mathcal{C}_{25}$	[1 2]
$\mathcal{C}_{26}$	[5 5]

$\mathcal{C}_{F_i}$
$\mathcal{C}_{F_0}$
$\mathcal{C}_{F_1}$
$\mathcal{C}_{F_2}$
$\mathcal{C}_{F_3}$
$\mathcal{C}_{F_4}$
$\mathcal{C}_{F_5}$
$\mathcal{C}_{F_6}$
$\mathcal{C}_{F_7}$
$\mathcal{C}_{F_8}$
$\mathcal{C}_{F_9}$
$\mathcal{C}_{F_{10}}$
$\mathcal{C}_{F_{11}}$
$\mathcal{C}_{F_{12}}$
$\mathcal{C}_{F_{13}}$
$\mathcal{C}_{F_{14}}$
$\mathcal{C}_{F_{15}}$
$\mathcal{C}_{F_{16}}$
$\mathcal{C}_{F_{17}}$
$\mathcal{C}_{F_{18}}$
$\mathcal{C}_{F_{19}}$
$\mathcal{C}_{F_{20}}$
$\mathcal{C}_{F_{21}}$
$\mathcal{C}_{F_{22}}$
$\mathcal{C}_{F_{23}}$
$\mathcal{C}_{F_{24}}$
$\mathcal{C}_{F_{25}}$
$\mathcal{C}_{F_{26}}$
$\mathcal{C}_{F_{27}}$

M_i	Nc_i
$p3p6$	$C_{1,4}$
$p5p6$	$C_{4,3}$
$p6p7$	$C_{4,5}$
$p3p4$	$C_{1,2}$

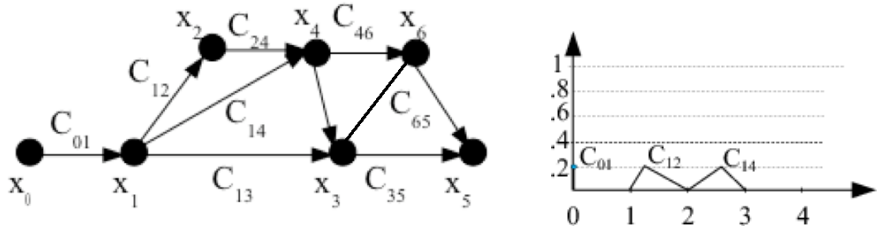
M_i	Nc_i
$p2p5$	$C_{1,3}$
$p4p7$	x_2
$p2p7$	$C_{1,5}$
$p5p8$	$C_{3,6}$
$p7p8$	x_6
$p5p6$	x_5
$p3p6$	$C_{4,3}$
$p3p4$	$C_{1,4}$
$p6p7$	$C_{1,2}$
$p2p5$	$C_{4,5}$

FIG. 11.17 – Classes emboîtées pour les G^α.



a) $C_{F_{13}}$ de $\mathcal{C}_{F_3}$ b) $C_{F_{14}}$ de $\mathcal{C}_{F_{14}}$

FIG. 11.18 – Contraintes floues associées à une classe.



a) b)

FIG. 11.19 – Séquence $s = t_1t_2t_4t_3t_6t_5$: a) STN, b) Contraintes floues C_{01}, C_{12} and C_{14}

3) $G^{0.2}$ et $G^{0.1}$. Mais deux cas intéressants apparaissent si G^{α_1} et G^{α} n'ont pas a même structure (voir figure 11.17) :

Cas 1 : pour $\alpha_1 > \alpha$, $[\mathcal{C}_j]^{\alpha_1} \subseteq [\mathcal{C}_k]^{\alpha}$ et $[\mathcal{C}_j]^{\alpha_1} \subseteq [\mathcal{C}_l]^{\alpha}$, i.e. une classe de G^{α_1} est contenue dans deux classes de G^{α}. C'est le cas pour G^1 and $G^{0.9}$: $[\mathcal{C}_7^+]^1$ est emboîtée dans les deux classes $[\mathcal{C}_7]^{0.9}$ et $[\mathcal{C}_9]^{0.9}$.

Cas 2 : pour $\alpha_1 > \alpha$, $[\mathcal{C}_j]^{\alpha_1} \subseteq [\mathcal{C}_k]^{\alpha}$ et $[\mathcal{C}_l]^{\alpha_1} \subseteq [\mathcal{C}_k]^{\alpha}$, i.e. deux classes G^{α_1} sont contenues dans une même classe de G^{α}. C'est le cas pour $G^{0.1}$ et G^0 : $[\mathcal{C}_2]^{0.1} \subseteq [\mathcal{C}_2*]^0$ et $[\mathcal{C}_{22}]^{0.1} \subseteq [\mathcal{C}_2*]^0$. Par ailleurs, la classe $[\mathcal{C}_{22}]^{0.1}$ est une classe restreinte de $[\mathcal{C}_2]^{0.1}$.

Considérons la séquence de transition $s = t_1; t_2; t_4; t_3; t_6; t_5$ in fig. 11.15.a, correspondant à la perte d'une réponse tardive. La classe atteinte est $\mathcal{C}_{F_{23}} = (p_7p_8, x_5)$ dans G_F ; cette classe n'existe que dans les graphes $G^{0.1}$ et $G^{0.2}$ (elle correspond à la classe $\mathcal{C}_{21}$. Donc la possibilité d'occurrence de ce comportement est de 0.2. Le réseau de contraintes de cette séquence est représenté sur la figure 11.19.a : nous avons besoin de connaître les contraintes $C_{i,j}$ entre chaque paire de noeuds (x_i, x_j) des réseaux de contraintes liés aux tir des transitions de la séquence s, à l'occurrence (les arcs indiqués entre parenthèses sont ceux de la figure 11.16.e.) :

- $Nt_{1,0} : C_{01}(x_0, x_1)$ (arc $a0$) ;
- $Nt_{2,1} : C_{12}(x1, x2)$ (arc $a29$) ;
- $Nt_{4,26} : C_{12}(x_1, x_2), C_{14}(x_1, x_4)$ et $C_{24}(x_2, x_4)$ (arc $a28$) ;
- $Nt_{3,25} : C_{14}(x_1, x_4), C_{13}(x_1, x_3)$ et $C_{43}(x_4, x_3)$ (arc $a27$) ;
- $Nt_{6,24} : C_{43}(x_4, x_3), C_{46}(x_4, x_6)$ et $C_{36}(x_3, x_6)$ (arc $a26$) ;
- $Nt_{5,17} : C_{36}(x_3, x_6), C_{35}(x_3, x_5)$ et $C_{65}(x_6, x_5)$ (arc $a20$).

Les contraintes $C_{0,1}$, $C_{1,2}$, $C_{1,3}$, $C_{1,4}$, $C_{2,4}$, $C_{3,5}$, $C_{4,6}$ et $C_{6,5}$ sont indiquées sur la table 11.2.

Le STN flou N_Fs est obtenu à partir de l'intersection des Nt pour chaque valeur d'α représenté à la table 11.2. Les contraintes floues (triangulaires) obtenues sont les suivantes : $C_{13} = [3\ 3.8\ 4]$, $C_{24} = [1\ 1.4\ 2]$, $C_{35} = [3.4\ 3.8\ 3.9]$, $C_{36} = [3\ 3.8\ 4]$, $C_{43} = [1\ 1.2\ 2]$, $C_{46} = [5\ 5\ 5]$, et $C_{65} = [0\ 0\ 1]$;

arc/α	C_{01}
$a_0/0:$	[0.0 0.0]
$a_0/.1:$	[0.0 0.0]
$a_0/.2:$	[0.0 0.0]

t_1

arc/α	C_{12}	C_{14}	C_{24}
$a_{27}/0:$	[1.0 2.0]	[2.0 3.0]	[1.0 2.0]
$a_{28}/.1:$	[1.1 1.6]	[2.3 2.8]	[1.2 1.7]
$a_{28}/.2:$	[1.2 1.2]	[2.6 2.6]	[1.4 1.4]

t_4

arc/α	C_{43}	C_{46}	C_{36}
$a_{25}/0:$	[1.0 2.0]	[5.0 5.0]	[3.0 4.0]
$a_{26}/.1:$	[1.1 1.6]	[5.0 5.0]	[3.4 3.9]
$a_{26}/.2:$	[1.2 1.2]	[5.0 5.0]	[3.8 3.8]

t_6

arc/α	C_{12}
$a_{28}/0:$	[1.0 2.0]
$a_{29}/.1:$	[1.1 1.6]
$a_{29}/.2:$	[1.2 1.2]

t_2

arc/α	C_{14}	C_{13}	C_{43}
$a_{26}/0:$	[2.0 3.0]	[3.0 4.0]	[1.0 2.0]
$a_{27}/.1:$	[2.3 2.8]	[3.4 3.9]	[1.1 1.6]
$a_{27}/.2:$	[2.6 2.6]	[3.8 3.8]	[1.2 1.2]

t_3

arc/α	C_{36}	C_{35}	C_{65}
$a_{21}/0:$	[3.0 4.0]	[3.4 3.9]	[0.0 1.0]
$a_{20}/.1:$	[3.4 3.9]	[3.4 3.9]	[0.0 0.5]
$a_{20}/.2:$	[3.8 3.8]	[3.8 3.8]	[0.0 0.0]

t_5

TAB. 11.2 – Les réseaux de contraintes Nt de la séquence s pour les graphes G^0, $G^{0.1}$ et $G^{0.2}$.

C_{01}, C_{12} et C_{14} sont représentées fig. 11.19.b. On peut voir que le noyau de ces contraintes sont vides et que la hauteur est de .2. Cela veut dire que la *possibilité* que la transition t_6 puisse être franchie est de .2 et donc la *nécessité* (certitude) est nulle.

Les travaux présentés dans ce chapitre sont les derniers que j'ai entrepris et sont encore en cours. Il est sans doute encore tôt pour que je puisse faire un bilan définitif de cet axe. En effet, il est au centre de mes perspectives que je présente maintenant.

Chapitre 12

Bilan et Perspectives

12.1 Bilan ou préambule à la perspective

L'outil est un objet qui prolonge la main de l'homme, cet organe non spécialisé qui peut faire tant de choses. Depuis les premiers outils, l'homme agit sur son milieu, le transforme, et se transforme aussi. De l'invention de la roue, on est arrivé à l'invention de l'automobile. Peut-être pas par une voie directe ; il y a certes les roues, mais sans compter le moteur, il a eu d'abord le tour du potier... et ensuite, les machines-outils qui fabriquent les pièces, et puis les robots. Et aujourd'hui, nous avons les systèmes embarqués : pas seulement dans les systèmes de transport (aéronautique, espace, ferroviaire, automobile, etc) ou dans domaine de l'énergie (nucléaire, chimie, etc), mais aussi dans les appareils personnels tels que les téléphones mobiles, les systèmes de paiement, les PALM, les ordinateurs de bord dans les véhicules et les terminaux TV interactifs. Nos outils – objets prolongeant un organe non spécialisé (la main) – sont eux mêmes aujourd'hui munis des systèmes non-spécialisés, parce que programmables et en plus, capables d'interagir avec l'environnement. Et donc, ces outils (embarqués) sont aussi un prolongement du cerveau... et en même temps ils permettent à la main d'interagir à distance.

L'histoire de la science est longue et intéressante ; dans ce parcours, nous avons eu récemment, la révolution industrielle et l'automatisation. Mais je veux cibler sur la numérisation, encore plus récente (à peine une cinquantaine d'années), et indiquer où ce sont placés mes travaux dans l'évolution générale de l'informatique et de l'automatique, et au sein des systèmes embarqués. En effet, chaque découverte ouvre de nouveaux domaines de recherche, et l'avancée dans chaque domaine fait en même temps apparaître le besoin de, à nouveau, interagir avec les domaines issus de la souche commune. C'est le cas par exemple, des systèmes embarqués, et en particulier des systèmes hybrides pour donner un exemple.

Qui dit numérisation, dit algèbre booléenne. En 1854, Georges Boole a publié *An investigation into the Laws of Thought, on Which are founded the Mathematical Theories of Logic and Probabilities*, avec une algèbre de la logique, l'algèbre booléenne. Cette algèbre a trouvé des applications dans la conception des circuits combinatoires et par la suite elle a été à la base de la révolution des ordinateurs. Utilisée par Shannon, on arrive aussi à l'automatique échantillonée[1]... La logique binaire a été critiquée par Lukasiewicz, qui a proposé (dans les années 20) une logique tri-valuée ; Zadeh a

[1]Le théorème de Shannon énonce que la fréquence d'échantillonnage d'un signal doit être égale ou supérieure au double de la fréquence maximale contenue dans ce signal, afin de convertir ce signal d'une forme analogique à une forme numérique. Ce théorème est à la base de la conversion numérique des signaux.

introduit en 1965 la logique floue et en 1978 la théorie des possibilités, qui permet de prendre en compte l'imprécision et l'incertitude.

Or la logique est à la fois un outil de compréhension et de raisonnement et un outil d'abstraction permettant d'avoir la vue qualitative sur laquelle s'est construite, en grande partie, l'automatique discrète (logique séquentielle). Cela a donné naissance à de nombreuses écoles de pensées et à de nombreux types de *logiques formelles*, comme les logiques temporelles, la logique linéaire de Girard ou la logique floue. Recombiner diverses visions de la logique pour mieux appréhender la complexité des systèmes embarqués tout au long du cycle de vie (spécification, vérification et mise en œuvre est à la base de mes préoccupations.

Bien que les premiers ordinateurs existaient déjà au milieu des années 50, ce sont plutôt les automates de Mealy et Moore qui ont permis de concevoir les circuits séquentiels pour la commande utilisant l'électronique numérique. La miniaturisation des puces a permis une très large intégration et l'électronique numérique, appelée aujourd'hui *électronique* tout simplement, a envahi dans les années 70 des domaines tels que l'électronique de puissance, remplacé les vannes électromécaniques, etc. La conception et la vérification des circuits séquentiels à cédé sa place, dans la plupart des cas, à la conception de programmes qui tournent sur des micro-ordinateurs, devenus à bas prix depuis les années 80.

L'automatique linéaire des années 60 a intégré dans les années 70 la commande numérique d'abord par la théorie des systèmes échantillonnés (en utilisant par ailleurs la théorie de Shannon). En parallèle, l'informatique s'intéressait alors à valider les programmes, et dans les cas de la commande, en faisant abstraction de la partie continue du système. L'automatique a introduit alors dans les années 80 l'étude des systèmes à événements discrets (SED) – mon domaine d'application – , pour traiter ces systèmes où les événements venaient en général de la commande numérique générée par des ordinateurs ou des automates programmables. Beaucoup de ces travaux se sont inspirés de recherches en informatique, mais aussi recherche opérationnelle. Des termes comme *productique* ont vu le jour. Le système à contrôler, même s'il était continu, était aussi représenté (de manière qualitative) par un SED de façon à valider le système global (commande + procédé) par la simulation par exemple. Un des modèles utilisés étaient les réseaux de Petri ; par ailleurs, à la fin des années la norme industrielle Grafcet a vu le jour, avec beaucoup de similitudes avec les réseaux de Petri. Dans le cadre de ma trajectoire professionnelle, les travaux de F. Assis, par exemple, qui utilisait les réseaux de Petri pour la coordination (chapitre 9), se placent exactement dans ce cadre.

Les systèmes devaient aussi être intelligents (on peut convenir que derrière ce terme un peu racoleur on trouve simplement le fait que des techniques issues de l'intelligence artificielle (IA) sont utilisées). La complexité des systèmes est souvent traitée par une décomposition hiérarchique ; les niveaux les plus hauts, comme la planification ou l'ordonnancement font alors appel à l'IA. C'est pourquoi, dans mes travaux au niveau de la coordination, je me suis efforcée de à définir des ponts avec l'IA. Du point de vue de la surveillance, j'ai combiné la théorie des possibilités aux réseaux de Petri pour pouvoir décrire un état du système qui est mal connu (chapitre 10). L. Caimi a implementé un simulateur de réseaux de Petri avec marquage flou en utilisant, par ailleurs, un système expert à base de frames (section 10.2). Néanmoins cette traduction n'est pas naturelle, et je me suis intéressée à la logique linéaire pour représenter ces réseaux de Petri. Il avait déjà été établi que les réseaux de Petri (ordinaires) pouvaient être une sémantique pour cette logique. Mes travaux avec B. Chezalviel et R. Valette ont montré que la logique linéaire permettait aussi de représenter les réseaux de Petri avec marquage imprécis (section 10.3).

J'ai eu deux grands moments charnières dans ma vie professionnelle. Le premier c'est mon intégration en 1999 au département d'Informatique de l'Université des Sciences Sociales et à l'équipe Systèmes à Objets Coopératifs (SOC) de l'Institut de Recherches en Informatique de Toulouse (IRIT). J'ai découvert les systèmes d'information d'abord par le biais de l'enseignement. Au niveau recherche, les problématiques de la modélisation et de la vérification ont été prépondérantes. C'est dans ce contexte que j'ai co-encadré la thèse d'Omar Tahir sur une sémantique pour les diagrammes de séquence UML (chapitre 8) dans le cadre de Féria (Fédération de Recherche en Informatique et Automatique). Dans les thèses en co-tutelle de L. Bolan et E. Pozzebon (section 9.3), j'ai appliqué les réseaux de Petri à objets pour spécifier un outil pour la conception de cours dans un système tuteur intelligent (un domaine d'application totalement différent donc de celui de F. Assis et M. Braga, décrits dans la section 9.2) basé sur une architecture multi-agent. Des grands défis existent dans ces systèmes tels que trouver un modèle de l'étudiant qui le représente le mieux possible dans le contexte de l'apprentissage et la compréhension du langage naturel lors d'une interaction avec correction d'un exercice. Cependant, les réseaux de Petri à objets permettent d'exprimer des politiques d'interaction entre les agents, et de modéliser de façon élégante l'intégration des modèles pédagogique et de l'étudiant en vue d'une interaction adaptative entre le système tuteur et l'étudiant.

L'équipe SOC faisait partie à cette époque[2] du thème Sûreté du Développement du Logiciel. C'est pourquoi une partie des mes travaux, notamment ceux avec S. Cousy (réseaux de Petri temporels flous), X. Mao et A. Hamdani (réseaux de Petri temporels), étaient d'avantage liés à ce thème, en coopération avec le LAAS dans le cadre de FÉRIA. Dans ces travaux (chapitre 11), il y avait toujours une abstraction de la partie continue du procédé. La fin de l'activité, même si elle était dictée par un système décrit par des équations différentielles, est vue comme un événement pouvant avoir lieu dans un intervalle de temps (flou) donné. J'avais fait un premier pas vers l'analyse formelle de ces réseaux par une extension du graphe de classes (proposé par B. Berthomieu), réalisée par S. Cousy (section 11.2). Une comparaison avec les réseaux de Petri stochastiques a aussi été faite. Cependant, le souhait de pouvoir définir de façon quantitative les contraintes temporelles que doivent vérifier les dates de franchissement d'une séquence de transitions m'a amené à définir un nouveau graphe de classes. La mise en oeuvre d'un outil informatisée, GraphC, pour la génération de ce graphe pour les réseaux de Petri temporels, cas particulier des réseaux de Petri flous, a été réalisé par X. Mao et amélioré par A. Hamdani (section 11.3). Une première extension de GraphC pour les réseaux de Petri flous a été faite en utilisant la notion d'α-coupe (section 11.4).

Le deuxième moment charnière c'est mon intégration en octobre 2006 à Supaero dans le poste de professeur en Systèmes Embarqués. Mon intérêt pour ce poste est en parfaite cohérence avec mes motivations professionnelles et ma spécialité. En effet, mon activité de recherche est centrée sur les systèmes à événements discrets et les réseaux de Petri, des modèles et des outils essentiels pour les systèmes embarqués. En plus, les systèmes aéronautiques et spatiaux sont au coeur du domaine des systèmes embarqués, et Supaéro est partie prenante du pôle de compétitivité "Aéronautique, Espace et Systèmes embarqués" . Travailler à Supaéro a eu pour conséquence de rendre mes activités d'enseignement plus proches et plus en cohérence avec mes activités de recherche. Je suis actuellement responsable du mastère spécialisé "Systèmes Embarqués"[3], en partenariat avec l'ENSEEIHT. Mon implication dans l'enseignement du domaine "Systèmes Embarqués" de la troisième année à Supaéro, en particulier dans la création d'une discipline sur les Systèmes Hybrides, ma permis de

[2]Aujourd'hui, et depuis 2006, l'équipe SOC fait partie du thème "Interaction, Autonomie, Dialogue et Coopération" de l'IRIT.

[3]Accrédité par la CGE à partir de septembre 2007.
`http ://www.supaero.fr/fr/formations/masteres-specialises/systemes-embarques.html`

rejoindre ce mouvement récent de l'automatique. En effet, depuis les années 90, la communauté qui étudie les systèmes hybrides essaie de construire une théorie des systèmes basée sur les avancées dans la commande des SED et celle des systèmes continus (ou de la partie continue des systèmes). Il faut donc prendre en compte conjointement les parties continue et discrète du système dynamique. J'ai amorcé un travail exploratoire de ce domaine en encadrant le stage de mastère spécialisé en systèmes embarqués de Z. Saharaoui [Sah07].

Dans le chapitre 7, j'avais posé la question : *dans quels domaines d'application la théorie des réseaux de Petri est-elle la plus prometteuse et quels sont les fondements logiques de cette théorie ?* Répondre que son application est prometteuse dans les systèmes qui possèdent une partie importante dictée par des événements discrets peut paraître évidente. J'ai essayé de montrer que c'était le cas dans tous les travaux que j'ai menés. Aujourd'hui je souhaite comparer non seulement le pouvoir de modélisation et vérification des réseaux de Petri, mais aussi comparer le pouvoir algorithmique des solutions proposées, avec celles proposées par d'autres théories dans les différents domaines. À ma connaissance, un premier pas a été fait dans [zBer05] où l'expressivité des réseaux de Petri temporels est comparée à celle des automates temporisés par rapport à la bissimilarité temporelle. Il est important aussi de montrer si cette théorie permet de mener de façon efficace la résolution des problèmes liés à la vérification. En effet, souvent la question revient : pourquoi utiliser les réseaux de Petri, si le pouvoir d'expression est le même que celui d'autres outils ? Je trouve que les réseaux de Petri ont au moins un avantage, par exemple, par rapport aux automates, c'est le fait qu'ils permettent de faire la différence entre conflit et parallélisme. Comme l'a explicité E. Villani dans [Vil04], si l'on veut aborder un système complexe avec une vision *objet*, il est nécessaire d'être capable de travailler dans le cadre d'une sémantique avec parallélisme vrai pour laisser une grande autonomie aux objets.

12.2 Perspectives

Il y a certainement dans mes travaux une grande diversité des domaines et des outils. Cela est d'une grande richesse, mais aussi source de dispersion. Dans les cinq prochaines années, et dans le cadre de ma mission à Supaéro, je compte continuer à travailler dans le domaine de l'Informatique mais aussi renforcer les liens avec l'automatique discrète et la supervision par le biais des systèmes hybrides. C'est pourquoi je compte développer plus particulièrement les points suivants, en cohérence avec la plus grande partie de mes travaux antérieurs.

Dans le cadre de la vérification de systèmes avec des informations mal-connues, il s'agira de continuer à développer l'outil GraphC, qui n'est pas encore complètement fonctionnel, en particulier pour les transitions non-bornées. Il faut utiliser une structure de données plus performante que celle utilisée maintenant tout en gardant, au niveau de la représentation des contraintes temporelles du réseau de Petri, les réseaux de contraintes temporelles (STN). Je trouve que d'un point de vue didactique cette représentation est plus claire (plus explicative) que la représentation graphique des régions d'horloge (sur le plan XY). Je compte aussi mettre en oeuvre l'approche proposée dans [Car06b] pour l'extension du graphe de classes quand l'intervalle temporel est flou qui utilise les α-coupes en développant un outil informatique et en traitant des exemples. Une autre approche possible est celle d'utiliser directement dans la génération du graphe de classes flou les réseaux de contraintes flous. Des travaux existent sur ce sujet, par exemple ceux de H. Fargier de l'IRIT (Toulouse). Une nouvelle collaboration pourrait s'établir.

Plus généralement, l'analyse des réseaux de Petri temporels flous est à affiner selon deux points

de vue que je considère complémentaires : la vérification de modèles et la preuve de théorèmes. Les travaux sur un graphe de classes d'état implementé par l'outil GraphC sont dans la ligne de la vérification de programmes, où le raisonnement est basé sur la notion de chroniques. Toutefois le graphe de classes d'états GraphC, au lieu de considérer seulement une chronique, considère toutes les chroniques possibles ; la sémantique forte est préservée et nous avons un ordre total. Outre le problème de l'explosion combinatoire, si le marquage initial du réseau de Petri change, il est nécessaire de recalculer le graphe de classes. Il serait donc intéressant d'utiliser l'approche preuve de théorèmes et d'exploiter la structure du réseau de Petri en utilisant des résultats obtenus avec la logique linéaire (qui traite la notion de ressource en logique). L'avantage principal est de se placer dans le cadre du parallélisme vrai à la place du parallélisme par entrelacement et de travailler sur un ordre partiel pouvant représenter un très grand nombre de séquences. La sémantique opérationnelle des "pas couvrants" (proposé par F. Vernadat du LAAS) pourrait être également utilisée pour permettre de mettre en exergue un ensemble de transitions franchies en parallèle (le pas), permettant ainsi de réinjecter du parallélisme dans l'approche "graphe de classe", comme elle l'a été dans l'approche *énumération des marquages* pour un réseau de Petri ordinaire. Cela permettrait de faire un compromis entre l'approche *vérification* (impliquant l'exhaustivité) et l'approche *preuve* (sur un seule chronique). En particulier, la prise en compte de la structure du réseau peut permettre de déterminer les valeurs d'α-cut à utiliser. Les résultats seront appliqués pour évaluer la fiabilité opérationnelle d'un aéronef. Ce sujet sera développé dans le cadre d'une thèse de doctorat.

Enfin, le dernier axe que je compte privilégier concerne le passage des systèmes avec des contraintes temporelles aux systèmes hybrides. La vérification des SED est essentielle dans les systèmes embarqués, mais la composante hybride y joue un rôle très important, notamment dans le cas où il y a une interaction avec le monde physique, comme c'est le cas pour les systèmes aéronautiques. Comme je l'ai déjà dit, je travaille actuellement à Supaéro, et je souhaite renforcer les liens avec l'automatique. Je pense que les systèmes hybrides sont un domaine de recherche particulièrement propice à ce lien. Mes travaux sur les réseaux de Petri temporels fournissent une bonne base pour amorcer l'étude de ces systèmes. Ayant fait, jusqu'à présent, une abstraction de la dynamique continue sous la forme d'un intervalle temporel, il est naturel de vouloir regarder plus en détail l'interaction entre les dynamiques discrète et continue du système hybride analysé en tant que tel. Il me faut commencer par une comparaison entre modèles déjà existants, tels que les automates hybrides et les réseaux de Petri hybrides, en particulier les réseaux de Petri prédicat-transition différentiels, développés initialement au Laas et par la suite à l'ENSIACET par Jean Marc Le Lann et G. Hetreux [zPer03]. À moyen terme, je compte travailler sur la modélisation et à la vérification des systèmes hybrides. À long terme, les compétences à l'ONERA-Toulouse sur la simulation distribuée avec l'architecture HLA - *High Level Architecture*[4], peuvent permettre aussi d'envisager une simulation distribuée d'un système hybride (distribué). En effet, du à la complexité de tels systèmes, la simulation est, en générale, actuellement, centralisée.

[4]L'Architecture de Haut Niveau est une spécification d'architecture logicielle qui définit comment créer une simulation globale composée de simulations distribuées interagissant sans être recodées. Dans HLA, chaque simulation participante est appelée fédéré ; elle interagit avec d'autre fédérés au sein d'une fédération HLA, qui est en fait un groupe de fédérés. C'est un standard IEEE pour la simulation distribuée à événements discrets.

Bibliographie

Mes publications :

[Cou04] Sébastien Cousy, *Les Réseaux de Petri et le traitement de l'imperfection de l'information.* Projet de fin d'études ENSEEIHT et DEA Systèmes Automatiques. Co-encadré avec G. Juanole, LAAS, septembre 2004.

[Bit98] G. Bittencourt, J. Cardoso, L.L. Caimi : A Frame-Based Representation for Fuzzy Petri Net. IEEE WCCI'98, Anchorage, Alaska, 4-9 mai, 1998.

[Bra93a] M.C.L.F. Braga, *Um gerente de célula flexível de manufatura (Un gestionnaire de cellule flexible de production)* (en portugais). Thèse de *mestrado* (équivalent M2R), Brésil, septembre 1993.

[Bra93b] M.C.L.F. Braga, C.A. Martin, J. Cardoso : Une expérience d'utilisation des réseaux de Petri pour le développement de logiciel pour la gestion d'une cellule flexible(en portugais). 12^o *Congresso Brasileiro de Engenharia Mecânica,* Brasília, Brésil, p. 425-428, décembre 1993.

[Cai98] L.L. Caimi, *Simulateur de réseaux de Petri avec marquage flou* (en portugais). Thèse de *mestrado* (équivalent M2R), Brésil, février 1998.

[Cai98a] L.L. Caimi, J. Cardoso, G. Bittencourt : A Rule-based tool for Petri net management. CESA'98, Tunisie, 1-4 avril 1998, p. 589-594.

[Cai98b] J. Cardoso, G. Bittencourt, L.L. Caimi : A rule-based tool for Fuzzy Petri net simulation. IPMU'98, Paris 6-10 juillet 1998.

[Car90a] J. Cardoso. *Sur les réseaux de Petri à marquages flous.* PhD thesis, Université Paul Sabatier/Laboratoire d'Architecture et d'Analyse de Systèmes, Toulouse, France, October 1990.

[Car90b] J. Cardoso, R. Valette, D. Dubois : Petri nets with incertain markings. In *Lecture Notes in Computer Science*, G. Rozenberg (Ed), Springer Verlag, 483 :64-78, 1990. Présenté aussi à *10th Int. Conf. on Applications and Theory of Petri nets*, p. 35-51, Bonn, Germany, juin 1989.

[Car90c] J. Cardoso, R. Valette, D. Dubois : Réseaux de Petri avec marquages imprécis (en portugais). 8^o *Congresso Brasileiro de Automática, CBA 90-IFAC*, p. 578-584, Belém, Brésil, 10-14 Septembre 1990.

[Car92] J. Cardoso, J.M. Farines, J.E.R. Cury : Un système de coordination pour des processus de fabrication basé sur le modèle réseaux de Petri (en portugais). 9^o *Congresso Brasileiro de Automática CBA 92-IFAC*, p. 1066-1071, Vitória-ES, Brésil, 14-18 septembre 1992.

[Car93] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Fuzzy Petri nets and linear logic. *1993 IEEE Int. Conf. on Systems, Man and Cybernetics*, 2 :258-263, Le Touquet, 17-20 October 1993.

[Car94] J. Cardoso, G. Bittencourt, A. Castilho : Rule-Based Simulation of Petri nets with Imprecise Markings, *Brasil-Japan Joint Symposium on Fuzzy Systems*, p. 233-238, Campinas-Brésil, 20-22 Juillet 1994.

[Car95a] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Linear logic for imprecise firings in Fuzzy Petri nets, In *Fuzzy logic and soft computing*, B. Bouchon-Meunier, L. Zadeh and R. Yager (Eds), World Scientific, p. 119-128, 1995. *Papier sélectionné et re-évalué d'après la présentation au* 5^{th} IPMU - Information Processing and Management of Uncertainty in Knowledge-based Systems, p. 1269-1274, Paris-France, 4-8 July 1994

[Car95b] J. Cardoso, L.A. Künzle, R. Valette : Petri net based reasoning for the diagnosis of dynamic discrete event systems. *VI Int. Fuzzy Systems Association World Congress*, São Paulo, Brazil, July 22-28, 1995.

[Car96] J. Cardoso, R. Valette, D. Dubois. Fuzzy Petri Net : an overview, *13th IFAC World Congress*, San Francisco, EUA, 30 June-5 July 1996, p. 443-448.

[Car96a] J. Cardoso, R. Valette, B. Pradin-Chézalviel : Handling Uncertainty and Resources : Petri nets, Fuzzy and Linear Logic. *Journal of the Interest Group in Pure and Applied Logics (IGPL)* 1996, p. 491-493, ISBN 0945-9103. *Papier sélectionné et re-évalué d'après la présentation au* 3^{rd} rd Workshop on Logic, Language, Information and Computation - WoLLIC'96, Salvador, May 8-10, 1996.

[Car97] J. Cardoso, B. Pradin-Chézalviel : Logic and Fuzzy Petri nets. *Invited speaker* in *2nd Int. Workshop on Manufacturing and Petri nets*, Toulouse, France, June 23 1997, pp 17-34.

[Car98] J. Cardoso : Time Fuzzy Petri nets. *Fuzziness in Petri nets*, J. Cardoso and H. Camargo (Ed), Studies in Fuzziness, Physica Verlag, pp 115-145, 1998.

[Car99] J. Cardoso, R. Valette, D. Dubois : Possibilistic Petri nets. *IEEE Trans. on System, Man and Cybernetics, vol. 29, part B : Cybernetics, nº 5, Oct. 1999.*

[Car01a] J. Cardoso, C. Sibertin-Blanc, C. Soulé-Dupuy. Une sémantique formelle des diagrammes d'interaction d'UML via les réseaux de Petri, In Modélisation de Systèmes Réactifs (Actes MSR 2001, Toulouse, France), pg 497-512, ISBN 2-7462-0329-4, Hermès.

[Car01c] J. Cardoso, C. Sibertin-Blanc : Ordering actions in sequence diagrams of UML, ITI 2001 Proceedings of the 23rd Int. Conf. on Information Technology Interfaces, Pula, Croatia, June 19-22, 2001, pg 3-14.

[Car02] J. Cardoso, C. Sibertin Blanc. An operational semantics for UML interaction : sequencing of actions and local control, APII-JESA 36/2002. Reactive Systems, pg 1015-1028, ISBN 2-7462-0573-4.

[Car04a] J. Cardoso, G. Bittencourt, L. B. Frigo, E. Pozzebon, A. Postal. MathTutor : A Multi-Agent Intelligent Tutoring System, 1st IFIP Int. Conf. on Artificial Intelligence Applications and Innovations - AIAI 2004, Toulouse, 23- 25 August 2004, Kluwer Academic Publishers, 2004. v. 1. p. 231-242. ISSN/ISBN : 1-4020-8150-2.

[Car04b] J. Cardoso, G. Bittencourt, L. B. Frigo et al. Petri Nets for Authoring Mechanism. In : XV SBIE - Simpósio Brasileiro de Informática na Educação, 2004, Manaus - AM. Anais do XV SBIE 2004, v. 1. p. 378-387. ISSN/ISBN : 85-7401-161-4.

[Car05a] J. Cardoso, S. Cousy, G.Juanole : Extending Time Petri Nets to Fuzzy Time Petri Nets : Definition of the Graph of Fuzzy State Class, 16th IFAC World Congress, Juillet 2005, Prague, République Tchèque.

[Car05b] J. Cardoso, X. Mao, R. Valette : A graph of classes preserving quantitative temporal constraints considering unbounded transitions, 7th Int. Workshop on Performability Modeling of Computer and Communication Systems (PMCCS), Turin (Italie), 23-24 Septembre 2005.

[Car05c] J. Cardoso, R. Valette, X. Mao : Un nouveau graphe de classes pour la préservation des contraintes temporelles quantitatives, MSR 2005, Modélisation des sytèmes réactifs, Grenoble (Autrans), 5-7 octobre 2005, JESA Vol.39 n.1-2-3/2005, Hermès, pp. 191-206.

[Car06a] J. Cardoso, R. Valette. Un graphe de classes pour les réseaux dePetri p-temporels. Journées FAC'06 - Formalisation des Activités Concurrentes. 14 pp.

[Car06b] J. Cardoso, X. Mao, R. Valette. State Class Graph for Fuzzy Time Petri Nets, ESM'2006, European Simulation and Modelling Conf., October 23-25, 2006, Toulouse, France.

[Cav94b] C. Cavalcante, J. Cardoso, J.J.G. Ramos, O.R. Neves : Application of Fuzzy Control to Helicopter Navigation, *Brasil-Japan Joint Symposium on Fuzzy Systems*, p. 72-76, Campinas-Brésil, 20-22 July 1994.

[Cav94a] C. Cavalcante, *Système de navigation pour hélicoptère sans pilote utilisant la commande floue)* (en portugais). Thèse de *mestrado*, Brésil, juillet 1994.

[Cav95] C. Cavalcante, J. Cardoso, J.J.G. Ramos, O.R. Neves : Design and Tuning of a Helicopter Fuzzy Controller, *Fourth IEEE Int. Conf. on Fuzzy Systems and the Second Int. Fuzzy Engineering Symposium FUZZ-IFES 95*, Yokohama, Japan, March 20-24, 1995, p. 1549-1554.

[Cur92] J.E.R. Cury, J.M. Farines, J. Cardoso : An FMS Coordination System : an approach based on high-level Petri net. *Int. Symposium on Robotics, Mechatronics and Manufacturing Systems - RM2S'92*, Kobe, Japan, 16-20 September 1992.

[Far92] J.M. Farines, J. Cardoso, J.E.R. Cury : Specification and implementation of an FMS Coordination System Based on high-level Petri net. *IFAC Workshop on Intelligent Manufacturing Systems - IMS'92*, p. 67-71, Dearborn-USA, 1-2 October 1992. Edited by R.P. Judd and N. A. Kheir, Pergamon Press.

[Far93] J.M. Farines, J.E.R. Cury, J. Cardoso : Un environnement d'outils intégrés pour la conception de systèmes à événements discrets (en portugais). *Workshop Nacional de Projetos Cooperativos de Informática*, Itaipava-RJ, Brésil, 15-17 décembre 1993.

[Fri05a] L. Frigo, J. Cardoso, G. Bittencourt. Adaptive Interaction in Intelligent Tutoring Systems. CIAH'2005, Int. Workshop on Combining Intelligent and Adaptive Hypermedia Methods/Techniques in Web-based Education Systems, Salzburg-Autriche, 4 septembre 9 septembre 2005. Ioannis Hatzilygeroudis (Editor), p. 33-38.

[Fri05b] L. Frigo, J. Cardoso, G. Bittencourt. A Method for Modeling Adaptive Interactions in Intelligent Tutoring Systems. IJCEELL Special Issue on "Integrating Intelligent and Adaptive Hypermedia Techniques in Web-Based Education Systems", 10 p., Inderscience Publishers, I. Hatzilygeroudis (Editor).

[Fri07] L. Bolan-Frigo. *Un outil de conception de cours adaptatifs pour les systèmes tuteurs intelligents*, 8 janvier 2007, thèse de doctorat en Informatique, cotutelle Université Toulouse 1/IRIT et Université Fédérale de Santa Catarina, Brésil.

[Ham06] Abdia Hamdani, *Un graphe de classes pour les réseaux de Petri p-temporels.* M2R SAID, parcours Systèmes Industriels. Co-encadré avec R. Valette (LAAS), juin 2006.

[Kun96a] L. A. Kunzle, B. Pradin-Chézalviel, R. Valette, J. Cardoso ; Raisonnement temporel et diagnostique dans les systèmes parallèles (en portugais). 11^o CBA, setembro, 1996, São Paulo, Brasil, p. 435-440.

[Kun96b] L. A. Künzle, B. Pradin-Chézalviel, R. Valette, J. Cardoso ; "Raisonnement Temporel pour des Systemes Paralleles en vue du Diagnostic". Colloque AGI'96, Tour, France, p. 259-262, 2-3 juin 1996.

[Mao05a] X. Mao, J. Cardoso, R. Valette : A New Graph of Classes for the Preservation of Quantitative Temporal Constraints Automated Technology for Verification and Analysis : Third Int. Symposium, ATVA 2005, Taipei, Taiwan, October 4-7, 2005, Lecture Notes in Computer Science, Volume 3707 / 2005, pp.278-292 Springer-Verlag.

[Mao05] Xiaoyu Mao, *L'outil GraphC : un graphe de classes pour les réseaux de Petri t-temporels préservant les contraintes temporelles quantitatives*. Projet de fin d'études INSAT et M2R SAID Systèmes Automatiques, Informatiques et Décisionnels, parcours SRIC - Systèmes et Réseaux Informatiques Critiques. Co-encadré avec R. Valette (LAAS), septembre 2005.

[Pos04] Adriana Postal, *Un outil d'auteur pour le système tuteur intelligent MathTutor* (en portugais). Thèse de *mestrado* (équivalent M2R) co-encadrée avec M. Bittencourt, Florianópolis, Brésil, avril 2004.

[Poz06] E. Pozzebon, J. Cardoso, G. Bittencourt, C. Hanachi. A Multi-Agent Architecture for Group Learning, IADIS Int. Conf. e-Society, V1, ISBN : 972-8924-16-X, pg 60-67, July 2006, Dublin, Ireland.

[Poz07] Eliane Pozzebon. *Un outil de conception de cours adaptatifs pour les systèmes tuteurs intelligents*, 8 janvier 2007, thèse de doctorat en Informatique, cotutelle Université Toulouse 1/IRIT et Université Fédérale de Santa Catarina, Brésil.

[Poz07] E. Pozzebon, J. Cardoso, G. Bittencourt, C. Hanachi. A Group Learning Management Method for Intelligent Tutoring Systems. In : Special Issue of the International Journal of Computing and Informatics, M. McPherson and P. Isaías Ed., à paraître (2007).

[Sah07] Z. Saharoui. *Modélisation et simulation de systèmes hybrides*, projet de fin d'études du mastère Commande et Systèmes embarqués, Supaéro, en cours (soutenance prévue septembre 2007).

[San93] S.A. Sandri, J. Cardoso : On necessity-valued Petri nets,. 5^{th} *Int. Fuzzy Systems Association World Congress - IFSA 93*, p. 1338-1341, Seoul-Korea, 4-9 July 1993.

[San97] S.A. Sandri, J. Cardoso : Management of incomplete information in the processing of safe Petri nets with fuzzy durations. 7^{th} *Int. Fuzzy Systems Association World Congress - IFSA 97*, Prague, Rep. Tchèque, 25-29 June 1997, p. 300-305.

[San98] S. Sandri, J. Cardoso : Possibilistic timed Petri nets. IEEE WCCI'98, Anchorage, Alaska, 4-9 mai, 1998. Fuzzy Systems Proc., 1998, V1, pg 89-94. ISBN : 0-7803-4863-X.

[San99] S. Sandri, J. Cardoso : On Possibilistic timed safe Petri nets. *Threads in Fuzzy Petri nets research, Int. Journal of Intelligent Systems*, vol. 14, n^o 7, 1999.

[Sib00] C. Sibertin-Blanc, C. Hanachi , J. Cardoso. Protocols as First-class Components of Multiagent Systems. In Fourth Intern. Conf. on Multi-Agent Systems, ICMAS 2000, S. Kraus, H. Nakashima & M. Tambe (Eds.), 10-12 July 2000, Boston (MA), USA, IEEE Computer Society Press, p. 21-22 (poster).

[Sib01b] C. Sibertin-Blanc, J. Cardoso, C. Hanachi. Les protocoles comme composants à part entière des systèmes coopératifs. RERIR n. 11 (Revue Electronique sur les Réseaux et l'Informatique Répartie, ISSN 1262-3261), mars 2001 (article présenté à NOTERE'2000 3ème Colloque Int. sur les NOuvelles TEchnologies de la REpartition, I. Demeure & E. Najm (Eds.), Paris, 21 - 24 Novembre 2000 et sélectionné pour publication dans la revue RERIR).

[Sib01c] C. Sibertin-Blanc, J. Cardoso, C. Hanachi. Les protocoles comme composants à part entière des systèmes coopératifs. Actes de NOTERE '2000, 3ème Colloque International sur les NOuvelles TEchnologies de la REpartition, I. Demeure & E. Najm (Eds.), Paris, 21 - 24 Novembre 2000. Repris dans [Sib01b].

[Sib01a] C. Sibertin-Blanc, J. Cardoso, C. Hanachi. La spécification de protocoles d'interaction par réseaux de Petri. Dans : Journées francophones pour l'intelligence artificielle distribuée et les systèmes multi-agents JFIADSMA'01, A. El Fallah Seghrouchni & L. Magnin (Eds), Montréal, Canada, 12-14.11.2001, Hermes, p. 121-147, novembre 2001.

[Sib05] C. Sibertin-Blanc, O. Tahir, J. Cardoso. Interpretation of UML Sequence Diagrams as Causality Flows. In Advanced Distributed Systems, 5th Int. School and Symposium (ISSASD'05), LNCS 3563, pp. 126-140, Guadalajara, Mexique, janvier 2005.

[Soa94a] F.A. Soares, J. Cardoso, J.E. Cury : An integrated environment of tools for the design of manufacturing systems. *Int. Workshop on Intelligent Manufacturing Systems - IMS'94*, p. 489-494, Vienna-Austria, 13-15 June 1994.

[Soa94b] F. A. Soares, *Un environnement intégré d'outils pour la conception de systèmes à événements discrets* (en portugais). Thèse de *mestrado* (équivalent M2R), Brésil, juillet 1994.

[Tah03a] O. Tahir, J. Cardoso, C. Sibertin-Blanc. Une nouvelle approche d'intégration de diagrammes de séquence UML basée sur les RdP de haut niveau. Journées FAC'03, Toulouse, France, 12 et 13 Mars 2003, Rapport Irit 2003-03-R.

[Tah03b] O. Tahir, J. Cardoso, C. Sibertin-Blanc. Génération automatique de Diagrammes d'États-Transitions à partir de Diagrammes de Séquence UML : Une approche basée sur la sémantique des Réseaux de Petri. MSR'03 Colloque Francophone sur la Modélisation des Systèmes Réactifs, Metz, France, 6-8 octobre 2003.

[Tah05b] O. Tahir, C. Sibertin-Blanc and J. Cardoso. A Causality-Based Semantics for UML Sequence Diagrams. In Peter Kokol editor, Proceedings of the 23rd IASTED Int. Conf. on Software Engineering, Acta Press, pp. 106-111, Innsbruck, Austria, 2005.

[Tah06] O. Tahir. *Étude des diagrammes de la dynamique UML par les réseaux de Petri*, 8 décembre 2006, thèse de doctorat en Informatique, Université Toulouse 1/IRIT.

[Val88] R. Valette, D. Dubois, J. Cardoso : Représentation de l'état d'un atelier de fabrication avec prise en compte des incidents. *Congrès Automatique AFCET 1988 : Quelle Automatique dans les Industries Manufacturières*, p. 95-104, Grenoble-France, 10-12 octobre 1988.

[Val89a] R. Valette, J. Cardoso, D. Dubois : Monitoring Manufacturing Systems by means of Petri Nets with Imprecise Markings. *IEEE Int. Symposium on Intelligent Control*, p. 233-238, Albany N.Y., U.S.A., 25-26 September 1989.

[Val89b] R. Valette, J. Cardoso, H. Atabakhche, M. Courvoisier, T. Lemaire : Petri nets and production rules for decision levels in FMS control. In *Artificial Intelligence in Scientific Computation : towards second generation systems*, R. Huber et al. (Eds), JC Baltzer AG, Scientific Publishing Co (IMACS), p. 301-305, 1989.

[Val94] R. Valette, D. Andreu, J. Cardoso, J.C. Pascal : Fuzzy Petri nets and their application in CIME, *Special Issue of Recent Applications of Petri Nets of the Transactions of the Institute of Electrical Engineers of Japan*, Vol. 114-C, N^o 9, p. 876-880, 1994.

[Val96] R. Valette, F. Girault, L. A. Künzle, B. Pradin-Chézalviel, J. Cardoso : Vérification de contraintes temporelles pour des systèmes de contrôle-commande à l'aide des réseaux de Petri. In *Colloque Applications des méthodes formelles au développement de systèmes VLSI et systèmes de contrôle-command temps réel*, pp. 255–267, Montréal, Canada, 2–4 Octobre 1996.

Références :

[zAin06] Ainsworth, S. and Fleming, P. Evaluating authoring tools for teachers as instructional designers. Computers in Human Behavior, pp. 131–148, 2006.

[zAle06] Aleven, V., McLaren, B. M., Sewall, J., and Koedinger, K. The Cognitive Tutor Authoring Tools (CTAT) : Preliminary evaluation of efficiency gains. In Ikeda, M., Ashley, K., and Chan, T., editors, Proceedings of the 8th Int. Conf. on Intelligent Tutoring Systems (ITS 2006), pp. 61–70, Berlin, 2006. Springer Verlag.

[zAlu01] R. Alur, K. Etessami and M. Yannakakis. Realizability and verification of MSC graphs. In Proceedings of the 28th Int. Colloquium on Automata, Languages, and Programming, (ICALP'01) Greece, July, 2001, Springer, LNCS 2076, pp. 797-808, 2001.

[zBer02] S. Bernadi, S. Donatelli and J. Merseguer. From UML Sequence Diagrams and Statecharts to analysable Petri Models. In Proceedings of the 3rd Int. workshop on Software and performance, Rome, Italy, 2002, pp. 35-45 ISBN 1-58113-563-7, 2002.

[zBer05] B. Bérard, F. Cassez, S. Haddad, D. Lime, and O. (H.) Roux. Comparison of the expressiveness of timed automata and time Petri nets. In 3rd International Conference on Formal Modelling and Analysis of Timed Systems (FORMATS 05), volume 3829 of Lecture Notes in Computer Science, Uppsala, Sweden, September 2005. Springer.

[zBer04] B. Berthomieu, P.O. Ribet, F. Vernadat : The tool TINA : construction of abstract state spaces for Petri nets and time Petri nets, IJPR, Vol.42, N°14, pp.2741-2756, 15 Juillet 2004.

[zBit93] Bittencourt, G. and Marengoni, M., A Customizable Tool for the Generation of Production-Based Systems, in 8th Int. Conf. on Applications of Artificial Intelligence in Engineering (AIENG'93), pg 337-352, G. Rzevski, J. Pastor and R.A. Adey Ed., 1993 , Elsevier Applied Science.

[zBot05] Botafogo, R. and Moss, D. The MORENA Model for Hypermedia Authoring and Browsing. In Proc. of the Int. Conf. on Multimedia Computing and Systems, pp. 42–49, 1995.

[zBru03] Brusilovsky, P. and Peylo, C. Adaptative and Intelligent Web-based Educational Systems. In IJAIED : Int. Journal of Artificial Intelligence in Education, volume 13, pp. 159–172. IOS Press, 2003.

[zChe00] B. Pradin-Chézalviel : Application de la logique linéaire au raisonnement temporel sur les réseaux de Petri. Habilitation à Diriger des Recherches, 8 décembre 2000.

[zCos95] E. de B. Costa, M.A. Lopes, and E. Ferneda. Mathema : A learning environment based on a multi-agent architecture. In *L. N. in Artificial Intelligence*, vol. 991, pages 141–150, Oct. 1995.

[zCri04] Cristea, A. Evaluating Adaptive Hypermedia Authoring while Teaching Adaptive Systems. In ACM Symposium on Applied Computing (SAC2004), pp. 929–934, 2004.

[zDem02] B.A. Demissie A Framework for Semantics of UML Sequence Diagrams. In PVS Journal of Universal Computer Science (JUCS), vol. 8, no. 7, pp. 674-697, July 2002.

[zDub88] D. Dubois, H. Prade : Possibility theory : an approach to computerized processing of uncertainty. Plenum Press, 1988.

[zDub89] D. Dubois, H. Prade : Processing fuzzy temporal knowledge, *IEEE Trans. on Systems, Man and Cybernetics*, 14, N° 4, p.729-744, 1989.

[zFer99] Ferber, J. Multi-Agent System, An Introduction to Distributed Artificial Intelligence. Addison-Wesley Publishers, 1999.

[zFer03] Ferber, J., Gutknecht, O., Michel,F. From Agents to Organizations : an Organizational View of Multi-Agent Systems. 4th Int. Workshop on agent-oriented software engineering, IV AOSE 2003, Melbourne, Australia, 2003.

[zFrie03] J. Friedman-Hill, E.J. Jess, The Rule Engine for the Java Platform. Sandia National Laboratories, Livermore, CA, distributed computing systems edition, September 2003. http ://herzberg.ca.sandia.gov/jess/.

[zGal97] L. Gallon. Le modèle Réseaux de Petri Temporisés Stochastique : Extensions et Applications. Thèse de doctorat, LAAS-CNRS, Université Paul Sabatier de Toulouse, 1997.

[zGeh98] T. Gehrke, U. Goltz and H. Whrheim. The dynamic models of UML : towards a Semantics and its Application in the Development Process. In Hildesheimer Informatik-Bericht 11/98. Institut für Informatik, Universität Hildesheim, 1998.

[zGir95] J.Y. Girard : Linear Logic : its syntax and semantics, *in Advances in Linear Logic, Lecture Note Series 222*, Ed. Girard, Lafont, Regnier, Cambridge University Press, 1995.

[zGra93] J. Grabowski, P. Graubmann and E. Rudolph. Towards a Petri net based semantics definition for Message Sequence Charts. In O. Færgemand et al. editors. In Using Objects, Proceedings of the 6th SDL Forum, (SDL'93), pp. 179-190, North- Holland, 1993.

[zHar02] D. Harel and H. Kugler. Synthesizing State-Based Object Systems from LSC Specifications. Int. Journal of Foundations of Computer Science vol. 13, no. 1, pp. 5-51, 2002.

[zHey00] S. Heymer. A Semantics for MSC based on Petri-Net Components. In Proceedings of the 2nd Workshop of the SDL Forum Society on SDL and MSC (SAM'2000), Grenoble, France, June, 26-28, 2000.

[zITU96] ITU, Recommendation Z.120 : Message Sequence Charts (MSC), ITU General Secretariat, 1996. http ://www.itu.int.

[zJos94] C. M. Jos Baeten, Sjouke Mauw Sjouke Mauw. Delayed choice an operator for joining Message Sequence Charts. In Dieter Hogrefe and Stefan Leue editors, Formal Description Techniques VII, Proceedings of the 7th of the Seventh Int. Conf. on Formal Description Techniques, pp. 340-354, Chapman & Hall Berne, Switzerland, 1994.

[zJua97] G. Juanole, L. Gallon Concept of quantified abstract automaton and its advantage. Joint Int. Conf. on Formal Description Techniques for Distributed Systems and Communication Protocols (FORTE/PSTV'97), Osaka, Japan, November 18-21, 1997.

[zKli88] G.J. Klir, T.A. Folger. *Fuzzy sets, uncertainty, and information.* Prentice-Hall 1988, 355 p.

[zKna99] A. Knapp. A Formal Semantics for UML Interactions. In Robert France and Bernhard Rumpe, editors, Proceedings of the 2nd Int. Conf. on the Unified Modeling Language (UML'99), LNCS 1723, Springer, pp. 116-130, USA, 1999.

[zKos04] K. Koskimies and E. Makinen. Automatic Synthesis of State Machines from Trace Diagrams. Software Practice and Experience, vol. 24, no. 7, pp. 643-658, 1994.

[zKhi01] I. Khriss, M. El-koutbi and R.K. Keller. Automatic Synthesis of Behavioral Specifications from Scenarios. Journal of Integrated Design et Process Science, vol. 5, no. 3, pp. 53-77, September 2001.

[zKrü99] I. Krüger, R. Grosu, P. Scholz and M. Broy. From MSCs to Statecharts. In Distributed and Parallel Embedded Systems (DIPES), Kluwer Academic Publishers. 1999.

[zLam78] L. Lamport. Time, Clocks and the Ordering of Events in a Distributed System. Communications of the ACM, vol. 21, no.7, pp, 558-565, July 1978.

[zLi04] X. Li, Z. Liu and H. Jifeng. A Formal Semantics of UML Sequence Diagram. In Proceedings of the Australian Software Engineering Conf. 2004, Australia, April 2004.

[zLin93] C. Lin, A. Chaudhury, A. B. Whinston and D. C. Marinescu : Logical inference of Horn clauses in Petri nets models, In *IEEE Trans. on Knowledge and Data Engineering*, V5, n3, June 1993, p. 416-425.

[zLoo88] C. G. Looney. Fuzzy Petri nets for rule-based decision making, In IEEE Trans. on Systems Man and Cybernetics, V18, n1, 1988, p. 178-183.

[zMak01] E. Mäkinen and T. Systä. MAS - an interactive synthesizer to support behavioral modeling in UML. In Proceedings of the 23rd Int. Conf. on Software Engineering (ICSE'01), pp. 15-24, Toronto, Canada, 2001.

[zMan99] N. Mansurov and D. Zhukov. Automatic synthesis of SDL models in use case methodology. In R. Dssouli et al. editors, SDL Forum, Elsevier, pp. 225-240, 1999.

[zMer74] P. Merlin. *A Study of the recoverability of Computer Systems*. Phd thesis. University of California, Irvine, 1974.

[zNa01] Na, J. and Furuta, R. Dynamic Documents : Authoring, Browsing, and Analysis. Using a High-Level Petri Net-Based Hypermedia System. In Proc. of 10th ACM Symposium on Document Engineering (DocEng'01), pp. 38–47, New York, NY, November 2001. ACM PRESS.

[zNak97] Nakabayashi, K., Koike, Y., Maruyama, M., Touhei, H., Kato, Y., and Fukuhara, Y. Architecture of an Intelligent Tutoring System on the WWW. Proceedings of the 8th World Conf. of the AIED Society, August 1997.

[zNuz05] Nuzzo-Jones, G.,Walonoski, J., Heffernan, N., and Livak, T. The eXtensible Tutor Architecture : A New Foundations for ITS. Proceedings of the 12th Artificial Intelligence In Education, pp. 555–562, 2005.

[zPer03] Intégration des réseaux de Petri différentiels à objets dans une plate-forme de simulation dynamique hybride : application aux procédés industriels. Thèse INSA, novembre 2003, Toulouse.

[zPet89] G. Peterka and T. Murata : Proof procedure and answer extraction in Petri net model of logic programs, In *IEEE Transaction on Software Engineering*, V15, n2,February 1989, p.209-217.

[zPom92] L. Pomello, G. Rozenberg and C. Simone. A Survey of Equivalence Notions for Net Based System. In G. Rozenberg editor, Advances in Petri Nets 1992, Springer, LNCS 609, pp. 410-472, 1992.

[zOka03] M Okazaki, T Aoki and T Katayama. Formalizing sequence diagrams and state machines using Concurrent Regular Expression. In Proceeding of the 2nd Int. Workshop on Scenarios and State Machines : Models, Algorithms, and Tools; Held at the Int. Conf. on Software Engineering (ICSE'03) Portland, Oregon, USA, 2003.

[zOla02] K. Olaf. Compositional Semantics for Message Sequence Charts based on Petri Nets. PHD thesis, University of Berlin, 2002.

[zPik03] S. Pikin. Test des composants logiciels pour les télécommunications. PHD thesis. Université de Rennes, France, 2003.

[zRum91] J. Rumbaugh, M. Blaha, W. Premerlani, F. Eddy and W. Lorensen, Object-Oriented Modeling and Design. Prentice Hall, 1991.

[zSca93] H. Scarpelli and F. Gomide, Fuzzy Reasoning and High Level Fuzzy Petri Nets, in Proc. First European Congress on Fuzzy and Intelligent Technologies, Aachen, Germany, Sep 7-10, 1993, pg 600-605.

[zSeu02] M.C. Seung, H.K. Hyung, D.C. Sung and H.B. Doo. A semantics of sequence diagrams. Information Processing Letters, vol. 48, no. 3, pp. 125-130, Elsevier North-Holland, ISSN 0020-0190, 2002.

[zSib99] C. Sibertin-Blanc. CoOperative Objects : Principles, Use and Implementation. In Petri Nets and Object Orientation, G. Agha and F. De Cindio Eds., Lectures Notes in Computer Science, Springer-Verlag, 1999.

[zSib85] C. Sibertin-Blanc : High-level Petri nets with data structures, 6^{th} European Workshop on Application and Theory of Petri nets, Helsinki, Finland, June 1985.

[zSto99] H. Störrle. A Petri-Net Semantics for Sequence Diagrams. In KatharinaSpies and Bernhard Schätz, editors, 9. GI/ITG Fachgespräch Formale.

[zUch03] S.Uchitel, J.Kramer and J.Magee. Synthesis of behavioral models from scenarios. IEEE Transaction on Software Engineering, vol. 29, no. 2, pp. 99-115, February 2003.

[zVas01] Vassileva, J., Deters, R., Geer, J., McCalla, G., Bull, S., and Kettel, L. Lessons from Deploying I-Help. In Proceedings Int. Conf. on AI and Education, Texas, 2001.

[Vil04] Villani, E. ; Pascal, J.C. ; Miyage, P.E. ; Valette, R. Object oriented approach for cane sugar production : modelling and analysis. Control Engineering Practice, Amsterdam, v.12, n.10, p. 1279-1289, 2004.

[zWar97] Warendorf, K. and Tan, C. ADIS - An animated data structure intelligent tutoring system or Putting an interactive tutor on the WWW. Proceedings of the 8th World Conf. of the AIED Society, August 1997.

[zWen87] E. Wengers, Artificial Intelligence and Tutoring Systems : Computational and Cognitive Approaches to the Communication of Knowledge, 1987, Los Altos (CA), Morgan Kaufmann Publishers, ISBN 978-0934613262.

[zWil02] Willrich, R., Saqui-Sannes, P., P.Sénac, and Diaz, M. Multimedia authoring with hierarchical timed stream Petri nets and Java. In Multimedia Tools Appl., volume 16, pp. 7–27, Dordrecht (Holanda), 2002.

[zWhi00] J. Whittle and J. Schumann. Generating Statechart Designs from Scenarios. In Proceedings of the 22nd Int. Conf. on Software Engineering (ICSE'00), pp. 314-323, Limerick, Ireland, 2000.

[zYon98] T. Yoneda, H. Ryuba, CTL Model checking of time Petri nets using geometric regions, *IEICE Trans. inf. & Syst.*, Vol E81-D, No. 3, pp.297-396, 1998.

[zZia04] T. Ziadi, L. Hélouët and J-M. Jézéquel. Revisiting statecharts synthesis with an algebraic approach. In 26th Int. Conf. on Software Engineering ICSE'2004.

Annexe A

Théorie des possibilités et le temps

A.1 Date floue

Une date a a *une seule valeur* qui peut être mal-connue, et l'ensemble flou des ses valeurs possibles est un ensemble disjonctif [Yag84]. La connaissance que l'on dispose à propos de la date a est représenté par une distribution de possibilité $\pi_a(\tau) : \mathcal{T} \rightarrow [0, 1]$, $\tau \in \mathcal{T}$ (l'ensemble universel des instants de temps), délimité par l'ensemble flou A, représenté par le quadruplet $[\underline{a}, a_*, a^*, \overline{a}]$ [Dub89] sous la forme trapézoïdale (figure A.1.a).

Plus grande est la valeur de $\pi_a(\tau)$, plus grande est la possibilité que a soit égale à τ. Dans l'intervalle de temps $[\underline{a}, \overline{a}]$ (appelé support), $0 < \pi_a \leq 1$. Dans l'intervalle de temps $[a_*, a^*]$ (appelé noyau), $\pi_a(\tau) = 1$. En dehors de cet intervalle $[\underline{a}, \overline{a}]$, $\pi_a(\tau) = 0$. Plus grand est le support de l'ensemble flou A, et plus grande est l'incertitude. Il y a trois cas particuliers : a) forme triangulaire, $a_* = a^* = a$, que l'on note $[\underline{a}, a, \overline{a}]$ (figure A.1.b) ; b) cas imprécis, $\underline{a} = a_*$ et $a^* = \overline{a}$, que l'on note $[\underline{a}, \overline{a}]$; c) cas précis, $\underline{a} = a_* = a^* = \overline{a}$. L'ensemble flou A est normalisé si son noyau est non vide.

Dans le cas d'un réseau de Petri temporel flou, des intervalles flous sont associés aux transitions du réseau. Mais le tir d'une transition a lieu à une seule date a, même si cette date est mal-connue (représentée par une distribution de possibilité π_a). Le réseau de Petri temporel est un cas particulier des réseaux de Petri flou, celui où les intervalles temporels associés aux transitions sont imprécis.

A.2 Mesures de possibilité Π et de nécessité N

La théorie des possibilités permet d'évaluer l'incertitude à l'aide de deux mesures, la mesure de nécessité N (ou de certitude) et la mesure de possibilité Π. La mesure de possibilité $\Pi : 2^X \rightarrow [0, 1]$

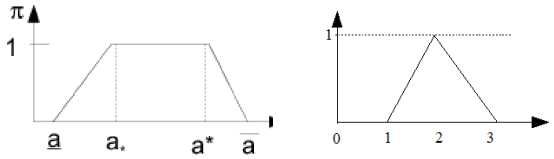


FIG. A.1 – Dates floues : a) trapézoïdale ; b) triangulaire.

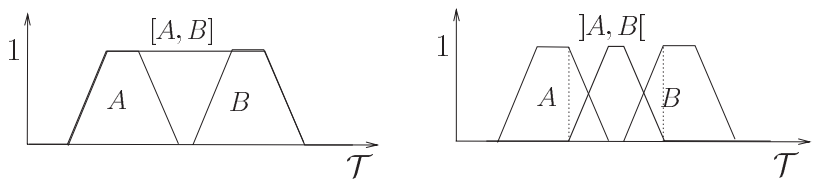


FIG. A.2 – Ensemble flou de dates : a) *possiblement* entre a et b ; b) *nécessairement* entre a et b.

permet d'évaluer la confiance que nous pouvons avoir sur l'affirmation $x \in A$, en connaissant la distribution de possibilités $\pi(x)$, $\Pi(A) = max_{x \in A}\pi(x)$. La mesure de nécessité est donnée par : $N(A) = 1 - \Pi(\bar{A}) = inf_{x \notin A}(1 - \pi(x))$.

La théorie des possibilités permet de représenter :
– l'ignorance totale : $\Pi(A) = \Pi(\bar{A}) = 1$, $N(A) = N(\bar{A}) = 0$
– la certitude que A est faux : $N(\bar{A}) = 1$ et $\Pi(A) = 0$,
– l'absence de certitude que A est vrai : $N(A) = 0$ et $\Pi(\bar{A}) = 1$.
La mesure de possibilité Π est décompositionnelle par rapport à l'union : $\Pi(A \cup B) = max(\Pi(A), \Pi(B))$;
la mesure de nécessité N est décompositionnelle par rapport à l'intersection : $N(A \cap B) = min(N(A), N(B))$.

Dans le cas de la construction du graphe de classes d'un réseau de Petri flou, ou de la construction d'une séquence de tir, il faut vérifier si une transition sensibilisée peut effectivement être franchissable, surtout s'il y a plusieurs transitions sensibilisées. À partir de la distribution de possibilité d'une date a (délimitée par A), quatre ensembles flous ont été définis [Dub89] :
– $[A, \infty)$: ensemble flou des instants de temps qui sont *possiblement après* la date t_i ;
– $]A, \infty)$: ensemble flou des instants de temps qui sont *nécessairement après* la date t_i ;
– $(-\infty, A]$: ensemble flou des instants de temps qui sont *possiblement avant* la date t_i ;
– $(-\infty, A[$: ensemble flou des instants de temps qui sont *nécessairement avant* la date t_i ;
L'ensemble flou $[A, B]$ des instants de temps qui sont *possiblement* après a et avant b est défini par la fonction d'appartenance

$$\tau \in x, \mu_{[A,B]}(\tau) = [A, \infty) \cap (-\infty, B] \tag{A.1}$$

et représenté sur la figure A.2.a. L'ensemble flou $]A, B[$ des instants de temps qui sont *nécessairement* après a et avant b est représenté sur la figure A.2.b.

Étant donné les distributions de possibilité des dates de tir de deux transitions t_1 et t_2 (par soucis de simplification la date de tir d'une transition t_i est aussi noté t_i), il est possible d'évaluer la possibilité et la nécessité pour que la date de t_1 soit avant celle de t_2 (et vice-versa), que l'on note $\Pi(t_1 \leq t_2)$ et $N(t_1 \leq t_2)$ [Dub89] par la formule :

$$\Pi(t_1 \leq t_2) = max\ [T_1, \infty) \cap (-\infty, T_2] \tag{A.2}$$

[Dub89] D. Dubois, H. Prade : Processing fuzzy temporal knowledge, *IEEE Trans. on Systems, Man and Cybernetics*, 14, N^o 4, p.729-744, 1989.

[Yag84] R.R. Yager. On different classes of linguistic variables defined via fuzzy subsets. Kibernetes, 13 :103-110, 1984.

Annexe B

Les réseaux de Petri et la logique linéaire

Un réseau de Petri (RdP) peux être décrit par une collection de formules bien-formées de la logique linéaire : les marquages sont décrits par des formules consommables et les transitions par des formules rémanentes.

En logique linéaire, si A et B sont des ressources, $A \otimes B$ représente une ressource consistant de A et B and $A \multimap B$ représente l'action d'utiliser un élément de A pour produire un élément de B (dépendance causale).

L'ensemble des connectifs de la logique linéaire est le suivant (pour plus de détails voir [zGir95]) :

- connectifs multiplicatifs : $\otimes$ (appelé "*times*") exprime l'existence simultanée de ressources, $\multimap$ ("*implication linéaire*") exprime une dépendance causale entre les ressources, et $\parr$ (appelé "*par*"), quand utilisé en connection avec la négation, exprime des notions de précédence ;
- connectifs additifs : & (appelé "*avec*") et $\oplus$ (appelé "*plus*") qui permet d'exprimer différents types de choix ;
- connectifs exponentiels : **!** (appelé "*of course*") et **?** (appelé "*why not*"). Ils permettent de réintroduire, si nécessaire, les notions de contraction et weakening, mais seulement pour des formules spécifiques et d'une façon contrôlée.

Un RdP peut être décrit par une collection de formules bien-formées de la logique linéaire. Les marquages sont décrits par une formule conjonctive (connectif $\otimes$) entre les atomes des places marquées, chaque atome étant factorisé par le nombre de jetons en chaque place. La formule M_o : $A \otimes 2.B$ décrit la distribution de jetons dans un réseau de Petri (un jeton dans la place A et deux dans la place B) : ça peut être un marquage partiel.

Chaque transition est traduite par une formule implicative entre deux formules de marquage : à gauche le marquage minimal nécessaire pour franchir la transition et à droite le marquage produit par son tir. Ainsi, un ensemble de formules de la logique linéaire correspondant à la description d'un réseau de Petri est de la forme ($m_k A_k$ indique qu'il y a m_k jetons dans A_k) :

- pour le marquage initial , M : $(\bigotimes_{k \in \{\text{places marquees}\}} m_k.A_k)$
- pour chaque transition t_j du réseau, t_j : $((\bigotimes_{i \in \text{places entree de } t_j} m_i.A_i) \multimap (\bigotimes_{o \in \text{places sortie de } t_j} m_o.A_o))$

Tous les aspects impliqués dans la dans le tir d'une transition (enlever les jetons des places d'entrée et en mettre dans les places de sortie) sont correctement traités par le calcul de séquents de la logique linéaire.