Modélisation polychrone et évaluation de systèmes temps réel

Abdoulaye Gamatié

N° d'ordre: 2881

THÈSE

présentée

devant l'université de Rennes 1

pour obtenir

le grade de : Docteur de l'université de Rennes 1
Mention Informatique

par

Abdoulaye El Hadji Gamatié

Équipe d'accueil : Espresso
École doctorale : MATISSE
Composante universitaire : IFSIC / IRISA

Titre de la thèse :

Modélisation polychrone et évaluation de systèmes temps réel

Soutenue le 25 Mai 2004 devant la commission d'examen

MM. :	Jean-Pierre	Elloy	Rapporteur
	Patrick	Farail	Examinateur
	Thierry	Gautier	Examinateur
	Paul	Le Guernic	Directeur de thèse
	Michel	Raynal	Président
	Yves	Sorel	Rapporteur

À la mémoire de Boubakar,
À ma mère.

Remerciements

Le travail présenté dans ce document a été réalisé à l'Institut de Recherche en Informatique et Systèmes Aléatoires (Irisa), au sein des équipes Environnement de Programmation pour Applications Temps-Réel (Ep-atr) puis Environnement de Spécification de Programmes Réactifs Synchrones (Espresso).

En premier lieu, je tiens à remercier les membres du jury qui ont accepté de juger mon travail

- Michel Raynal, Professeur à l'Université de Rennes 1, qui a accepté de présider le jury ;
- Jean-Pierre Elloy, Professeur à l'École Centrale de Nantes, et Yves Sorel, Directeur de Recherche à l'Inria, tous deux rapporteurs de ma thèse, qui ont lu et évalué mon travail ;
- Patrick Farail, Responsable du support méthode de développement logiciel embarqué à Airbus France, qui a bien voulu juger mon travail.

J'exprime également toute ma gratitude à Paul Le Guernic, pour m'avoir accueilli dans l'équipe, et m'avoir offert l'opportunité de travailler sur un sujet à la fois intéressant et plein de défis. Je suis très sensible à la confiance qu'il m'a accordée durant ces années pour mener à bien ce travail. Ses conseils m'ont été très précieux. Finalement, ce travail n'aurait sans doute pas été dans sa forme actuelle sans l'aide de Thierry Gautier, au contact de qui j'ai énormément appris. Je lui suis très reconnaissant pour sa grande disponibilité, pour son avis très critique sur ce travail, pour sa patience dans la lecture très attentive de ce document. Il n'a pas cessé de m'encourager durant ces années de thèse. Enfin, j'ai été particulièrement touché par ses qualités humaines.

Je remercie très chaleureusement tous les membres des équipes Ep-atr et Espresso pour l'ambiance agréable et détendue qu'ils ont toujours su faire régner, pour leurs conseils et leurs encouragements. Je remercie en particulier Loïc pour ses inestimables conseils pratiques, et Jean-Pierre pour sa collaboration fructueuse.

Ne pouvant remercier individuellement tous mes collègues et amis, anciens et nouveaux, qui n'ont cessé de me soutenir jusqu'au bout, je voudrais simplement leur adresser mes sincères remerciements.

Je remercie toute ma famille pour sa présence permanente à mes cotés, sans laquelle je n'aurais pas pu terminer cette thèse.

Enfin, il serait injuste de ma part de ne pas exprimer ma gratitude envers tous ceux qui m'ont permis d'arriver à ce stade de mes études : à ma famille, à mes enseignants, au système universitaire français, et à toutes les personnes qui ont cru en moi.

Table des matières

Introduction

Les systèmes réactifs temps réel

Les *systèmes réactifs* [HP85] [Ber89] sont des systèmes qui interagissent de manière continue avec l'environnement physique auquel ils sont connectés. Ils reçoivent de celui-ci des flots d'informations sur leurs ports d'entrée, effectuent les traitements requis, et produisent des flots d'informations sur leurs ports de sortie. On rencontre de tels systèmes dans des applications comme le traitement du signal, la gestion des réseaux de communication ou le contrôle de processus industriels.

Il convient de distinguer les systèmes *réactifs* des systèmes *interactifs et transformationnels* qui constituent les deux autres familles de systèmes informatiques habituellement mentionnées. Dans le fonctionnement d'un système réactif, c'est l'environnement qui fixe les instants de réaction du système. Cela n'est pas le cas pour un système interactif où les interactions avec l'environnement se déroulent au rythme du système. Les systèmes d'interrogation de bases de données et les systèmes d'exploitation en sont des exemples types. Quant au système transformationnel, il requiert des données d'entrée dès son initialisation et ne fournit des données en sortie qu'après sa terminaison. C'est le cas d'un solveur d'équations ou d'un compilateur, où finalement l'unique contrainte liée au temps est qu'ils doivent se terminer en un temps fini.

Parmi les caractéristiques des systèmes réactifs [Hal93], il convient de souligner les suivantes :

- *La concurrence.* L'évolution du système se fait de façon concurrente avec son environnement. Il est naturel de voir un tel système comme un ensemble de composants parallèles qui coopèrent pour réaliser la tâche assignée au système entier. Ainsi, le système comprend au moins deux types de processus coopérants qui assurent d'une part l'échange des données et d'autre part le traitement de celles-ci.
- *Les contraintes temporelles.* Les systèmes réactifs sont soumis à des contraintes temporelles fortes (temps de réponse borné, fréquence d'échantillonnage d'un signal, etc.), spécifiées lors de la conception. La vérification de celles-ci avant utilisation est nécessaire pour la validation du système. On parle dans ce cas de **systèmes temps réel**.
- *La sûreté.* Du fait de leurs divers rôles **critiques** (par exemple, le contrôle d'une centrale nucléaire ou bien celui du train d'atterrissage d'un avion) impliquant des enjeux importants aussi bien en vies humaines qu'en termes économiques, il est primordial d'éviter tout risque de dysfonctionnement. La sûreté de fonctionnement devient de fait une question essentielle pour ces systèmes. Cela entraîne un besoin de méthodes de développement qui intègrent des outils et techniques adéquats pour la vérification et l'analyse.
- *Le déterminisme.* C'est une propriété fortement souhaitée dans les systèmes temps réel. Elle facilite la prédiction sur les comportements d'un système. Elle permet aussi de reproduire des comportements d'un système dans l'optique, par exemple, de mettre en évidence des erreurs

susceptibles de survenir lors d'une exécution de celui-ci.
- *La maintenance.* Ils sont parfois **embarqués** (dits aussi **enfouis** de façon équivalente dans le reste de ce document), posant ainsi quelques difficultés pour les modifier après la réalisation. Dans ce cas, une stratégie prédictive de maintenance, basée sur des modèles *a priori*, peut jouer un rôle important dans le choix d'une structure de système facile à modifier. Les systèmes embarqués sont surtout caractérisés par des contraintes matérielles (ressources limitées en espace mémoire et puissance de calcul).
- *La répartition.* Ce sont souvent des systèmes géographiquement **distribués** pour diverses raisons : la délocalisation des éléments d'un système (capteurs chargés de récupérer les entrées, actionneurs responsables de la production des sorties, processeurs servant de support d'exécution), le gain de performance grâce à l'utilisation de plusieurs calculateurs pour améliorer les temps de réponse des systèmes, la tolérance aux fautes à travers la duplication de certaines parties d'un système.

Au vu de toutes ces caractéristiques, la conception des systèmes réactifs temps réel requiert des méthodologies suffisamment élaborées pour répondre aux exigences d'une mise en œuvre fiable. Pour ces raisons, la prise en compte des techniques formelles dans ces méthodologies apparaît de plus en plus indispensable.

La conception des systèmes temps réel aujourd'hui

Une grande partie des réalisations concernant les systèmes temps réel repose sur les approches classiques de haut niveau. Elles sont d'une part l'*approche asynchrone*, qui définit des systèmes multi-tâche utilisant des *systèmes d'exploitation temps réel* (en anglais, *Real-Time Operating Systems* - RTOS) à l'instar de RT-POSIX et d'autre part, l'utilisation de *langages de programmation concurrente* à l'image de ADA.

Aujourd'hui, nous pouvons clairement observer certaines insuffisances de ces approches si l'on souhaite garantir à la fois la fiabilité requise par les systèmes temps réel et la réduction nécessaire du coût de conception. Une des raisons est liée notamment aux pratiques concernant le développement de ces systèmes [Wir01]. En particulier, l'absence de méthodologies rigoureuses de programmation a longtemps entraîné la présence de *bogues*, souvent difficilement repérables dans le code. Une autre raison provient de la façon dont on procède pour la vérification des comportements du système conçu. Les techniques utilisées reposent souvent sur la simulation et le test. Dans les deux cas, il se pose d'abord la question de l'exhaustivité des situations considérées pour juger de la pertinence des résultats obtenus. De plus, ces derniers dépendent d'une plate-forme donnée ; qu'en serait-il sur d'autres plates-formes ? Le test est en soi coûteux.

Ces insuffisances liées aux approches classiques motivent la définition de nouvelles approches pour résoudre les lacunes des premières. De l'avis de plusieurs experts, tant du monde académique qu'industriel, les *approches basées sur des modèles* représentent une bonne alternative. Parmi elles, nous privilégierons ici l'*approche synchrone*, qui est fondée sur des bases mathématiques solides, permettant ainsi des raisonnements formels et facilitant par la même occasion la validation des mises en œuvre d'un système. De plus, la technologie synchrone offre les outils adéquats au développeur pour mener à bien sa mission (à travers des spécifications formelles, la génération automatique de code ou la vérification). Les *approches par modèles temporisés* proposent, au même titre que l'approche synchrone, des moyens formels pour la conception des systèmes temps réel. Cela est

réalisé au travers de langages spécifiques ou bien d'extensions temporisées de concepts comme les automates.

Une autre observation résulte des thèmes d'étude ayant focalisé l'attention des experts dans le domaine des systèmes temps réel. Raj Rajkumar de Carnegie Melon University[1] note que l'étude de l'ordonnancement dans ces systèmes est l'un des sujets les plus traités. Si cela a permis d'établir plusieurs résultats intéressants, il n'en demeure pas moins que c'est au détriment des autres aspects, restés peu explorés comparativement à la problématique de l'ordonnancement. Par exemple, on peut souligner le caractère récent des travaux autour de notions comme la *qualité de service* (en anglais *Quality of Service* - QoS), utile dans le cadre du développement d'applications multimédia temps réel, ou les *intergiciels* qui offrent beaucoup de flexibilité aux architectures de système. On peut aussi mentionner l'importance de plus en plus grandissante de la modélisation à travers l'émergence de concepts tels que MDA (*Model Driven Architecture*) basé sur le *méta-modèle* UML (défini par l'OMG - *Object Management Group*), ou des technologies parmi lesquelles figurent les outils et techniques synchrones. Il y a donc nécessité de définir de nouvelles orientations sur les aspects à explorer dans les systèmes temps réel.

De manière générale, l'étude des propriétés non fonctionnelles fait l'objet de plus en plus d'attention. À ce sujet, un thème comme l'évaluation de performances suivant différentes métriques (temps de réponse, énergie consommée, etc.) doit être mis en avant.

La maîtrise d'une conception fiable des systèmes temps réel passe par la prise en compte de toute la problématique soulevée en partie par les observations ci-dessus. Il faut disposer d'approches à la fois formelles et pratiques qui permettraient d'aborder efficacement les problèmes liés aux propriétés comportementales et non fonctionnelles de ces systèmes, afin d'être capable de garantir la validité de leur mise en œuvre.

Contexte de l'étude et approche proposée

Nous présentons d'abord le contexte de notre travail, ensuite nous donnons l'idée principale de la solution que nous proposons pour répondre aux attentes considérées dans un tel contexte.

Contexte. L'étude menée dans cette thèse se place dans le cadre du projet européen IST-1999-10913 SAFEAIR[2] (Advanced Design Tools for Safety-Critical Systems), qui a débuté en janvier 2000. Ce projet a pris la suite du projet ESPRIT SACRES[3] (Safety-Critical Embedded Systems) qui avait pour mission principale de fournir aux concepteurs de systèmes critiques enfouis, une méthodologie de développement réduisant de façon significative le risque d'erreurs et le temps de conception. Les principaux résultats de ce projet ont été :

– d'une part, la définition d'une plate-forme intégrant des outils liés aux différents langages considérés. Exemple : l'outil MAGNUM STATEMATE de la société *i-Logix*[4] pour le langage STATECHARTS, et SILDEX développé et commercialisé par *TNI-Valiosys*[5] pour SIGNAL.

[1] Communication orale : "*Advances in Large-Scale Distributed Real-time and Embedded Systems*" (IEEE *Real-Time and Embedded Technology and Applications Symposium*, Mai 2003, Washington D.C. - USA).

[2] `http://www.safeair.org/`.

[3] `http://www.tni.fr/sacres/`.

[4] `http://www.ilogix.com/`.

[5] `http://www.tni.fr/`.