Proposition de recherche doctorale

Intégration de politiques de sécurité et de sûreté de fonctionnement pour la modélisation, la vérification et la génération de systèmes critiques

Mots clés :

- **Directeur de thèse** : LAURENT PAUTET
- **Co-encadrant(s)** :
- **Unité de recherche** : Laboratoire Traitement et Communication de l'Information
- **Ecole doctorale** : École Doctorale Informatique, Télécommunications, Électronique de Paris
- **Domaine scientifique principal**: Divers

Résumé du projet de recherche (Langue 1)

1 Introduction Le travail de thèse a lieu dans l'équipe S3 (System, Software and Services), au sein du département « Informatique et Réseaux» de Télécom ParisTech. S3 se compose actuellement de 8 enseignants-chercheurs permanents (dont 3 habilités à diriger des recherches), 12 doctorants. L'équipe S3 s'intéresse à la modélisation, la conception et la vérification des systèmes répartis et en particulier des systèmes Temps Réel Répartis Embarqués (TR2E). Notre objectif est de développer des techniques de modélisation, des outils de vérification, des platesformes d'exécution et des générateurs de code permettant à des ingénieurs de développer sans risque des systèmes TR2E. En particulier, S3 a acquis une expertise reconnue internationalement dans l'architecture d'intergiciels hautement configurables en fonction des propriétés requises par le système. Ces cinq dernières années, S3 a également développé une forte compétence dans la modélisation et la génération automatique des systèmes TR2E par le biais d'ADL et notamment d'AADL. 2 Contexte L'usage des systèmes Temps Réel Répartis Embarqués (TR2E) s'est aujourd'hui généralisé, tant auprès du grand public (téléphonie mobile, PDA, ...) qu'au sein de systèmes complexes (issus du domaine aérospatial, automobile, énergétique). Le recours à ces systèmes dans de nombreuses activités critiques (aéronautique, spatial, transport, médical, bancaire) ajoute une contrainte forte en termes de sûreté de fonctionnement : une erreur peut avoir un impact économique, industriel voire humain important. Il est donc crucial d'intégrer la sécurité et la sûreté de fonctionnement au processus de construction de l'application afin d'évaluer et limiter tout risque d'erreurs. Ceci passe autant par l'application d'éléments formels et théoriques (étude stochastique) que techniques de conception (DO-178B, IMA, OMG DDS, MILS). En parallèle, la complexité croissante du logiciel a conduit à la définition de couches d'abstraction et de méthodes de conception. Ainsi l'industrie s'oriente vers des environnements de modélisation et des infrastructures logicielles d'exécution à base de composants logiciels permettant d'améliorer le processus de développement par le biais d'une rationalisation de la gestion de projet (réutilisation, intégration, maintenance évolutive), d'une séparation entre les préoccupations fonctionnelles et celles purement techniques (adaptation aux couches sous-jacentes, communication, auto-observation, intégrité ...), et d'un outillage permettant de valider ou analyser le couple application/environnement d'exécution (évaluation des taux de pannes, de l'ordonnancement, etc.). Les environnements de modélisation et de conception ainsi que les intergiciels qui les accompagnent doivent donc intégrer des préoccupations de vérification formelle des composants comme leur isolation en fonction de leur niveau de sécurité ou de criticité. Cette thèse s'intègre dans les travaux menés dans le contexte du projet PARSEC. A titre indicatif, nous rappelons l'objet de celui-ci. Le projet PARSEC vise à définir un atelier de développement des systèmes temps-réel embarqués distribués critiques qui nécessitent une certification selon les standards les plus stricts tels que les normes DO-178B (avionique), IEC 61508 (transports), ou les Critères Communs pour l'évaluation de la Sécurité des Systèmes d'Information (Sécurité des systèmes d'information). La démarche poursuivie par PARSEC a pour but de fournir aux ingénieurs en charge de ces systèmes une suite d'outils intégrée pour répondre aux contraintes de certification des systèmes embarqués. En l'occurrence, ces systèmes doivent : – être rigoureusement spécifiés - PARSEC s'appuiera sur l'approche B événementielle pour aboutir à un modèle prouvé de l'attendu du système. – être rigoureusement conçus - PARSEC offrira alors les moyens de décliner cet attendu en un modèle de composants (technologie MyCCM High Integrity) décrivant l'application et ses paramètres nonfonctionnels (caractéristiques temps réel, projection sur une plate-forme partitionnée notamment). – être rigoureusement implantés - PARSEC définira des générateurs pour le code correct par construction pour l'assemblage et le contrôle des composants logiciels, en suivant d'une part une approche synchrone (technologie SynDEx) et d'autre part une approche asynchrone (technologie Ocarina). Les deux approches ont des applications différentes (implantation de lois de pilotage ou de protocoles de radiocommunications, par exemple). – être rigoureusement évalués (remontée du cycle en V) - PARSEC intégrera l'outil PathCrawler de génération de scénarios de tests et définira un outil de traçabilité d'exigences. 3 Positionnement du travail Le logiciel et son environnement d'exécution doivent fournir des éléments permettant de valider finement son bon fonctionnement, dans des contextes où la certification (avionique) ou la validation (spatial) sont cruciaux pour autoriser l'utilisation du système. Se posent alors de multiples questions lors de la construction du couple logiciel/environnement d'exécution : quelles sont les propriétés à tester ? Quelles sont les techniques de validation associées ? Quelles sont les processus et les outils à mettre en oeuvre pour guider l'ingénieur dans ces choix de conception et lors de la réalisation ? La variété des besoins impose la définition d'un processus intégré reposant sur des standards, combinés et intégrés et sa mise en oeuvre. La sécurité et la sûreté de fonctionnement imposent la définition de politiques, leur validation (e.g. théorème de Bell-Lapadula) et leur mise en oeuvre dans le cadre de standards (OMG DDS, RT-CORBA, ARINC 653, etc). La prise en compte simultanée de ces deux aspects pose pour le moment de nombreux problèmes théoriques visant à terme la suppression des méthodes ad hoc employées aujourd'hui et son remplacement par un processus fiable autorisant séparation des préoccupations, analyse guidée, configuration de l'environnement d'exécution. 4 Problématique de la thèse L'hétérogénéité des besoins tant théoriques que pratiques amène de nouveaux problèmes pour l'ingénierie logicielle, qui doit pouvoir s'adapter à une

large variété de théories (model checking classique, stochastique, temporisé, preuves) et technologies (intergiciels, noyaux temps réel), sans pour autant en maîtriser tous les aspects. Il devient alors nécessaire d'abstraire l'architecture du système pour permettre son analyse, et la définition de ces environnements. Une architecture comme celle proposée dans les spécifications de MILS permet d'effectuer des vérifications mathématiques (sur l'ordonnancement par exemple) en réduisant et isolant de manière significative les portions de code critiques. La description de l'architecture vise à traiter de plusieurs problèmes difficiles. Tout d'abord, de pouvoir vérifier que certaines contraintes de sécurité et de sûreté de fonctionnement peuvent être vérifiées hors ligne. La problématique est d'autant plus grande que ces deux aspects peuvent s'entremêler voire se composer par une approche hiérarchique. Or jusqu'à présent ces deux aspects ne sont modélisés ni de manière commune ni selon une approche composite. Par ailleurs, s'il existe de nombreuses démarches de générations de code, peu visent à produire un code déterministe. Cependant, on conçoit bien que pour assurer des propriétés de sécurité et de sûreté, il faut garantir une certaine confiance dans le code généré. Au regard des efforts déployés dans le cadre de SCADE, il convient d'adopter des solutions pragmatiques pour assurer une confiance suffisamment forte. Enfin, toutes les contraintes de sécurité et de sûreté ne peuvent être vérifiées hors ligne et certaines devront se faire en ligne. Le code généré devra donc se charger de ces vérifications à l'exécution avec les problèmes de dualité et de hiérarchie des aspects sécurité et sûreté déjà exposé. Il conviendra également de produire du code déterministe vers un intergiciel adapté. De nombreux problèmes méthodologiques se posent : des problèmes tant d'ingénierie logicielle, que d'implantation et validation ; et leur traitement nécessite des oe dirigée par les modèles. L'approche proposée pour atteindre ces objectifs consistera à : – Élaborer une typologie des standards existants ou en cours de définition pour la sûreté de fonctionnement et la sécurité ; des éléments théoriques les supportant et les outils associés. On s'intéressera en particulier aux initiatives MILS, IMA et aux standards D0-178B/C et OMG DDS ; on s'intéressera également dans la validation des générateurs de code ; – Définir des formalismes pour la prise en compte simultanée des propriétés de sécurité et de sûreté de fonctionnement. Au travers d'un langage comme AADL, cela pourra s'effectuer au travers d'annexes qu'il conviendra de rendre compatible pour assurer l'utilisation simultanée de ces aspects. – Produire automatiquement les composants logiciels de telle manière que l'on assure une certaine confiance dans la partie générée. Dans un premier temps, cela pourra s'effectuer par génération d'annotations qui seront par la suite analysées. Le code généré s'appuiera sur des intergiciels dédiés intégrant des fonctionnalités de sécurité et de sûreté de fonctionnement. – Vérifier la cohérence des modèles exprimées en fonction des formalismes précédemment définis ainsi que la conformité du code généré. Il faut noter que, dans le cas de hiérarchie de partitions intégrant simultanément des aspects de sécurité et de sûreté de fonctionnement, la cohérence peut s'avérer complexe à réaliser.

Résumé du projet de recherche (Langue 2)

L'hétérogénéité des besoins tant théoriques que pratiques amène de nouveaux problèmes pour l'ingénierie logicielle, qui doit pouvoir s'adapter à une large variété de théories (model checking classique, stochastique, temporisé, preuves) et technologies (intergiciels, noyaux temps réel), sans pour autant en maîtriser tous les aspects. Il devient alors nécessaire d'abstraire l'architecture du système pour permettre son analyse, et la définition de ces environnements. Une architecture comme celle proposée dans les spécifications de MILS permet d'effectuer des vérifications mathématiques (sur l'ordonnancement par exemple) en réduisant et isolant de manière significative les portions de code critiques. La description de l'architecture vise à traiter de plusieurs problèmes difficiles. Tout d'abord, de pouvoir vérifier que certaines contraintes de sécurité et de sûreté de fonctionnement peuvent être vérifiées hors ligne. La problématique est d'autant plus grande que ces deux aspects peuvent s'entremêler voire se composer par une approche hiérarchique. Or jusqu'à présent ces deux aspects ne sont modélisés ni de manière commune ni selon une approche composite. Par ailleurs, s'il existe de nombreuses démarches de générations de code, peu visent à produire un code déterministe. Cependant, on conçoit bien que pour assurer des propriétés de sécurité et de sûreté, il faut garantir une certaine confiance dans le code généré. Au regard des efforts déployés dans le cadre de SCADE, il convient d'adopter des solutions pragmatiques pour assurer une confiance suffisamment forte. Enfin, toutes les contraintes de sécurité et de sûreté ne peuvent être vérifiées hors ligne et certaines devront se faire en ligne. Le code généré devra donc se charger de ces vérifications à l'exécution avec les problèmes de dualité et de hiérarchie des aspects sécurité et sûreté déjà exposé. Il conviendra également de produire du code déterministe vers un intergiciel adapté. De nombreux problèmes méthodologiques se posent : des problèmes tant d'ingénierie logicielle, que d'implantation et validation ; et leur traitement nécessite des outils dédiés afin d'aider l'ingénieur dans sa phase de conception pour à la fois valider et déployer l'application sur l'environnement configuré.