

Proposition de projet

S_4

Synthèse et supervision de systèmes, scénarios

System synthesis and supervision, scenarios

Benoît Caillaud, avec les contributions de :

Albert Benveniste, Philippe Darondeau, Gilles Lesventes et Sophie Pinchinat

Document disponible sous la forme de pages web à l'URL suivante :

<http://www.irisa.fr/pampa/perso/bcaillaud/s4-proposition/index.html>

5 novembre 2001

Résumé

Le projet a pour objectif la réalisation, par des méthodes algorithmiques, de systèmes réactifs et répartis à partir de spécifications partielles ou hétérogènes. Il s'agit de développer un ensemble de techniques, d'algorithmes et d'outils qui permette la synthèse de logiciels réactifs à partir d'une ou de plusieurs descriptions incomplètes spécifiant le comportement attendu du système du point de vue de la fonctionnalité (synchronisation, conflits, communication), du contrôle (sûreté, atteignabilité, vivacité), de l'architecture d'exécution (placement, cloisonnement), ou bien encore des performances quantitatives (temps de réponse, coût de communication, etc).

Ces techniques seront d'abord étudiées sur des modèles de base, comme les automates, les réseaux de Petri, les structures d'événements et leurs extensions temporisées. Les résultats obtenus seront alors adaptés aux modèles plus réalistes mais plus complexes généralement utilisés en télécommunications, ou pour les systèmes de production. En particulier, la notation UML (les scénarios, les statecharts et l'adjonction du langage réactif synchrone BDL) sera à la fois la base des outils développés et le coeur de la stratégie de valorisation des travaux du projet.

Table des matières

1	Membres du projet	3
2	Objectif du projet	3
2.1	Un domaine applicatif privilégié : le développement de logiciels de télécommunications	4
2.2	Conception par “composants-modèles” décrits selon des formalismes hétérogènes	6
2.3	Techniques développées et fondements scientifiques	7
2.4	Développements d’outils logiciels	8
3	Programme de recherche	9
3.1	Présentation d’ensemble	9
3.2	Techniques de synthèse de réseaux de Petri	10
3.2.1	Contexte des travaux sur les réseaux de Petri	10
3.2.2	Le point sur la synthèse de réseaux	11
3.2.3	Programme de recherche	13
3.3	Langages de scénarios	14
3.3.1	Contexte des travaux sur les langages de scénarios	14
3.3.2	Programme de recherche sur les langages de scénarios	15
3.4	BDL : un formalisme comportemental synchrone pour la description des systèmes d’objets répartis	17
3.4.1	Contexte des travaux sur le langage BDL	17
3.4.2	Programme de recherche	19
3.4.3	Développements logiciels et transferts	20
3.5	Thèmes de recherches transversaux	22
3.5.1	Automates et logiques pour le contrôle de systèmes ouverts répartis	22
3.5.2	Analyse quantitative de systèmes à l’aide d’algèbres tropicales	26
3.5.3	Modélisation de systèmes à l’aide de structures d’événements régulières	26
4	Positionnement vis-à-vis des autres projets du programme 1c	27
5	Relations industrielles et académiques	29
5.1	Insertion dans la communauté scientifique	29
5.2	Relations industrielles	30
6	Enseignement, formation, animation scientifique	30
7	Bibliographie des membres du projet	31

Préambule

Ce document présente une proposition de projet de recherche dans le domaine de l'ingénierie des logiciels réactifs et répartis. Il est composé de chercheurs issus des projets Ep-Atr, Pampa et Paragraphe. Il a vocation à évoluer, en fonction des objectifs de recherche de ses membres ainsi que des collaborations avec les autres projets du programme 1c à l'IRISA.

1 Membres du projet

Chercheurs permanents

- Albert Benveniste, DR INRIA, à temps partiel,
- Benoît Caillaud, CR INRIA, responsable scientifique,
- Philippe Darondeau, DR INRIA,
- Gilles Lesventes, MdC IFSIC, université de Rennes 1,
- Sophie Pinchinat, MdC IFSIC, université de Rennes 1.

Assistante de projet

- Marie-Noëlle Georgeault, INRIA.

Doctorant

- Pierre Le Maigat, Boursier MENESR,
- Stéphane Riedweg, Boursier INRIA.

2 Objectif du projet

L'objectif scientifique du projet peut être caractérisé par les éléments suivants :

Un domaine applicatif privilégié : le développement de logiciels de télécommunications, avec comme cible particulière l'adaptation de services. Ce domaine nous semble être le plus propice au transfert de nos résultats, sans pour autant être exclusif. Les systèmes embarqués et les systèmes de production sont également des cibles envisageables.

Un contexte méthodologique : le développement de méthodes et outils pouvant assister les concepteurs lors des premières étapes de la conception de logiciels réactifs et répartis. Le point difficile est d'offrir une conception par "composants-modèles" décrits selon des formalismes hétérogènes, avec un assemblage fiable de ces composants.

Des fondements scientifiques et techniques : les modèles et méthodes formelles associés, prenant en compte les aspects répartis et de concurrence vraie.

Un effort sur le développement de prototypes et leurs transferts, effort qui sera réparti entre l'outil BDL pour UML, et des outils plus exploratoires, pour les langages de scénarios et la synthèse de réseaux.

Ces divers aspects sont détaillés ci-après.

2.1 Un domaine applicatif privilégié : le développement de logiciels de télécommunications

La proposition de projet Inria SACRES présentée par Jean-Bernard Stéfani à l'UR Rhône-Alpes offre une analyse approfondie de l'état de l'art en matière de logiciel pour les télécommunications, nous y renvoyons le lecteur. Les tendances majeures peuvent être résumées comme suit, pour ce qui concerne les objectifs de la présente proposition :

Spécifications des protocoles et services. Services et protocoles doivent être aisés à adapter, et composables. Les spécifications sont naturellement exprimées sous la forme d'exigences que le système doit être capable de satisfaire. Ces exigences sont de quatre sortes et servent à décrire le comportement attendu du système du point de vue de la fonctionnalité (synchronisation, conflits, communication), du contrôle (sûreté, atteignabilité, vivacité), de l'architecture d'exécution (placement, cloisonnement), ou bien encore des performances quantitatives (temps de réponse, coût de communication, etc).

Déploiement sur une infrastructure donnée. Une tendance importante de l'industrie des télécommunications est le développement d'un nombre rapidement croissant d'infrastructures logicielles (*middleware*) destinées à faciliter l'intégration des protocoles et services, l'administrabilité des systèmes résultants et leur adaptabilité [33,68,76,30,25]. Le déploiement sur de telles infrastructures, des protocoles et services spécifiés doit donc pouvoir se faire de manière sûre et efficace.

Parmi les thématiques sous-jacentes, nous avons choisi de nous focaliser sur quelques problèmes plus particulièrement, nous les détaillons maintenant.

L'adaptation de services Les équipementiers et opérateurs sont fréquemment confrontés au besoin d'intégrer rapidement de nouvelles fonctionnalités dans leurs mises en oeuvres

[33] N. Davies, K. Raymond, and J. Seitz, editors. *Proceedings of Middleware'98, IFIP International Conference on Distributed Systems Platforms and Open Distributed Processing*, Lake District National Park, UK, September 1998. Springer.

[68] J. Sventek and G. Coulson, editors. *Middleware 2000 · IFIP/ACM International Conference on Distributed Systems Platforms and Open Distributed Processing*, volume 1795 of *LNCS*, New York, NY, USA, April 2000. Springer.

[76] Odp reference model: foundations and architecture. ITU-T Recommendations X.902 and X.903, ISO/IEC 10746, November 1995.

[30] The common object request broker: architecture and specification. OMG, March 1999.

[25] Corba components. OMG, March 1999.

(piles) de protocoles et de services. De l'avis général des intéressés, ceci présente des difficultés sérieuses, et requiert de longues phases de tests après modification et intégration, afin de valider le comportement d'ensemble et l'absence d'interactions indésirables entre services. Actuellement, cette intégration de nouveaux services est réalisée au niveau du code exécutable, et non pas de la spécification ou des exigences ^[54,24].

Nous proposons d'étudier une nouvelle méthodologie pour l'adaptation de services, et de l'outiller.

Dans un premier temps, nous supposons que le système, constitué d'un ensemble de services existants, ainsi que les nouveaux services à intégrer, sont décrits par un ensemble de spécifications formelles. En l'occurrence, nous considérons que UML ^[72], standard de fait émergent pour la modélisation de logiciels en télécommunications, est un bon cadre pour exprimer de telles spécifications. De façon à préserver la souplesse de spécification, nous pensons important de permettre le recours à des formalismes hétérogènes pour une telle spécification (par exemple la combinaison de diagrammes d'états pour les services préexistants, et de diagrammes de séquence ou HMSC ^[52] pour les services additionnels à intégrer). Les tâches devant alors être conduites sont pour l'essentiel les suivantes :

- composer les spécifications hétérogènes décrivant le système à réaliser (services existants et additionnels) ;
- vérifier certaines propriétés sur cette composition de spécifications (fonctionnalités, contrôle, adéquation à l'architecture, performances quantitatives) ;
- restituer un modèle homogène, validé, de l'ensemble des services ainsi adapté, ce modèle constituant le point d'entrée de la phase de déploiement, développée ci-après.

La démarche ci-dessus repose sur la disponibilité d'un modèle pour les services préexistants, ainsi que pour les adaptations. Or, les organismes de normalisation ne fournissent pas de tels modèles aujourd'hui. En l'absence d'outil adéquat, ces modèles doivent donc être construits à la main.

Par conséquent, à plus long terme, il nous semble pertinent d'étudier, en collaboration avec d'autres équipes plus centralement axées sur l'analyse de programmes, des techniques de *reverse engineering* permettant d'abstraire le squelette de contrôle et de communication à partir de code objet exécutable, et de le représenter à l'aide des formalismes de description déjà mentionnés.

Déploiement sur une infrastructure donnée La démarche générale est la suivante. On produit une structure de contrôle et de communication distribuée, dont la validité repose uniquement sur des hypothèses de nature générique portant sur les services de communication et de nommage offerts par le *middleware* considéré. Ceci permet de couvrir

[54] K. Kimbler and W. Bouma, editors. *FIW'98 Feature Interactions in Telecommunications and Software Systems*, V, Lund, Sweden, September 1998. IOS Press.

[24] E. Cameron. A feature interaction benchmark for in and beyond, 1994.

[72] Omg unified modeling language specification. <http://cgi.omg.org/cgi-bin/doc?ad/99-06-09>, June 1999. Version 1.3.

[52] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, September 1993.

des familles de *middleware* support, plutôt qu'un *middleware* en particulier. Dans l'état actuel de notre savoir-faire, nous ne savons prendre en compte que des aspects purement fonctionnels. Notamment, nous supposons la disponibilité d'une architecture support fiable. Et nous ne savons pas prendre en compte les aspects temporels lors de cette phase de synthèse qu'est le déploiement. Enfin, nous devons apprendre à maîtriser la dynamique. Nous considérons que ce thème restera un terrain de coopération étroite entre plusieurs projets.

2.2 Conception par “composants-modèles” décrits selon des formalismes hétérogènes

Les éléments suivants sont au centre de notre démarche :

Hétérogénéité des notations. Nous pensons qu'UML ^[72] offre une diversité adéquate dans les styles de notations (notations de type “scenarios”, notations de type “machines à états”, aspects architecturaux). Pour les scenarios on dispose des cas d'utilisation (use case), diagrammes de séquence et de collaboration. Pour les machines à états, ce sont les diagrammes d'états, les *statecharts* ^[42], l'*action semantics* ^[6], et les évolutions futures concernant la convergence de SDL ^[73] vers UML. Les aspects architecturaux sont les plus populaires dans UML : diagrammes de classes, d'objets, de déploiement.

Notre objectif est d'unifier les sémantiques des vues comportementales d'UML, et d'en permettre, par là même, la combinaison.

Des exigences aux Multiples facettes. Les exigences sur le système devant être réalisé ou modifié sont de plusieurs natures :

- fonctionnalité : synchronisation, conflits et communication ;
- contrôle : sûreté, atteignabilité et vivacité ;
- Architecture d'exécution : localisation et cloisonnement ;
- performances quantitatives : temps de réponse et coût de communication — nous laissons de côté les aspects, importants en ce qui concerne les logiciels embarqués, liés à la consommation de ressources : mémoire, énergie, etc.

Ces différents aspects sont exprimables en UML ou dans ses extensions envisagées (UML-RT ^[40]).

[72] Omg unified modeling language specification. <http://cgi.omg.org/cgi-bin/doc?ad/99-06-09>, June 1999. Version 1.3.

[42] D. Harel. Statecharts: A visual formulation for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[6] Response to omg rfp ad/98-11-01, action semantics for the uml. OMG, February 2001.

[73] International Telecommunication Union, editor. *Recommendation Z.100 - Specification and description language (SDL)*. ITU, November 1999.

[40] R. Grosu, M. Broy, B. Selic, and Gh. Stefanescu. Towards a calculus for uml-rt specifications. *Seventh OOPSLA Workshop on Behavioral Semantics of OO Business and System Specifications, Vancouver, Canada, Monday, October, 19th, 1998.*, 1998.

Le problème est de pouvoir manipuler ces exigences au niveau système. C'est un de nos objectifs

Ces techniques de modélisation et d'analyse étant disponibles, il convient de procéder à la *synthèse* vers un modèle homogène et exécutable, satisfaisant aux exigences ci-dessus.

2.3 Techniques développées et fondements scientifiques

Le projet s'attachera à développer un ensemble de techniques, d'algorithmes et d'outils qui permette la synthèse de logiciels réactifs à partir d'une ou de plusieurs descriptions incomplètes spécifiant le comportement attendu du système (fonctionnalité, contrôle, architecture d'exécution, performances quantitatives). Ceci implique, d'une part, de savoir transformer ces descriptions hétérogènes en descriptions homogènes de plus bas niveau : automates communicants, réseaux de Petri, etc. D'autre part il est aussi nécessaire de pouvoir valider les descriptions hétérogènes : vérification de propriétés.

La démarche scientifique du projet s'appuiera sur une approche rigoureuse de ces questions et la constitution de fondements théoriques solides. Cela permettra de prouver la correction des transformations envisagées (fonctionnalités et contrôle) ; mais aussi l'optimalité des performances quantitatives des réalisations produites par nos méthodes. Les techniques de synthèse et de vérification envisagées seront d'abord étudiées sur des modèles de base (automates, réseaux de Petri, structures d'événements, systèmes de transitions synchrones). Les résultats obtenus seront alors adaptés aux modèles plus réalistes mais plus complexes généralement utilisés pour la conception des systèmes de télécommunications (problèmes évoqués au paragraphe 2.1), de production ou les systèmes embarqués.

Le programme scientifique du projet est décrit en détail en partie 3 du présent document. On y trouve trois axes principaux de recherches, et plusieurs thèmes transversaux à ces trois axes.

Synthèse de réseaux de Petri : Il s'agit de poursuivre l'activité principale du projet Paragraphe au cours de ces dernières années. En plus du développement de cette théorie, les applications visées sont la réalisation répartie de spécifications incomplètes (scénarios, etc) et le contrôle réparti de systèmes à événements discrets. Le programme de recherche proposé est détaillé au paragraphe 3.2

Langages de scénarios : HMSC ^[52] et nouveaux formalismes à base de structures d'événements infinies régulières. Les aspects développés seront la caractérisation de classes décidables de scénarios, la réalisation de scénarios en familles de processus communicants et l'utilisation de nouveaux formalismes comportementaux pour l'expression des exigences — paragraphe 3.3.

[52] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, September 1993.

BDL : Le troisième axe porte sur le développement des fondements théoriques et des aspects expérimentaux d'un formalisme synchrone pour la modélisation de systèmes d'objets réactifs : le langage BDL — paragraphe 3.4.

Recherches transversales De ces trois axes de recherche, nous dégagons trois sujets de recherches transversaux : les approches automates et logiques pour le contrôle, avec un accent porté sur les systèmes ouverts et la répartition (paragraphe 3.5.1), la modélisation par structures d'événements régulières (paragraphe 3.5.3) et l'analyse quantitative de systèmes à l'aide d'algèbres tropicales — paragraphe 3.5.2.

2.4 Développements d'outils logiciels

Toutes les activités de développement d'outils seront en harmonie avec le programme de recherche évoqué ci-dessus. Lorsque nécessaire, le développement d'outils plus conséquents sera conduit en coopération étroite avec d'autres projets ou partenaires industriels.

BDL : Nous considérons a priori que BDL est notre effort principal en matière de tentative d'industrialisation et de transfert. Un premier ensemble de prototypes ont été réalisés à des fins de démonstration de faisabilité uniquement ^[13,69,23]. Une deuxième phase est maintenant envisagée, toujours en collaboration avec le projet Espresso. Il s'agira alors de développer des outils de qualité pré-industrielle, en collaboration avec les partenaires cités. Les développements prévus sont détaillés, plus loin, au paragraphe 3.4.3

SLIM : Il s'agit de poursuivre le développement du prototype de démonstration SLIM ^[46] développé par Loïc Hérouët, pendant son travail de doctorat ^[45]. SLIM est destiné à accueillir l'ensemble des algorithmes d'analyse, de transformation et de réalisation de HMSC, que nous serons amené à développer au cours du travail de recherche sur les langages de scénarios — paragraphe 3.3. Le prototype SLIM sera aussi utilisé au cours de la collaboration entre les projets S_4 , Triskell et France Télécom, Recherche et Développement (Lannion) — commencé le 11 septembre 2001, durée deux ans.

[13] The bdl project, home page. <http://www.irisa.fr/pampa/bdl/>.

[69] J.-P. Talpin, A. Benveniste, B. Caillaud, C. Jard, Z. Bouziane, and H. Canon. BDL, a language of distributed reactive objects. In *ISORC'98, The 1st IEEE International Symposium on Object-oriented Real-time Distributed Computing*, Kyoto, Japan, April 1998.

[23] Benoît Caillaud, Jean-Pierre Talpin, Jean-Marc Jézéquel, Albert Benveniste, and Claude Jard. Bdl: A semantics backbone for uml dynamic diagrams. Technical Report RR-4003, Irisa/INRIA Rennes, September 2000. <http://www.inria.fr/RRRT/RR-4003.html>.

[46] L. Hérouët. Utiliser et maintenir slim. <http://www.irisa.fr/pampa/perso/helouet/>, octobre 2000.

[45] L. Hérouët. *Analyse des exigences des systèmes répartis exprimées par des langages de scénarios*. PhD thesis, Ecole doctorale MATHISSE, Université de Rennes 1, Octobre 2000.

SYNET : Ce prototype sert de démonstrateur pour les algorithmes de synthèse de réseaux ^[7,8] et de répartition d'automates ^[20,9] développés à l'IRISA. Son développement sera poursuivi et nous y intégrerons les nouvelles techniques développées au cours des recherches sur la synthèse de réseaux de Petri — paragraphe 3.2.

3 Programme de recherche

3.1 Présentation d'ensemble

Le programme de recherche se décompose en trois axes distincts qui s'inscrivent bien dans la thématique générale du projet : la réalisation et la vérification de descriptions hétérogènes et incomplètes de systèmes réactifs et répartis.

Le premier axe de recherche porte sur la synthèse de réseaux de Petri et est la poursuite de l'activité principale du projet Paragraphe au cours de ces dernières années. En plus de la poursuite du développement de cette théorie, les applications visées sont la réalisation répartie de spécifications incomplètes (scénarios, etc) et le contrôle réparti de systèmes à événements discrets.

Un deuxième axe porte sur les langages de scénarios : HMSC ^[52] et nouveaux formalismes à base de structures d'événements infinies régulières. Les aspects développés seront la caractérisation de classes décidables de scénarios, la réalisation de scénarios en familles de processus communicants et l'utilisation de nouveaux formalismes comportementaux pour l'expression des exigences (structures d'événements, etc).

Le troisième axe porte sur le développement des fondements théoriques et des aspects expérimentaux d'un formalisme synchrone pour la modélisation de systèmes d'objets réactifs : le langage BDL.

Des travaux sur l'analyse quantitative de systèmes décrits de manière incomplète ou hétérogène (réseaux de Petri et langages de scénarios), utilisant des résultats sur les algèbres tropicales, viendraient compléter chacun des trois axes de recherche.

De ces trois axes de recherche, nous en dégageons trois sujets de recherches transversaux : les approches automates/logiques pour le contrôle de systèmes ouverts et répartis, la modélisation par structures d'événements régulières et l'analyse quantitative de systèmes à l'aide d'algèbres tropicales.

-
- [7] E. Badouel, L. Bernardinello, and Ph. Darondeau. Polynomial algorithms for the synthesis of bounded nets. In *CAAP'95*, volume 915 of *LNCS*, pages 647–679, Aarhus, 1995. Springer.
 - [8] E. Badouel, L. Bernardinello, and Ph. Darondeau. The synthesis problem for elementary nets is np-complete. *Theoretical Computer Science*, 186:107–134, 1997.
 - [20] B. Caillaud. Bounded petri-net synthesis techniques and their applications to the distribution of reactive automata. *JESA, European Journal on Automated Systems*, 33(8–9):925–942, 1999.
 - [9] E. Badouel, B. Caillaud, and Ph. Darondeau. Distributing finite automata through petri net synthesis. article soumis pour publication dans le "Journal on Fundamental Aspects of Computer Science", 2000.
 - [52] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, September 1993.

3.2 Techniques de synthèse de réseaux de Petri

Participants : Benoît Caillaud, Philippe Darondeau et Gilles Lesventes

3.2.1 Contexte des travaux sur les réseaux de Petri

En 1992, Eric Badouel et Philippe Darondeau ont abordé accidentellement les problèmes liés à la synthèse des réseaux de Petri ^[12]. L'objectif était de produire une caractérisation de la classe des automates concurrents définissables dans des algèbres de processus données par des règles sémantiques structurelles. A cette fin, il s'est avéré nécessaire d'insérer entre automates concurrents et règles opérationnelles, un type particulier de réseaux de Petri, celui des réseaux traces. Ayant saisi l'intérêt et les enjeux de la synthèse de réseaux, ils se sont éloignés de leurs sujets alors familiers (vraie concurrence, effectivité et modèles extensionnels dans les calculs de processus) pour entrer plus avant dans ce nouveau domaine.

Il s'en est suivi une collaboration entre Eric Badouel, Philippe Darondeau et plus tard Benoît Caillaud. Un ensemble d'algorithmes de synthèse de réseaux de Petri de différents types ont été conçus et implantés dans l'outil SYNET ^[21]. Les applications de ces algorithmes sont la réalisation et le contrôle de supervision de systèmes réactifs répartis. Des résultats expérimentaux préliminaires ont permis de montrer le potentiel de la synthèse de réseaux vis à vis de ce type d'applications. C'est une méthode puissante et réaliste de réalisation de systèmes répartis accomplissant un service spécifié par un automate ou un langage ^[9]. C'est une méthode possible pour prototyper un système réparti dont certains comportements (cas d'utilisations) sont décrits par des scénarios ^[22]. C'est la seule méthode connue pour calculer directement des contrôleurs asynchrones répartis de systèmes à événements discrets. Beaucoup reste néanmoins à entreprendre afin de corroborer et faire partager cette thèse. L'enjeu est important : il s'agit non seulement de construire des systèmes répartis sûrs et fiables, mais encore de réduire leurs coûts de conception, en offrant à l'ingénierie des logiciels une technologie de conception assistée semblable à celle utilisée pour la conception des circuits matériels. Tel est notre objectif de recherche sur la synthèse de réseaux de Petri, pour les prochaines années.

[12] E. Badouel and Darondeau Ph. Trace nets. In *REX Workshop on "Semantics: Foundations and Applications"*, volume 666 of *LNCS*, pages 21–50, Beekbergen, 1992. Springer.

[21] B. Caillaud. Synet: A synthesizer of distributable bounded petri-nets from finite automata, version 2.0a. <http://www.irisa.fr/pampa/logiciels/synet/>, March 2000.

[9] E. Badouel, B. Caillaud, and Ph. Darondeau. Distributing finite automata through petri net synthesis. article soumis pour publication dans le "Journal on Fundamental Aspects of Computer Science", 2000.

[22] B. Caillaud, P. Darondeau, L. Hélouët, and G. Lesventes. Hmscs as specifications... with pn as completions. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Proceedings of the summer school MOVEP'2k: Modelling and verification of parallel processes*, pages 87–103, Nantes, June 2000.

3.2.2 Le point sur la synthèse de réseaux

Avant de présenter les éléments du programme de recherche concernant la synthèse de réseaux, faisons le point sur ce sujet. La forme la plus classique de ce problème est la suivante : étant donné un système de transitions fini étiqueté, et une classe particulière de réseaux de Petri, décider si le système de transitions est représentable par un réseau de cette classe, et construire un réseau minimal le réalisant — c'est à dire, dont le graphe des marquages accessibles est isomorphe à l'automate considéré. On doit ici prendre la notion de réseaux de Petri au sens large : un réseau de Petri se comporte comme le produit synchronisé d'une famille finie de systèmes de transitions, choisis dans un éventail donné de systèmes composants, à raison d'une composante par place du réseau. Le problème de la synthèse de réseaux est ainsi le problème de la décomposition en produit. Une solution non effective a été donnée en identifiant places de réseaux et morphismes de systèmes de transitions, ayant pour cible le type de composant des réseaux ^[10,11]. Le contenu algorithmique de la synthèse a été étudié pour les réseaux élémentaires, dont les places sont valuées dans les booléens, et pour les réseaux de Petri généraux, dont les places sont valuées dans les entiers.

En dehors de l'IRISA, la plupart des chercheurs étudient la synthèse des réseaux élémentaires. Un outil performant développé à cette fin (PETRIFY ^[31]) a permis des progrès significatifs dans la conception des circuits matériels asynchrones. Luca Bernardinello, Eric Badouel et Philippe Darondeau ont montré que que la synthèse des réseaux élémentaires est pourtant un problème NP-complet ^[8]. Nous avons choisi à l'opposé d'étudier la synthèse de réseaux de Petri généraux. Ce choix est justifié par les résultats suivants : la synthèse de réseaux de Petri généraux est un problème décidable en temps polynomial ^[7]. Il l'est aussi pour la facilité de prise en compte algorithmique des contraintes de répartition proposées par Benoît Caillaud ^[20,9], permettant la traduction systématique des réseaux synthétisés en ensembles d'automates communiquant par échanges asynchrones de messages.

Il n'est pas toujours facile de donner les spécifications d'un système ou d'un service sous la forme d'un système de transitions donné explicitement. On peut préférer décrire le service attendu par un langage, ou mieux, l'encadrer (par inclusion ensembliste) entre

-
- [10] E. Badouel and Ph. Darondeau. Dualities between nets and automata induced by schizophrenic objects. In *CTCS'95*, volume 953 of *LNCS*, pages 24–43. Springer, 1995.
 - [11] E. Badouel and Ph. Darondeau. Theory of regions. In *Dagstuhl*, volume 1491 of *LNCS*, pages 529–586. Springer, 1999.
 - [31] J. Cortadella and al. Petrify: a tool for synthesizing elementary petri nets and asynchronous circuits. <http://www.lsi.upc.es/~jordic/petrify/petrify.html>, 1999.
 - [8] E. Badouel, L. Bernardinello, and Ph. Darondeau. The synthesis problem for elementary nets is np-complete. *Theoretical Computer Science*, 186:107–134, 1997.
 - [7] E. Badouel, L. Bernardinello, and Ph. Darondeau. Polynomial algorithms for the synthesis of bounded nets. In *CAAP'95*, volume 915 of *LNCS*, pages 647–679, Aarhus, 1995. Springer.
 - [20] B. Caillaud. Bounded petri-net synthesis techniques and their applications to the distribution of reactive automata. *JESA, European Journal on Automated Systems*, 33(8–9):925–942, 1999.
 - [9] E. Badouel, B. Caillaud, and Ph. Darondeau. Distributing finite automata through petri net synthesis. article soumis pour publication dans le "Journal on Fundamental Aspects of Computer Science", 2000.

deux langages spécifiant pour l'un le comportement minimal exigé, pour l'autre le comportement maximal autorisé. Le problème est alors de construire un réseau de Petri qui réponde à ces spécifications, si il en existe un. Cette forme du problème de synthèse a été résolue pour les langages réguliers, et pour d'autres classes de langages ayant une image commutative semi-linéaire ^[32]. Allant plus loint, on peut juger que la donnée d'un langage par une expression régulière ou par une grammaire n'est pas le style spécifications préféré d'un concepteur, mieux à même de procéder à la description graphique de scénarios. Dans le cas des langages de scénarios donnés par des HMSCs, nous avons montré comment calculer la meilleure approximation supérieure d'un tel ensemble par un réseau de Petri répartissable ^[22]. L'algorithme procède par calcul de l'ensemble des rayons extrémaux d'un cône polyédrique. Ses performances restent à tester.

Ces différents résultats sont porteurs de promesses qui doivent maintenant être validées par l'expérience. La synthèse de réseaux de Petri a été utilisée par Benoît Caillaud pour générer des protocoles de communications ^[20,9]. Elle a été utilisée par Xiaolan Xie et ses collègues du Loria (dans le cadre de l'ARC MARS) pour superviser des systèmes de production. N'ont servi dans les deux cas que les algorithmes de synthèse de réseaux à partir de systèmes de transitions donnés explicitement, seuls implantés à ce jour dans l'outil SYNETH ^[21]. De meilleurs contrôleurs peuvent être obtenus en synthétisant les réseaux à partir de langages. Les résultats montrent néanmoins que la méthode est viable : on peut construire des contrôleurs pour des systèmes comportant environ 10000 états, et on obtient des contrôleurs assez compacts.

On devra tôt ou tard faire face à plusieurs types de problèmes. Il faudra développer des algorithmes de synthèse symboliques pour traiter des systèmes de très grande taille. Il faudra définir d'autres algorithmes de synthèse approchée (par exemple avec insertion d'actions silencieuses) pour répondre dans tous les cas à la demande de l'utilisateur. Il faudra enfin résoudre les problèmes nés de l'adaptation pragmatique de la synthèse à tel ou tel domaine spécifique : prise en compte d'actions incontrôlables, etc. C'est certainement sur ces aspects techniques que se jouera l'avenir de la synthèse des réseaux de Petri.

-
- [32] Ph. Darondeau. Deriving unbounded nets from formal languages. In *CONCUR '98*, volume 1466 of *LNCS*, pages 533–548. Springer, 1998.
 - [22] B. Caillaud, P. Darondeau, L. Hélouët, and G. Lesventes. Hmscs as specifications... with pn as completions. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Proceedings of the summer school MOVEP'2k: Modelling and verification of parallel processes*, pages 87–103, Nantes, June 2000.
 - [20] B. Caillaud. Bounded petri-net synthesis techniques and their applications to the distribution of reactive automata. *JESA, European Journal on Automated Systems*, 33(8–9):925–942, 1999.
 - [9] E. Badouel, B. Caillaud, and Ph. Darondeau. Distributing finite automata through petri net synthesis. article soumis pour publication dans le "Journal on Fundamental Aspects of Computer Science", 2000.
 - [21] B. Caillaud. Syneth: A synthesizer of distributable bounded petri-nets from finite automata, version 2.0a. <http://www.irisa.fr/pampa/logiciels/syneth/>, March 2000.

3.2.3 Programme de recherche

Synthèse de contrôleurs asynchrones Les travaux menés dans le cadre de l'ARC MARS ont montré que la synthèse de réseaux de Petri peut être appliquée avec succès à la supervision de systèmes discrets. Le principe en est le suivant. Soit un système discret décrit par un réseau, avec un ensemble d'états (marquages) interdits et un ensemble d'événements (transitions) incontrôlables. On calcule le graphe des marquages accessibles du réseau, puis l'ensemble des marquages critiques menant à des marquages interdits par des suites de transitions incontrôlables. On calcule ensuite la restriction du graphe induite par l'exclusion des marquages critiques. On construit finalement un réseau de Petri (le superviseur) à partir du graphe restreint.

La synthèse de réseaux à partir de langages semble mieux adaptée au calcul de superviseurs puisque tout impératif de contrôle portant sur les états peut être reformulé en terme de comportements. Sur cette voie, divers problèmes devront être traités. L'un de ceux-ci est d'étendre la synthèse à des langages dont l'image commutative n'est pas semi-linéaire (tel est d'ailleurs le cas général des langages de réseaux de Petri). Un autre est de calculer des approximations supérieures (plus précisément des fermetures). Le problème crucial reste toutefois la synthèse de superviseurs sans blocages, ou mieux de superviseurs vivants.

On peut appliquer la synthèse de réseaux de Petri à la construction de contrôleurs asynchrones et répartis. Ceci ne pose pas de difficulté particulière : étant donné un système réparti dont l'ensemble des états accessibles est fini, un ensemble d'états interdits, un ensemble d'événements incontrôlables, SYNETH [21] peut être utilisé pour calculer un superviseur composé de plusieurs automates synchronisés localement avec le système réparti supervisé et communiquant par échanges asynchrones de messages. Le problème qui se pose alors est d'étudier la qualité des contrôleurs ainsi obtenus. Ceux-ci peuvent ne pas être optimaux au sens strict puisque les contrôleurs optimaux sont souvent synchrones ou séquentiels. D'autres notions doivent donc être introduites dans le cas des systèmes sans synchronisation globale.

Philippe Darondeau et Xiaolan Xie (du Loria) poursuivent par ailleurs un travail indépendant sur le contrôle optimal des graphes marqués, une classe de réseaux de Petri qui couvre divers systèmes de production.

Instrumentation de la synthèse Nous nous intéresserons à deux sujets liés qui semblent incontournables. Il s'agit, par deux méthodes différentes, de fournir un retour utilisable en cas d'échec de la synthèse exacte (c'est à dire à un isomorphisme près). La première méthode consiste à transformer le système donné, en y insérant des transitions silencieuses afin de le rendre synthétisable. Les études menées avec SYNETH [21] indiquent que c'est l'une des voies à suivre, mais la "méthode" reste à construire. Le second problème est le calcul d'un "résidu" de la synthèse, c'est à dire la composante non synthétisable du système donné. Le but est d'itérer la synthèse à partir de ce résidu en utilisant un nouveau jeu de composants.

[21] B. Caillaud. Syneth: A synthesizer of distributable bounded petri-nets from finite automata, version 2.0a. <http://www.irisa.fr/pampa/logiciels/syneth/>, March 2000.

3.3 Langages de scénarios

Participants : Albert Benveniste, Benoît Caillaud, Philippe Darondeau, Gilles Lesventes, Pierre Le Maigat, Sophie Pinchinat et Stéphane Riedweg

3.3.1 Contexte des travaux sur les langages de scénarios

Le programme de recherche proposé recouvre en partie celui proposé pour l'ARC FISC (responsable Benoît Caillaud, d'une durée de deux ans et ayant commencé au début de l'année 2001) qui regroupe des membres des projets Triskell, S_4 , Metalau (Rocquencourt) et Bip (Rhône-Alpes), ainsi que des chercheurs du LIAFA, université Paris 7.

La popularité croissante des notations graphiques en tant que spécifications est bien illustrée par les langages de scénarios et plus particulièrement les diagrammes de séquences ou *message sequence charts* (MSC) [52]. Le langage des MSC est une notation standardisée par l'union internationale des télécommunications (ITU), et fait aussi partie intégrante d'UML, où on l'utilise pour spécifier l'interaction entre les instances des classes d'un modèle (diagrammes de collaborations et de séquences, cas d'utilisations, ...).

Les MSC et les HMSC (high-level MSC) servent, dans le domaine de la conception des logiciels de télécommunications, à spécifier partiellement les propriétés d'un système. Certains détails du système sont omis de ces spécifications, par exemple en faisant abstraction des valeurs de certaines variables, ou des activités internes de certains sous-systèmes (composants), etc.

Cette vue préliminaire et incomplète du système projeté fait qu'il est indispensable de pouvoir valider les scénarios. Savoir transformer ces spécifications partielles en programmes (ou tout au moins squelettes de programmes) pouvant s'exécuter sur une architecture de déploiement est tout aussi indispensable, afin de rendre le processus de développement sûr et moins coûteux.

Jusqu'à présent, les études menées par les membres du projet [47,22,49] concernent principalement les propriétés fonctionnelles des scénarios : caractérisation de sous-ensembles de scénarios susceptibles d'être mis en oeuvre sur des architectures réparties par des transformations simples, calculs de clôtures de langages de scénarios dans les sous-ensembles précédents, décision d'une relation d'équivalence entre scénarios. Les résultats actuels sont

-
- [52] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, September 1993.
- [47] L. Hélouët and C. Jard. Conditions for synthesis of communicating automata from hmscs. In Axel Rennoch (Eds.) Stefania Gnesi, Ina Schieferdecker, editor, *5th International Workshop on Formal Methods for Industrial Critical Systems (FMICS)*, Berlin, avril 2000. GMD FOKUS. <http://www.gmd.de/publications/report/0091>.
- [22] B. Caillaud, P. Darondeau, L. Hélouët, and G. Lesventes. Hmscs as specifications... with pn as completions. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Proceedings of the summer school MOVEP'2k: Modelling and verification of parallel processes*, pages 87–103, Nantes, June 2000.
- [49] L. Hélouët and P. Le Maigat. Decomposition of Message Sequence Charts. In S. Graf, C. Jard, and Y. Lahav, editors, *Proceedings of the 2nd Workshop on SDL and MSC (SAM2000)*, pages 46–60, Col de Porte, Grenoble, France, 2000.

ponctuels : il est nécessaire de les relier afin de construire l'embryon d'une théorie cohérente sur laquelle puissent reposer des outils généraux et pérennes.

Les scénarios temporisés ont fait l'objet de travaux récents faisant usage de l'algèbre $(\max, +)$: il a été montré dans [17,16,44] que le comportement temporel de systèmes répartis modélisés à l'aide d'automates d'ordres partiels ou de HMSC peut s'étudier au travers d'automates à multiplicités sur le semi-anneau max-plus, ou, de manière équivalente, de séries rationnelles max-plus. Ceci permet de traduire certains problèmes d'évaluation et de vérification de performances en termes algébriques, et d'y apporter des solutions effectives.

3.3.2 Programme de recherche sur les langages de scénarios

Exécution et réalisation de HMSC en environnement réparti Une spécification sous forme de scénarios (MSC ou HMSC) peut ne pas être réalisable, sur l'architecture de déploiement visée pour son implémentation, en égard aux contraintes de concurrence, ou de non-localité des choix, exprimées de manière visuelle dans les HMSC. Nous chercherons à caractériser les sous-ensembles de HMSC réalisables, en fonction des modèles d'implémentation visés : automates communicants (SDL ou diagrammes d'états UML), state-charts de Harel [42], ou d'autres modèles comme les réseaux de Petri.

Ce problème de réalisabilité des scénarios a déjà reçu des solutions partielles [47,22], pour des HMSC et pour des modèles d'implémentation de même niveau d'abstraction. Nous étendrons notre effort à des HMSC et à des modèles d'implémentation de niveaux d'abstraction différents. En particulier, nous considérerons des HMSC possédant une hiérarchie de niveaux d'abstractions et nous étudierons leurs réalisations dans des modèles d'implémentation hiérarchisables (state-charts [42], BDL, sync-charts [5], réseaux de Petri hiérarchiques, etc).

-
- [17] A. Benveniste, C. Jard, and S. Gaubert. Algebraic techniques for timed systems. In *Proceedings of CONCUR'98*, Nice, France, September 1998.
 - [16] A. Benveniste, S. Gaubert, and C. Jard. Monotone rational series and max-plus algebraic models of real-time systems. In *Proc. of the Fourth Workshop on Discrete Event Systems (WODES98)*, Cagliari, Italy, 1998. IEE.
 - [44] L. Helouët and P. Le Maigat. A $(\max, +)$ approach for time in message sequence charts. In *Proc. of the Fifth Workshop on Discrete Event Systems (WODES00)*, Ghent, Belgium, 2000. IEE.
 - [42] D. Harel. Statecharts: A visual formulation for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.
 - [47] L. H elou et and C. Jard. Conditions for synthesis of communicating automata from hmscs. In Axel Rennoch (Eds.) Stefania Gnesi, Ina Schieferdecker, editor, *5th International Workshop on Formal Methods for Industrial Critical Systems (FMICS)*, Berlin, avril 2000. GMD FOKUS. <http://www.gmd.de/publications/report/0091>.
 - [22] B. Caillaud, P. Darondeau, L. H elou et, and G. Lesventes. Hmscs as specifications... with pn as completions. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Proceedings of the summer school MOVEP'2k: Modelling and verification of parallel processes*, pages 87–103, Nantes, June 2000.
 - [5] C. Andr e. Representation and analysis of reactive behaviors: A synchronous approach. In *CESA'96, IEEE-SMC*, Lille, France, July 1996. <http://www-sop.inria.fr/meije/esterel/syncCharts/TR96-28.pdf>.

Validation de HMSC Dans les applications des scénarios en télécommunications on utilise des bibliothèques de scénarios, représentant des comportements typiques du système («sunny day scenarios»), des comportements exceptionnels (anti-scénarios, «rainy day scenarios») ou même des comportements interdits. Du simple fait que les scénarios sont issus des multiples phases du développement d'un système (spécification, implémentation, test), et qu'ils portent sur des vues non nécessairement disjointes de celui-ci, vérifier leur cohérence est une question centrale.

Nous aborderons le problème de la validation d'un ensemble de scénarios, en vérifiant certaines propriétés de consistance afin de déterminer les parties cohérentes d'un ensemble de scénarios. Il faut cependant savoir que la plupart des problèmes de vérification sur les HMSC généraux sont indécidables ^[61,22] : égalité, inclusion ou vacuité de l'intersection des langages de HMSC, etc. Nous étudierons les classes de scénarios pour lesquels ces questions sont décidables et nous en chercherons des caractérisations effectives.

Analyse qualitative et quantitative de HMSC temporisés Nous proposons d'étendre les problématiques de validation et de réalisation présentées ci-dessus à des scénarios temporisés. Les travaux de thèse de L. Hérouët et P. Le Maigat ont permis de voir comment l'algèbre max-plus pouvait servir à analyser les dynamiques temporelles des HMSC. C'est ce travail que nous entendons développer, notamment dans les directions suivantes.

Il s'agit d'exploiter un progrès théorique récent et important : la généralisation des résultats de théorie spectrale max-plus au cas de dynamiques mélangeant plusieurs algèbres (faisant intervenir par exemple simultanément les lois max,min,+,x). Ces dynamiques permettent de modéliser des phénomènes de synchronisation limitée comme les "timeout", et plus généralement, de donner des estimations plus fines des temps de calcul.

Du point de vue de la spécification, on est amené à réfléchir sur la pertinence des diagrammes temporisés dans un cadre où la spécification est en général une spécification partielle du système. Cet aspect suggère en effet l'utilisation de contraintes temporelles symboliques, afin de pouvoir juste comparer la durée des actions spécifiées.

Fondements des langages de scénarios et nouveaux formalismes pour l'expression des exigences et des cas d'utilisations Dans son travail de doctorat, Loïc Hérouët a utilisé des structures d'événements régulières (définissables par grammaires de

-
- [61] A. Muscholl and D. Peled. Message sequence graphs and decision problems on Mazurkiewicz traces. In M. Kutyłowski and L. Pacholski, editors, *Proceedings of the 24th Symposium on Mathematical Foundations of Computer Science (MFCS'99)*, volume 1672 of *Lecture Notes in Computer Science*, pages 81–91, Szklarska Poreba, Poland, 1999. Springer.
- [22] B. Caillaud, P. Darondeau, L. Hérouët, and G. Lesventes. Hmscs as specifications... with pn as completions. In F. Cassez, C. Jard, B. Rozoy, and M. Ryan, editors, *Proceedings of the summer school MOVEP'2k: Modelling and verification of parallel processes*, pages 87–103, Nantes, June 2000.

graphes) comme modèle sémantique des HMSC [45,48]. Cette approche a permis d'obtenir un certain nombre de résultats sur la décomposition séquentielle et la simulation des HMSC ainsi que la décidabilité d'une équivalence comportementale sur les HMSC [49]. D'autres résultats sont encore à en attendre : problèmes de réalisabilité et de décomposition en produits des HMSC. Ces structures d'événements régulières permettent d'exprimer des familles de comportements répartis non définissables par HMSC — typiquement les comportements de protocoles de transport, avec “fenêtre glissante”. Il serait intéressant d'étendre l'expressivité des HMSC. Deux approches peuvent être envisagées : soit définir un produit de HMSC, qui permettrait de pouvoir exprimer les comportements de tout système d'automates communicants comme étant un produit de HMSC ; soit s'inspirer du mécanisme des grammaires de graphes, et d'introduire des arcs ou des sommets non terminaux dans les HMSC.

3.4 BDL : un formalisme comportemental synchrone pour la description des systèmes d'objets répartis

Participants : Albert Benveniste et Benoît Caillaud

Les développements théoriques, logiciels et expérimentaux de BDL se feront en commun avec le projet Espresso et comme BDL s'appuie fortement sur la notation UML, nous suivrons de près les travaux effectués sur UML dans le projet Triskell.

3.4.1 Contexte des travaux sur le langage BDL

Le langage BDL est un langage réactif synchrone issu du langage Signal, mais ayant la particularité d'être bien adapté à la spécification des comportements de systèmes d'objets répartis décrits en UML. Sa conception et son développement est le produit d'une coopération entre les anciens projets Pampa et Ep-Atr, soutenue par une collaboration avec Alcatel CRC à Marcoussis.

Le langage BDL trouve ses origines dans le constat suivant sur la notation UML : la notation UML comporte un grand nombre de diagrammes. Cependant, les ateliers de génie logiciel actuels n'offrent que des passerelles limitées entre ces formalismes ; ainsi il n'est actuellement pas possible de confronter des diagrammes de séquences avec la définition en StateCharts des comportements des objets qui composent un système. De même, tout changement d'architecture ou tout raffinement de diagrammes de séquences peut avoir un impact considérable sur les spécifications en StateCharts de ces objets.

[45] L. Hélouët. *Analyse des exigences des systèmes répartis exprimées par des langages de scénarios*. PhD thesis, Ecole doctorale MATISSE, Université de Rennes 1, Octobre 2000.

[48] L. Hélouët, C. Jard, and B. Caillaud. An event structure semantics for message sequence chart. *Mathematical Structures in Computer Science*, 2001. To appear.

[49] L. Hélouët and P. Le Maigat. Decomposition of Message Sequence Charts. In S. Graf, C. Jard, and Y. Lahav, editors, *Proceedings of the 2nd Workshop on SDL and MSC (SAM2000)*, pages 46–60, Col de Porte, Grenoble, France, 2000.

La production de code exécutable à partir de descriptions combinant plusieurs vue comportementales est largement manuelle : le code effectif à exécuter par un composant est généralement programmé “à la main”, en C, C++, ou Java. Il est inséré manuellement dans les squelettes de programme engendrés. De même, les communications ou appels de procédure entre composants sont réalisés par appels à une bibliothèque de services de communication fournis par un ORB support. Il est de la responsabilité du programmeur de s’assurer de la correction de ces appels bibliothèque vis-à-vis du service attendu. Un formalisme “colonne vertébrale” pouvant servir de pivot entre les vues comportementales d’UML est hautement désirable ; c’est l’ambition de BDL.

BDL est un langage qui combine à la fois les aspects hiérarchiques des SyncCharts ^[5] (lui même dérivé des StateCharts ^[42]) et certains éléments de la syntaxe des diagrammes de séquence UML (dérivé des MSC ^[52]). Chaque classe se présente comme la composition d’un ensemble de graphes orientés dont les sommets sont des événements valués. Ces graphes spécifient des contraintes d’ordonnement et d’activation sur les événements qui les composent. L’exécution d’une classe procède par réactions successives. Chaque réaction est provoquée par un ensemble d’événements provenant de l’environnement, pour ensuite comporter des événements à destination de l’environnement et transformer l’état de la classe. Deux classes se composent par unification de leurs réactions respectives. BDL récupère en fait la technologie développée pour les langages synchrones (Esterel, Lustre, et en l’occurrence plus particulièrement Signal), tout en offrant une notion d’abstraction appropriée. Le point nouveau, qui repose sur des résultats fondamentaux récents ^[15], est la possibilité d’interpréter une classe BDL indifféremment selon un mode synchrone ou asynchrone, avec la même faculté pour les communications entre classes. Ceci permet de décrire en BDL le comportements d’une classe importante de systèmes répartis. Il est aussi possible de coder en BDL la sémantique des diagrammes de séquences et d’états de la notation UML.

Une première implantation prototype de BDL a été réalisée, grâce à une réutilisation d’une partie de la chaîne de compilation du langage Signal. Les outils réalisés (compilateur, simulateur, vérificateur, traducteurs diagrammes de séquences (et HMSC) vers BDL et StateCharts vers BDL) manipulent directement une extension du méta-modèle (syntaxe abstraite) UML, dans lequel ont été ajoutés les éléments de syntaxe du langage BDL. Le simulateur et le vérificateur utilisent l’environnement Caesar/Aldebaran (CADP). Une interface sur la plate-forme UMLAUT ^[53] (développée dans le projet Triskell) existe égale-

[5] C. André. Representation and analysis of reactive behaviors: A synchronous approach. In *CESA’96, IEEE-SMC*, Lille, France, July 1996. <http://www-sop.inria.fr/meije/esterel/syncCharts/TR96-28.pdf>.

[42] D. Harel. Statecharts: A visual formulation for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.

[52] ITU-TS. *ITU-TS Recommendation Z.120: Message Sequence Chart (MSC)*. ITU-TS, Geneva, September 1993.

[15] A. Benveniste, B. Caillaud, and P. Le Guernic. Compositionality in dataflow synchronous languages: specification and distributed code generation. *Information and Computation*, 163:125–171, 2000.

[53] J.-M. Jézéquel and al. Umlaut: Unified modeling language all purposes transformer.

ment.

Nous avons étudié dans ^[14,15] les liens entre modèles synchrones et asynchrones. Des modèles simples ont été proposés pour ces deux paradigmes, ainsi qu'un ensemble de théorèmes permettant de garantir la correction de la désynchronisation d'un programme BDL isolé ou bien d'un réseau de programmes BDL. Ces théorèmes sont assortis d'hypothèses qui constituent des obligations de preuves vérifiables sur la composition synchrone des programmes considérés. Si ces conditions ne sont pas vérifiées, Il est possible d'ajouter à chacun des programmes BDL, un nouveau programme synchrone permettant d'assurer la correction de la désynchronisation. Ceci peut être considéré comme étant une méthode de génération de protocoles de synchronisation en environnement asynchrone., qui toutefois n'assure pas la minimalité du nombre des synchronisations à réaliser.

Le formalisme BDL permet de résoudre de façon simple et élégante le problème de l'adaptation de services de télécommunications par adjonction de nouveaux comportements décrits par une famille de scénarios ^[23]. Le problème étant alors de partir d'un modèle UML (diagramme de classe et State-Charts) décrivant le protocole ou le système de télécommunications et de mettre en oeuvre une méthode systématique pour y intégrer les comportements décrits par une famille de diagrammes de séquences. La méthode consiste à traduire en BDL l'ensemble des StateCharts et des diagrammes de séquences, pour ensuite les fusionner en un seul modèle BDL (grâce aux propriétés de compositionnalité du langage BDL). La dernière étape étant la désynchronisation et le déploiement de ce modèle BDL sur une architecture répartie.

3.4.2 Programme de recherche

Le programme de recherche proposé comporte quatre axes : le premier porte sur la consolidation des prototypes d'outils BDL développés à l'occasion de la collaboration Reutel (Alcatel CRC Marcoussis, 1998–2000). En particulier nous nous attacherons à améliorer et implanter l'algorithmique de désynchronisation de descriptions BDL publiée dans ^[14,15] ainsi que des algorithmes de hiérarchisation issus des travaux sur le langage Signal menés dans le projet Ep-Atr — environnement Polychronie, proposé par le projet Espresso. Les outils de traduction des diagrammes de séquences et des diagrammes d'états UML vers le langage BDL, déjà prototypés au cours de la collaboration Reutel, seront consolidés à l'occasion de cette nouvelle collaboration. Ce travail se fera en commun avec le projet Espresso, qui sera fournisseur de composants logiciels venant s'intégrer dans la plate-forme

<http://www.irisa.fr/UMLAUT/Welcome-fr.html>, mai 2000.

- [14] A. Benveniste, B. Caillaud, and P. Le Guernic. From synchrony to asynchrony. In J.C.M. Baeten and S. Mauw, editors, *CONCUR'99, Concurrency Theory, 10th International Conference*, volume 1664 of *Lecture Notes in Computer Science*, pages 162–177. Springer, August 1999.
- [15] A. Benveniste, B. Caillaud, and P. Le Guernic. Compositionality in dataflow synchronous languages: specification and distributed code generation. *Information and Computation*, 163:125–171, 2000.
- [23] Benoit Caillaud, Jean-Pierre Talpin, Jean-Marc Jézéquel, Albert Benveniste, and Claude Jard. Bdl: A semantics backbone for uml dynamic diagrams. Technical Report RR-4003, Irisa/INRIA Rennes, September 2000. <http://www.inria.fr/RRRT/RR-4003.html>.

BDL et assurant la hiérarchisation et une partie de l’algorithmique de désynchronisation de descriptions BDL. De fait, l’environnement BDL s’appuiera sur une plate-forme UML commune aux projets Triskell et S_4 , issue du prototype UMLAUT.

Le deuxième axe concerne la validation par l’expérience des techniques de désynchronisation préconisées. Ce travail de nature expérimental nous semble être une étape incontournable du transfert industriel de BDL et de la méthodologie qu’il sous-tend.

Le troisième axe porte sur le développement d’une application de BDL, utilisant les outils de désynchronisation et de hiérarchisation de spécifications BDL ainsi que les outils de traduction des diagrammes de séquences et d’états UML vers BDL : c’est l’adaptation de services de télécommunications^[23] — voir paragraphe 2.1.

Le dernier axe est un travail de nature théorique qui consiste à achever les études sur la désynchronisation de descriptions synchrones. Ce travail a déjà abouti à des publications^[14,15], mais se limite au modèle simple des systèmes de transitions synchrones, ou avec plus grande précision, aux traces de ces processus synchrones, sans se soucier ni des branchements, ni des conflits. Ce modèle n’est pas à proprement parler un modèle de la concurrence, et ne permet pas de représenter certains phénomènes classiques de la théorie de la concurrence. Le travail que nous entendons entreprendre est de transposer les résultats obtenus sur les systèmes de transitions synchrones au cadre d’un modèle de la concurrence reconnu : les structures d’événements premières. On peut espérer améliorer les résultats obtenus : affaiblissement des conditions suffisantes d’isochronie ; optimisation des synchronisations. A cette occasion, il serait intéressant de relier les questions de réalisabilité des scénarios, à la notion d’isochronie sur les systèmes de transitions synchrones.

3.4.3 Développements logiciels et transferts

Nous considérons a priori que BDL est notre effort principal en matière de tentative d’industrialisation et de transfert^[13].

Du point de vue de l’utilisateur, BDL se présentera comme suit :

- Un *plug-in* de l’outil hôte (Rose de Rational et Tau de Telelogic)
- Sans outil d’édition de dessin supplémentaire — BDL sera caché de l’utilisateur.
- Offrant des fonctionnalités de transformation de modèle.

Génération de descriptions BDL par compositions vues comportementales UML

On part de vues existantes, constituant diverses facettes du système à développer. Par des

-
- [23] Benoit Caillaud, Jean-Pierre Talpin, Jean-Marc Jézéquel, Albert Benveniste, and Claude Jard. Bdl: A semantics backbone for uml dynamic diagrams. Technical Report RR-4003, Irisa/INRIA Rennes, September 2000. <http://www.inria.fr/RRRT/RR-4003.html>.
- [14] A. Benveniste, B. Caillaud, and P. Le Guernic. From synchrony to asynchrony. In J.C.M. Baeten and S. Mauw, editors, *CONCUR’99, Concurrency Theory, 10th International Conference*, volume 1664 of *Lecture Notes in Computer Science*, pages 162–177. Springer, August 1999.
- [15] A. Benveniste, B. Caillaud, and P. Le Guernic. Compositionality in dataflow synchronous languages: specification and distributed code generation. *Information and Computation*, 163:125–171, 2000.
- [13] The bdl project, home page. <http://www.irisa.fr/pampa/bdl/>.

outils de traduction, on produit un modèle BDL global "équivalent" sémantiquement à l'ensemble des vues comportementales du système modélisé.

Un modèle comportemental BDL pourra toutefois être représenté et édité par l'intermédiaire de la combinaison de diagrammes d'état et de diagrammes de séquence UML. Pour définir un BDL, l'ensemble de ces diagrammes devra obéir à quelques restrictions syntaxiques simples.

Transformations Une fois disponible, une description BDL pourra subir les transformations suivantes :

Composition de descriptions hétérogènes BDL offre des mécanismes de composition de modèles par conjonction/disjonction d'exigences. Ceci permet en particulier de combiner des modèles préexistants sous forme de diagrammes d'états, à des exigences correspondant à de nouveaux services, décrits à l'aide de diagrammes de séquence (adaptation de service, paragraphe 2.1).

Déploiement BDL offre des facilités pour le déploiement assisté d'une application sur une architecture répartie. L'assistance porte sur la préservation du comportement lors de la mise en oeuvre des envois de messages à l'aide de l'architecture distribuée support.

Génération de code Plus précisément, il s'agit de produire et optimiser de manière fiable et fondée les structures de contrôle détaillées pour le code à exécuter.

Aide à la validation BDL étant sémantiquement fondé, il sera possible d'utiliser les services modernes de validation: simulation, vérifications, tests (automatiquement générés).

Exportation Il s'agit de réexporter du code produit par transformation vers des vues UML "exécutables" de type diagramme d'état.

Architecture de l'outil L'atelier UML hôte est considéré comme donné. Le développement proprement dit est constitué de l'ensemble des modules décrits ci-dessous :

Plug-in UML/BDL Sa fonction est de stocker les descriptions BDL, et d'assurer le lien et la cohérence avec les autres aspects de modélisation UML (fournis par l'hôte). Il assure l'interface entre l'atelier UML et les différents services BDL de base. Ce plug-in est constitué des éléments suivants :

- Une mise en oeuvre du méta-modèle UML de référence, étendue à l'aide de classes supplémentaires définissant BDL.
- Une interface entre ce méta-modèle et celui de l'atelier hôte : l'intégration des éléments BDL se fera par des mécanismes d'extension du méta-modèle de l'atelier hôte.
- L'activation des services BDL (point relativement mineur dans le développement).

Les services BDL de base – Edition/visualisation BDL par le biais de diagrammes d'état et de séquence obéissant à certaines règles syntaxiques.

- Traducteurs :
 1. Diagramme d'état vers BDL
 2. Diagramme de séquence vers BDL
 3. BDL vers diagramme d'état
- Transformations BDL2BDL et optimisations
- Vérification de propriétés comportementales (contrôle)
- Déploiement
- Génération de code de simulation instrumentée (model checking) et test
- Génération de code exécutable (C, Java)

3.5 Thèmes de recherches transversaux

3.5.1 Automates et logiques pour le contrôle de systèmes ouverts répartis

Participants : Sophie Pinchinat et Stéphane Riedweg

Contexte : logiques et automates Le problème de Church est un paradigme théorique de base pour la synthèse de programme. Proposé par Church dans [27] (voir aussi [71]), il pose le problème de la construction d'un transducteur de chaîne dont le comportement infini satisfait une formule de logique donnée, e.g. dans le calcul séquentiel SC de Büchi [58], c-à-d la théorie monadique du second ordre à un successeur, les logiques modales du temps linéaire comme PLTL.

Des solutions au problème de Church ont émergé de résultats fondamentaux sur les problèmes de décision les langages logiques associés. En particulier, la technique de synthèse d'un automate sur des objets infinis, tels que des mots ou des arbres selon que les logiques sont linéaires ou arborescentes [19,64], associé à une logique [38,74,65] et le test de vacuité du

-
- [27] A. Church. Logic, arithmetic and automata. In *Int. Congr. Math.*, pages 23–35, Stockholm, 1962.
 - [71] B. Trakhtenbrot and Y. Barzdin. *Finite automata: Behavior and synthesis*, volume 1 of *Fundamental studies in computer science*. North Holland, 1973.
 - [58] R. Mc Naughton. Büchi's sequential calculus. In S. MacLane and Siefkes D., editors, *The Collected Work of J. Richard Büchi*, pages 382–397. Springer-Verlag, 1990.
 - [19] J. R. Büchi. On a decision method in restricted second order arithmetic. In *Proc. 1960 Int. Congr. Logic, Methodology and Philosophy of Science*, pages 1–11. Stanford Univ. Press, 1962.
 - [64] M. O. Rabin. Decidability of second-order theories and automata on infinite trees. *Trans. Amer. Math. Soc.*, 141:1–35, 1969.
 - [38] E. A. Emerson and A. P. Sistla. Deciding full branching time logic. *Information and Control*, 61:175–201, 1984.
 - [74] M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logic of programs. *Journal of Computer and System Sciences*, 32:183–221, 1986.
 - [65] S. Safra. On the complexity of ω -automata. In *Proc. 29th IEEE Symp. Foundations of Computer Science*, pages 319–327, White Plains, October 1988.

langage qu'il dénote fournissent une méthode pour décider ces logiques ([70,35], [51,41,34,74,36,38,67,62]).

Les travaux issus des recherches de la théorie des automates ont eu des retombées très importantes sur les techniques de synthèse et de vérification ("model-checking") des programmes dits "non-terminants". Toutefois, pendant longtemps, l'approche par les automates pour le cas de la vérification de spécifications en logiques du temps arborescentes a souffert de complexités très élevées dans la construction de l'automate, laissant place à d'autres techniques [57,28,63,37]. Récemment, l'utilisation des automates alternés [60] ont permis de proposer un cadre "automate" satisfaisant pour les logiques arborescentes. Ces modèles généralisent la notion habituelle d'automates d'arbres non-déterministes. Alors que la construction d'un automate d'arbre classique pour une formule est de coût exponentiel (voire double exponentielle pour le cas de CTL* [38]), celle d'un automate alterné

-
- [70] W. Thomas. Infinite trees and automaton definable relations over ω -words. In *Proc. STACS 90*, volume 415 of *LNCS*, Rouen, February 1990. Springer-Verlag.
 - [35] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, volume B, chapter 16, pages 995–1072. Elsevier Science Publishers, 1990.
 - [51] R. Hossley and C. Rackoff. The emptiness problem for automata on infinite trees. In *13th Annual Symposium on Switching and Automata Theory*, pages 121–124, The University of Maryland, October 1972. IEEE.
 - [41] Y. Gurevich and L. Harrington. Trees, automata, and games. In *Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pages 60–65, San Francisco, California, May 1982.
 - [34] E. A. Emerson. Automata, tableaux and temporal logics. In *Proc. Logics of Programs Workshop*, volume 193 of *LNCS*, pages 79–88, Brooklyn, June 1985. Springer-Verlag.
 - [74] M. Y. Vardi and P. Wolper. Automata-theoretic techniques for modal logic of programs. *Journal of Computer and System Sciences*, 32:183–221, 1986.
 - [36] E. A. Emerson and C. S. Jutla. The complexity of tree automata and logics of programs. In *Proc. 29th IEEE Symp. Foundations of Computer Science*, pages 328–337, White Plains, October 1988. IEEE.
 - [38] E. A. Emerson and A. P. Sistla. Deciding full branching time logic. *Information and Control*, 61:175–201, 1984.
 - [67] R. S. Streett and E. A. Emerson. The propositional mu-calculus is elementary. In *Proc. 11th ICALP*, volume 172 of *LNCS*, pages 465–472, Antwerpen, 1984. Springer-Verlag.
 - [62] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Proc. 16th ACM Symp. Principles of Programming Languages*, pages 179–190, Austin, Texas, January 1989.
 - [57] O. Lichtenstein, A. Pnueli, and L. Zuck. The glory of the past. In *Proc. Logics of Programs Workshop*, volume 193 of *LNCS*, pages 196–218, Brooklyn, June 1985. Springer-Verlag.
 - [28] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, April 1986.
 - [63] J. P. Queille. The CESAR system: an aided design and certification system for distributed applications. In *Proc. of the 2nd Int. Conf. on Distributed Computing Systems*. Computer Society Press, 1981.
 - [37] E. A. Emerson, C. S. Jutla, and A. P. Sistla. On model-checking for fragments of mu-calculus. In *Proc. 5th International Computer Aided Verification Conference*, pages 385–396, 1993.
 - [60] D. E. Muller and P. E. Schupp. Alternating automata on infinite trees. *Theoretical Computer Science*, 54(2–3):267–276, October 1987.

est linéaire ^[59,1]. De plus, pour problème du model-checking il n'est plus nécessaire de tester la vacuité d'un automate d'arbre, mais seulement celle d'un automate de mots sur un alphabet de cardinal un, ce qui est substantiellement plus simple ^[18,56].

Systèmes ouverts et problème du contrôle Les domaines de la synthèse de programmes et de la vérification s'étendent naturellement à celui du contrôle. Le problème du contrôle consiste à contraindre un système de sorte que le comportement résultant satisfasse une propriété donnée (exprimée dans une logique par exemple). En général, le problème du contrôle se situe dans le cadre très général des systèmes ouverts, c'ad interagissant avec leur environnement (l'extérieur) et dont le comportement dépend cruciallement de cette interaction ^[43]. On peut citer par exemple le cas de CSP ^[50] ou des Reactives Modules ^[2,3]. Pour de tels systèmes, au delà des question "universelles" (est-ce que toutes les exécutions satisfont une propriété?) et "existentielles" (y a-t-il un exécution qui satisfait une propriété?), une troisième question apparaît naturellement : le système peut-il résoudre ses choix internes de sorte que la propriété soit garantie, et ceci indépendamment de la façon dont l'environnement résout les choix externes? On parle alors de propriétés *alternatives*. Pour le cas de telle propriétés, la synthèse de programme et le model-checking, appelé alors "module-checking", induit une augmentation très importante de la complexité des problèmes pour les logiques arborescentes par rapport au cas des systèmes "fermés" ^[55]. Cette constatation donne une indication sur la complexité en général élevée du problème du contrôle : par exemple, ^[55] montrent comment réduire le problème de la synthèse de programme à celui du contrôle.

-
- [59] D. E. Muller, A. Saoudi, and P. E. Schupp. Weak alternating automata give a simple explanation of why most temporal and dynamic logics are decidable in exponential time. In *Proc. 3rd IEEE Symp. Logic in Computer Science*, pages 422–427, Edinburgh, July 1988.
- [1] Emerson E. A. and C. S. Jutla. Tree automata, mu-calculus and determinacy. In *Proceedings 32nd Annual IEEE Symp. on Foundations of Computer Science, FOCS'91*, pages 368–377, San Jose, Puerto Rico, October 1991. IEEE Computer Society Press.
- [18] O. Bernholtz, M. Y. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. In D. L. Dill, editor, *Computer Aided Verification, Proc. 6th Int. Conference*, volume 818 of *Lecture Notes in Computer Science*, pages 142–155, Stanford, June 1994. Springer-Verlag.
- [56] O. Kupferman, M. Vardi, and P. Wolper. An automata-theoretic approach to branching-time model checking. *Journal of the ACM*, 47(2):312–360, March 2000.
- [43] D. Harel and D. Peleg. Process logic with regular formulas. *Theoretical Computer Science*, 38:307–322, 1985.
- [50] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice Hall Int., 1985.
- [2] R. Alur and T. A. Henzinger. Reactive modules. In *Proceedings, 11th Annual IEEE Symposium on Logic in Computer Science*, pages 207–218, New Brunswick, New Jersey, July 1996. IEEE Computer Society Press.
- [3] R. Alur and T. A. Henzinger. Reactive modules. *Formal Methods in System Design: An International Journal*, 15(1):7–48, July 1999.
- [55] O. Kupferman, P. Madhusudan, P. S. Thiagarajan, and M. Y. Vardi. Open systems in reactive environments: Control and synthesis. In *Proc. 11th Int. Conf. on Concurrency Theory*, volume 1877 of *Lecture Notes in Computer Science*, pages 92–107. Springer-Verlag, 2000.

Une autre manière de comprendre le point de vue des systèmes ouverts est de considérer qu'il existe des quantificateurs de chemins intermédiaires entre la quantification universelle et la quantification existentielle. Cette approche a été largement étudiée par [26].

Le problème du contrôle (sur des systèmes ouverts) peut être compris comme la mise au point d'une stratégie pour une condition de gain dans un jeu pour deux joueurs entre le système et l'environnement.

La plupart des travaux sur le problème du contrôle utilisent les techniques développées pour les systèmes fermés et reformulent leur sémantique comme des systèmes ouverts. Toutefois, [4] proposent d'enrichir la logique elle-même de sorte que les propriétés alternatives puissent être directement exprimées dans la logique. Les *systèmes de transition alternés* comme modèles généraux pour les systèmes ouverts sont introduits. Plusieurs logiques sont comparées (ATL, ATL*, une adaptation du mu-calcul et une logique de jeu) et leur model-checking est étudié.

Puisque dans une telle logique il est possible d'exprimer par une formule l'existence de choix internes du système en vue de garantir une propriété donnée, l'existence d'une solution au problème du contrôle est équivalente à la satisfaction de la formule par le système, et donc au model-checking de cette logique. De plus les systèmes de transitions alternés permettent de décrire des systèmes mettant à contribution plusieurs agents (deux et plus), ce qui offre un cadre mathématique au problème du contrôle réparti dans lequel plusieurs composants du système doivent coopérer

Objectifs de travail En général, le problème du contrôle passe par la donnée d'un système ouvert (formé éventuellement de plusieurs sous-systèmes) et d'une propriété à garantir par contrôle appelé *objectifs de contrôle*, dont la description peut prendre plusieurs formes : une formule d'une logique, un automate, etc. Dans un premier temps nous nous proposons de classifier les différents types de problèmes de contrôle sur la base de la structure de leur objectif de contrôle. En particulier, nous essaierons quand s'est possible de ramener les problèmes de décision pour l'existence d'une solution au contrôle à des problèmes tels que celui de la vacuité d'automates sur les objets infinis, et plus particulièrement à celle des automates alternés qui offrent des algorithmes efficaces/optimaux en complexité, ou encore à la satisfaisabilité ou le model-checking de logiques dédiées. Cette étude ne fournissant pas d'algorithme constructif pour définir la politique de contrôle nous aborderons dans un deuxième temps la construction effective de la stratégie de contrôle ces différentes catégories de problèmes.

Intégration dans le projet S4 Le langage de spécification offert par les HMSC fournit par essence même du formalisme une description partielle de systèmes. Nous nous nous

-
- [26] A. K. Chandra, D. C. Kozen, and L. J. Stockmeyer. Alternation. *Journal of the ACM*, 28(1):114–133, January 1981.
- [4] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. In W.-P. de Roever, H. Langmaack, and Pnueli A., editors, *Compositionality: The Significant Difference, COMPOS'97*, volume 1536 of *Lecture Notes in Computer Science*, pages 23–60, 1998.

proposons d'étudier des classes de réalisations des HMSC au moyen d'automates de mots ou d'arbres infinis, en prenant en compte autant que possible les informations de vrai parallélisme inhérentes à la spécification. En particulier, nos résultats au problème du contrôle permettraient de définir des réalisations de HMSC avec hypothèses de vivacité, d'équité, etc. dont on connaît l'importance pour représenter le parallélisme. Ces travaux offriront une direction d'attaque complémentaire à celles empruntées par les autres membres de l'équipe, e.g. par les réseaux de Petri.

3.5.2 Analyse quantitative de systèmes à l'aide d'algèbres tropicales

Participants : Albert Benveniste, Benoît Caillaud et Pierre Le Maigat

Les techniques et algorithmes sur des algèbres tropicales (algèbres $(\max,+)$ et aussi $(\min,\max,+,\times)$) permettent l'analyse des performances quantitatives de systèmes décrits dans des formalismes comme les HMSC [44], les réseaux de Petri (travail de doctorat en cours de Pierre Le Maigat). La caractéristique principale de ces techniques est qu'elles permettent le calcul symbolique des performances en régime asymptotique. Il nous semble intéressant de poursuivre l'application de l'analyse $(\max,+)$ à l'analyse de performance sur les langages et scénarios et les réseaux de Petri, mais aussi d'en étendre le champ d'application à d'autres formalismes comme BDL. Dans cette perspective nous sommes utilisateurs des techniques développées pour ces algèbres (théorie spectrale, etc) et n'avons pas l'intention d'y contribuer. C'est ainsi que nous envisageons la poursuite des collaborations avec Stéphane Gaubert du projet METALAU [17,16].

3.5.3 Modélisation de systèmes à l'aide de structures d'événements régulières

Les structures d'événements ont été introduites il y a une vingtaine d'années par Nielsen, Plotkin et Winskel [75]. Elles ont surtout été utilisées à des fins purement théoriques, pour relier entre eux différents modèles de la concurrence [66]. Plus récemment, elles ont été utilisées pour développer de nouvelles techniques de vérification, à l'aide de dépliages de

-
- [44] L. Helouët and P. Le Maigat. A $(\max,+)$ approach for time in message sequence charts. In *Proc. of the Fifth Workshop on Discrete Event Systems (WODES00)*, Ghent, Belgium, 2000. IEE.
 - [17] A. Benveniste, C. Jard, and S. Gaubert. Algebraic techniques for timed systems. In *Proceedings of CONCUR'98*, Nice, France, September 1998.
 - [16] A. Benveniste, S. Gaubert, and C. Jard. Monotone rational series and max-plus algebraic models of real-time systems. In *Proc. of the Fourth Workshop on Discrete Event Systems (WODES98)*, Cagliari, Italy, 1998. IEE.
 - [75] G. Winskel and M. Nielsen. Models for concurrency. In S. Abramsky, D. Gabbay, and T. S. E. Maibaum, editors, *Handbook of Logic in Computer Science*. Oxford University Press, 1995.
 - [66] Vladimiro Sassone, Mogens Nielsen, and Glynn Winskel. Models for concurrency: Towards a classification. *Theoretical Computer Science*, 170(1-2):297-348, 1996.

réseaux de Petri ^[29,39].

Dans son travail de doctorat, Loïc Hérouët a utilisé des structures d'événements comme modèle sémantique des HMSC ^[45,48]. Ces structures infinies sont en fait régulières et définissables par grammaires de graphes. Cette approche a permis d'obtenir un certain nombre de résultats sur la décomposition séquentielle et la simulation des HMSC et la décidabilité d'une équivalence comportamentale sur les HMSC ^[49]. D'autres résultats sont encore à en attendre : problèmes de réalisabilité et de décomposition en produits des HMSC.

Le modèle sous-jacent du formalisme BDL est celui des langages de *pomsets* (ensembles d'événements ordonnés étiquetés) et devrait pouvoir être étendu aux structures d'événements avec intérêt. Cela permettrait le transfert vers BDL de certains résultats obtenus sur les HMSC. Ce serait également une façon d'approfondir nos résultats sur la désynchronisation de processus synchrones ^[14,15] en prenant en compte les problèmes de branchements (conflits) et d'obtenir une théorie unifiée des comportements synchrones et asynchrones.

4 Positionnement vis-à-vis des autres projets du programme 1c

Espresso :

Même si le développement du langage BDL se fait et se fera en commun, les apports de chacun des projets seront différents de nature : Espresso apportera ses compétences en matière de vérification, de transformation et de compilation de descriptions synchrones ainsi que des techniques issues de l'analyse statique de programme. Le projet S_4 apportera

-
- [29] Edmund M. Clarke, Jeannette M. Wing, Rajeev Alur, Rance Cleaveland, David Dill, Allen Emerson, Stephen Garland, Steven German, John Guttag, Anthony Hall, Thomas Henzinger, Gerard Holzmann, Cliff Jones, Robert Kurshan, Nancy Leveson, Kenneth McMillan, J. Moore, Doron Peled, Amir Pnueli, John Rushby, Natarajan Shankar, Joseph Sifakis, Prasad Sistla, Bernhard Steffen, Pierre Wolper, Jim Woodcock, and Pamela Zave. Formal methods: state of the art and future directions. *ACM Computing Surveys*, 28(4):626–643, 1996.
 - [39] Javier Esparza and Stefan Romer. An unfolding algorithm for synchronous products of transition systems. In *International Conference on Concurrency Theory*, pages 2–20, 1999.
 - [45] L. Hérouët. *Analyse des exigences des systèmes répartis exprimées par des langages de scénarios*. PhD thesis, Ecole doctorale MATISSE, Université de Rennes 1, Octobre 2000.
 - [48] L. Hérouët, C. Jard, and B. Caillaud. An event structure semantics for message sequence chart. *Mathematical Structures in Computer Science*, 2001. To appear.
 - [49] L. Hérouët and P. Le Maigat. Decomposition of Message Sequence Charts. In S. Graf, C. Jard, and Y. Lahav, editors, *Proceedings of the 2nd Workshop on SDL and MSC (SAM2000)*, pages 46–60, Col de Porte, Grenoble, France, 2000.
 - [14] A. Benveniste, B. Caillaud, and P. Le Guernic. From synchrony to asynchrony. In J.C.M. Baeten and S. Mauw, editors, *CONCUR '99, Concurrency Theory, 10th International Conference*, volume 1664 of *Lecture Notes in Computer Science*, pages 162–177. Springer, August 1999.
 - [15] A. Benveniste, B. Caillaud, and P. Le Guernic. Compositionality in dataflow synchronous languages: specification and distributed code generation. *Information and Computation*, 163:125–171, 2000.

ses compétences en modèles de la concurrence, techniques de désynchronisation et de répartition.

Vertecs :

Les deux projets s'intéressent au contrôle de supervision des systèmes à événements discrets. L'approche envisagée dans Vertecs est fondée sur des systèmes de transitions représentés explicitement ou symboliquement (automates étendus). Nous avons une approche fondée sur des modèles concurrents (réseaux de Petri, structures d'événements). L'objectif n'est pas le même : nous visons avant tout la synthèse de contrôleurs répartis.

En ce qui concerne la génération de cas de test pour des architectures réparties, le projet Vertecs serait utilisateur de techniques de répartition développées dans S_4 .

Quant à S_4 , nous serions utilisateurs de techniques et d'outils symboliques développés dans le projet Vertecs : cela permettrait d'améliorer les procédures de décision sur les descriptions BDL en restreignant l'ensemble des états à ceux satisfaisant, par exemple, des invariants inductifs.

Triskell :

UML serait le médium commun du transfert d'une partie de nos travaux respectifs. Toutefois, les contributions des deux projets dans le domaine de la modélisation objet ne seront pas de même nature. Nos travaux s'attacheraient à la transformation de descriptions hétérogènes des comportements : scénarios (sequence diagrams) vers automates communicants (statecharts), par l'intermédiaire de BDL. Le projet Triskell serait éventuellement utilisateur de ces travaux, mais n'a pas l'ambition d'y contribuer. Ceci dit il y a matière à collaboration : 1/ l'atelier UMLAUT sera l'environnement d'accueil des prototypes réalisés pour BDL ; 2/ certaines questions concernant la sémantique de UML, en l'occurrence l'*action semantics*, trouvent des réponses claires et concises quand on adopte BDL comme formalisme comportemental de base ; 3/ certains résultats sur BDL et sur les HMSC pourraient être utilisés pour générer des contrats de composants logiciels à partir de descriptions de plus haut niveau.

plate-forme d'expérimentation commune à plusieurs projets

Nous appelons de nos vœux une mise en commun, entre plusieurs projets, de certains efforts de développement logiciel. Ainsi la plate-forme UMLAUT a vocation à être utilisée par plusieurs projets : Triskell de toute évidence, S_4 pour les scénarios et BDL, Espresso pour la hiérarchisation de descriptions BDL et Vertecs, dans une certaine mesure. La pérennisation de la plate-forme UMLAUT est donc un élément important du succès de ces projets.

Autres projets INRIA du thème 1, pouvant avoir des liens avec le projet S_4

Le domaine d'application privilégié du projet S_4 est celui des logiciels de télécommunications. La modélisation de ces logiciels requiert parfois de pouvoir créer dynamiquement de nouveaux processus et de pouvoir reconfigurer en cours d'exécution le graphe des communications entre processus. Instanciation dynamique, reconfiguration et mobilité sont des sujets qui ont fait l'objet de recherches spécifiques depuis plus de dix ans. Des modèles de la concurrence adaptés ont été proposés : le pi-calcul, le join-calcul ou les réseaux de Petri auto-modifiants en sont quelques exemple. Deux projets de recherche du thème 1c travaillent spécifiquement sur ces sujets, ce sont les projets MOSCOVA (mobilité, sécurité, concurrence, vérification et analyse) et MIMOSA (migration et mobilité : sémantique et applications). Nous n'avons pas l'intention de contribuer à ce sujet, ni même d'être utilisateurs de tels modèles. Nos études de cas se limiteront aux applications qui ne nécessitent pas toute la puissance d'expression des modèles cités plus haut. Au contraire, nous souhaitons nous concentrer sur quelques modèles simples (les réseaux de Petri, les MSC et les processus synchrones configurés statiquement, pour lesquels les méthodes peuvent être bien plus automatisées que pour les modèles de la mobilité.

Les objectifs scientifiques du projet TICK (étude et implémentation des systèmes réactifs synchrones) sont assez voisins de ceux du projet ESPRESSO, avec toutefois un fort accent mis sur la conception conjointe de logiciel et de matériel (*co-design*). Le sujet de la collaboration en cours avec le projet ESPRESSO (L'étude de la désynchronisation de programmes réactifs synchrones) n'est pas prioritaire pour le projet TICK. Enfin, la proximité géographique du projet ESPRESSO et les collaborations passées avec l'ancien projet EP-ATR font que nous collaborerons fréquemment avec le projet ESPRESSO et probablement peu avec le projet TICK.

Le quatrième projet INRIA dont les travaux de recherche recouvrent certains sujets du projet S_4 est le projet TRIO de l'INRIA Lorraine. Nous partageons à la fois certains objectifs (par exemple le contrôle de supervision) et certains modèles (modèles temporisés à l'aide d'algèbres tropicales, réseaux de Petri). Des collaborations sont envisageables sur plusieurs de ces sujets. Les domaines d'applications sont toutefois assez différents : nous nous concentrerons en priorité sur les logiciels de télécommunications et sur les questions de synthèse de logiciel réparti, alors que le projet TRIO a privilégié jusqu'à présent les systèmes de production ou de télésurveillance et les réseaux de télécommunications, mais là uniquement pour des questions d'analyse de performance et de dimensionnement.

5 Relations industrielles et académiques

5.1 Insertion dans la communauté scientifique

Philippe Darondeau et Benoît Caillaud participent la Collaboration Catalysis, gérée par le CNRS et qui permet des échanges avec Marek Bednarczyk et son équipe, de l'IIPAN,

Académie des sciences de Pologne, Gdansk, Pologne. Les thèmes de recherche abordés sont l'étude des structures algébriques de modèles de la concurrence, comme les réseaux de Petri, les systèmes de transitions asynchrones, etc.

Philippe Darondeau, Benoît Caillaud et Gilles Lesventes participent à l'ARC MARS¹, dirigée par Xiaolan Xie (LORIA), dont le thème est le contrôle de systèmes discrets à l'aide de réseaux de Petri. Notre contribution porte sur l'utilisation de techniques de synthèse de réseaux (distribuables) pour générer des contrôleurs (répartis asynchrones) de systèmes à événements discrets.

Nous participons tous à l'ARC FISC², coordonnée par Benoît Caillaud, dont la thématique est la formalisation et l'instrumentation des scénarios. Les partenaires sont le LIAFA (université Paris 7), et les projets METALAU et BIP de l'INRIA.

5.2 Relations industrielles

Nous entretenons des contacts suivis avec France Télécom Recherche et Développement à Lannion. Une collaboration avec eux, d'une durée de deux ans viens juste de commencer (11 septembre 2001 – 10 septembre 2003). Elle porte sur le développement d'outils et de méthodes d'analyse et de transformation d'exigences exprimées à l'aide de HMSC.

Nous entretenons de longue date des contacts avec Alcatel CRC à Marcoussis. Nous avons eu avec eux des collaborations depuis 1997, la dernière en date, la collaboration Reutel 2000, a été achevée à la fin de l'année 2000. Elle portait sur plusieurs thèmes (optimisation de code par évaluation partielle; architecture et algorithmes pour la tolérance aux défaillances; et enfin notre contribution, détaillée ci-après) et a sollicité les moyens de plusieurs projets de recherche de l'IRISA (projets Pampa, Ep-Atr, Compose et Adp). Quant à nous, nous y avons contribué en développant un langage, une méthodologie et des outils pour la modélisation de systèmes d'objets réactifs et répartis, à l'aide de descriptions synchrones. Nous avons développé le langage réactif synchrone BDL, dérivé du langage Signal et qui permet, à l'aide de schémas de traductions, de combiner entre elles les différentes vue comportementales de la notation UML. Le travail réalisé est détaillé au paragraphe 3.4.1.

6 Enseignement, formation, animation scientifique

Gilles Lesventes, enseigne un module optionnel de la maîtrise d'informatique de l'université de Rennes 1, qui comporte une initiation aux processus concurrents et aux réseaux de Petri.

Benoît Caillaud a été responsable pendant deux ans de la filière "génie logiciel et méthodes formelles" du DEA d'informatique de l'IFSIC, université de Rennes 1. Il participe, en collaboration avec Sophie Pinchinat, à l'enseignement d'un module du DEA, sur les logiques et les modèles de la concurrence.

¹MARS : <http://www.loria.fr/xie/Mars.html>

²FISC : <http://www.irisa.fr/pampa/arc-fisc/>

Benoît Caillaud est responsable de l'ARC FISC sur la formalisation et l'instrumentation des langages de scénarios.

Pour l'année 2001, Philippe Darondeau et Benoît Caillaud sont organisateurs de *workshops* sur la synthèse de systèmes concurrents (Synthesis of Concurrent Systems, satellite de ICATPN'01³) et le contrôle de systèmes à événements discrets (SCODES'2001⁴, affilié à CAV'01).

7 Bibliographie des membres du projet

Cette bibliographie regroupe les publications des permanents du projet depuis 1998.

Livres et monographies

- [1] B. CAILLAUD, P. DARONDEAU, L. LAVAGNO, X. XIE (ed.), *Synthesis and Control of Discrete Event Systems*, Kluwer Academic Press, 2001, To appear.
- [2] B. CAILLAUD, X. XIE (ed.), *Proceedings of the Symposium on the Supervisory Control of Discrete Event Systems*, INRIA, Paris, July 2001.
- [3] P. DARONDEAU, L. LAVAGNO (ed.), *Proceedings of the Workshop on Synthesis of Concurrent Systems*, University of Newcastle, Newcastle upon Tyne, UK, June 2001.

Articles

- [4] E. BADOUEL, P. DARONDEAU, B. CAILLAUD, « Distributing Finite Automata through Petri Net Synthesis », *Journal on Fundamental Aspects of Computer Science*, 2001, to appear.
- [5] E. BADOUEL, P. DARONDEAU, J.-C. RAOULT, « Context-Free Event Domains are Recognizable », *Information and Computation* 149, 2, 1999, p. 134–172.
- [6] A. BENVENISTE, B. CAILLAUD, P. LE GUERNIC, « Compositionality in dataflow synchronous languages: specification and distributed code generation », *Information and Computation* 163, 2000, p. 125–171.
- [7] A. BENVENISTE, « Compositional and Uniform Modelling of Hybrid Systems », *IEEE Transactions on Automatic Control* 43, 4, April 1998.
- [8] B. CAILLAUD, « Bounded Petri-net synthesis techniques and their applications to the distribution of reactive automata », *JESA, European Journal on Automated Systems* 33, 8–9, 1999, p. 925–942.
- [9] P. DARONDEAU, « On the Petri net realization of context-free graphs », *Theoretical Computer Science* 258, 1–2, 2001, p. 573–598.
- [10] L. HÉLOUËT, C. JARD, B. CAILLAUD, « An Event Structure Semantics for Message Sequence Chart », *Mathematical Structures in Computer Science*, 2001, To appear.

³ICATPN'01 : <http://www.cs.ncl.ac.uk/conferences/2001/pn/>

⁴SCODES'01 : <http://www.irisa.fr/pampa/scodes2001/index.html>

Chapitres de livre

- [11] E. BADOUEL, P. DARONDEAU, « Theory of regions », *in: Lectures on Petri Nets I: Basic Models, LNCS, 1491*, Springer, 1999, p. 529–586.
- [12] B. CAILLAUD, P. DARONDEAU, L. HÉLOUËT, G. LESVENTES, « HMSCs as specifications... with PN as completions », *in: Modeling and Verification of Parallel Processes*, F. Cassez, C. Jard, B. Rozoy, et M. Dermot (ed.), *Lecture Notes in Computer Science, 2067*, Springer, 2001, p. 125–152.

Communications à des manifestations scientifiques

- [13] E. BADOUEL, P. DARONDEAU, A. TOKMAKOFF, « Modelling Dynamic Agents Systems with Cooperating Automata », *in: PDPTA'99*, CSREA Press, p. 11–17, Las Vegas, USA, 1999.
- [14] M. BEDNARCZYK, P. DARONDEAU, « Looking for diamonds », *in: Proceedings of the Workshop on Synthesis of Concurrent Systems*, P. Darondeau, L. Lavagno (éd.), University of Newcastle, p. 11–18, Newcastle upon Tyne, UK, June 2001.
- [15] A. BENVENISTE, B. CAILLAUD, P. LE GUERNIC, « From Synchrony to Asynchrony », *in: CONCUR'99, Concurrency Theory, 10th International Conference*, J. Baeten, S. Mauw (éd.), *Lecture Notes in Computer Science, 1664*, Springer, p. 162–177, August 1999.
- [16] A. BENVENISTE, S. GAUBERT, C. JARD, « Monotone rational series and max-plus algebraic models of real-time systems », *in: Proc. of the Fourth Workshop on Discrete Event Systems (WODES98)*, IEE, Cagliari, Italy, 1998.
- [17] A. BENVENISTE, C. JARD, S. GAUBERT, « Algebraic techniques for timed systems », *in: Proceedings of CONCUR'98*, Nice, France, September 1998.
- [18] A. BENVENISTE, M. SIEGEL, L. HOLENDERSKI, K. WINKELMANN, E. SEFTON, E. RUTTEN, P. LE GUERNIC, T. GAUTIER, « Safety Critical Embedded Systems Design: the SACRES approach », *in: Formal Techniques in Real-Time and Fault Tolerant systems, FTRTFT'98 school*, Lyngby, Denmark, September 1998.
- [19] A. BEVENISTE, « Synchronous languages and reactive system design », *in: Proceedings of the 9th IFAC-INCOM'98*, Nancy, France, June 1998.
- [20] B. CAILLAUD, P. DARONDEAU, L. HÉLOUËT, G. LESVENTES, « HMSCs as specifications... with PN as completions », *in: Proceedings of the summer school MOVEP'2k: Modelling and verification of parallel processes*, F. Cassez, C. Jard, B. Rozoy, M. Ryan (éd.), p. 87–103, Nantes, June 2000.
- [21] P. DARONDEAU, « Deriving unbounded nets from formal languages », *in: CONCUR'98, LNCS, 1466*, Springer, p. 533–548, 1998.
- [22] P. DARONDEAU, « Region Based Synthesis of P/T-Nets and Its Potential Applications », *in: Proceedings of the 21st International Conference on Application and Theory of Petri Nets, ICATPN 2000, Lecture Notes in Computer Science, 1825*, Springer, Aarhus, Denmark, June 2000. Invited paper.
- [23] O. KUSHNARENKO, S. PINCHINAT, « Intensional Approaches for Symbolic Methods », *in: Electronic Notes in Theoretical Computer Science*, P. Jancar, M. Kretinsky (éd.), 18, Elsevier Science Publishers, 2000.

- [24] H. MARCHAND, S. PINCHINAT, « Supervisory Control Problem using Symbolic Bisimulation Techniques », *in: 2000 American Control Conference*, p. 4067–4071, Chicago, Illinois, USA, June 2000.
- [25] S. PINCHINAT, H. MARCHAND, « Symbolic Abstractions of Automata », *in: Proc of 5th Workshop on Discrete Event Systems, WODES 2000*, p. 39–48, Ghent, Belgium, August 2000.
- [26] S. PINCHINAT, E. RUTTEN, R. K. SHYAMASUNDAR, « Taxonomy and Expressiveness of Preemption: A syntactic approach », *in: Advances in Computing Science - ASIAN'98*, A. O. J. Hsiang (éd.), p. 111–125, Manila, The Philippines, December 1998.
- [27] J.-P. TALPIN, A. BENVENISTE, B. CAILLAUD, C. JARD, Z. BOUZIANE, H. CANON, « BDL, a Language of Distributed Reactive Objects », *in: ISORC'98, The 1st IEEE International Symposium on Object-oriented Real-time Distributed Computing*, Kyoto, Japan, April 1998.
- [28] A. TURODET, S. NADJM-TEHRANI, A. BENVENISTE, J. E. STRÖMBERG, « Co-simulation of Hybrid Systems: Signal-Simulink », *in: Proceedings of 6th international school and symposium on Formal Techniques in Real-time and Fault-tolerant Systems, Lecture Notes in Computer Science*, Springer, September 2000.
- [29] Y. WANG, J.-P. TALPIN, A. BENVENISTE, P. LE GUERNIC, « Compilation and distribution of state machines using Spots », *in: 16th IFIP World Computer Congress (WCC'2000)*, August 2000.
- [30] Y. WANG, J.-P. TALPIN, A. BENVENISTE, P. LE GUERNIC, « A semantics of UML state-machines using synchronous pre-order transition systems », *in: International Symposium on Object-Oriented Real-Time Distributed Computing (ISORC'2000)*, IEEE Press, March 2000.

Rapports de recherche

- [31] E. BADOUEL, P. DARONDEAU, D. QUICHAUD, A. TOKMAKOFF, « Modelling Dynamic Agents Systems with Cooperating Automata », publication interne n° PI-1253, IRISA, June 1999.
- [32] A. BENVENISTE, B. CAILLAUD, P. LE GUERNIC, « From synchrony to asynchrony », Rapport de Recherche n° RR-3641, INRIA Rennes, March 1999, Also published as IRISA Research Report PI-1233.
- [33] A. BENVENISTE, P. CASPI, « Distributing synchronous programs on a loosely synchronous, distributed architecture », Rapport de Recherche n° 1289, IRISA, December 1999.
- [34] B. CAILLAUD, P. DARONDEAU, L. HÉLOUËT, G. LESVENTES, « HMSCs as specifications... with PN as completions », Rapport de recherche n° RR-3970, INRIA Rennes, July 2000.
- [35] B. CAILLAUD, J.-P. TALPIN, J.-M. JÉZÉQUEL, A. BENVENISTE, C. JARD, « BDL: A Semantics Backbone for UML Dynamic Diagrams », rapport de recherche n° RR-4003, Irisa/INRIA Rennes, September 2000, <http://www.inria.fr/RRRT/RR-4003.html>.
- [36] P. DARONDEAU, V. SCHMITT, « State Graphs of Stratified Flip-Flop Nets », publication interne n° PI-1177, IRISA, March 1998.
- [37] P. DARONDEAU, « Deriving Unbounded Petri Nets from Formal Languages », Rapport de recherche n° RR-3365, INRIA Rennes, Février 1998.

- [38] P. DARONDEAU, « On the Petri Net Realization of Context-Free Graphs », Rapport de recherche n° RR-3674, INRIA Rennes, Mai 1999.
- [39] L. HÉLOUËT, C. JARD, B. CAILLAUD, « An Effective Equivalence for Sets of Scenarios Represented by HMSCs », Rapport de recherche n° RR-3499, INRIA Rennes, septembre 1998, Also published as IRISA Research Report PI-1205.
- [40] O. KUSHNARENKO, S. PINCHINAT, « Intensional Approaches for Symbolic Methods », Rapport de recherche n° 3448, Institut National de Recherche en Informatique et en Automatique (INRIA), July 1998.
- [41] O. KUSHNARENKO, S. PINCHINAT, « Intensional Approaches for Symbolic Methods », Rapport de recherche n° 3448, Institut National de Recherche en Informatique et en Automatique (INRIA), July 1998.
- [42] S. PINCHINAT, H. MARCHAND, M. LE BORGNE, « Symbolic Abstractions of Automata and their application to the Supervisory Control Problem Research », rapport de recherche n° 1279, Irisa, November 1999.
- [43] J.-P. TALPIN, A. BENVENISTE, B. CAILLAUD, C. JARD, Z. BOUZIANE, H. CANON, « BDL, a language of distributed reactive objects », Rapport de recherche n° RR-3353, Inria-Rennes, February 1998, Also published as IRISA Research Report PI-1145.
- [44] J.-P. TALPIN, A. BENVENISTE, B. CAILLAUD, P. LE GUERNIC, « Hierarchic Normal Forms for desynchronization », Rapport de Recherche n° 1288, IRISA, December 1999.
- [45] J.-P. TALPIN, A. BENVENISTE, B. CAILLAUD, P. LE GUERNIC, « Hierarchic Normal Forms for Desynchronization », rapport de recherche n° RR-3822, Irisa/INRIA Rennes, décembre 1999 1999, <http://www.inria.fr/RRRT/RR-3822.html>.
- [46] J.-P. TALPIN, A. BENVENISTE, P. LE GUERNIC, « Asynchronous deployment of synchronous transition systems », Rapport de Recherche n° 1269, IRISA, October 1999.
- [47] Y. WANG, J.-P. TALPIN, A. BENVENISTE, P. LE GUERNIC, « Pre-order semantics of UML state machines », Rapport de Recherche n° 1336, IRISA, June 2000.

Divers

- [48] O. KUSHNARENKO, S. PINCHINAT, « Labeling Automata with Polynomials », 10th European Summer School in Logic, Language and Information, 1998.