THÈSE

présentée par

Karim MEGZARI

pour obtenir le diplôme de
DOCTEUR DE L'UNIVERSITÉ DE SAVOIE
(Arrêté ministériel du 30 mars 1992)

Spécialité : Informatique

REFINER : Environnement logiciel pour le raffinement d'architectures logicielles fondé sur une logique de réécriture

---

Soutenue publiquement le** 17 décembre 2004 **devant le jury composé de :

Richard MCCLATCHEY	Président	Professeur à l'Université de West of England, CCCS, Bristol, Royaume Uni
Nacer BOUDJLIDA	Rapporteur	Professeur à l'Université Henri Poincaré de Nancy
François JACQUENET	Rapporteur	Professeur à l'Université Jean Monnet de Saint-Etienne
Flavio OQUENDO	Directeur	Professeur à l'Université de Savoie
Ilham ALLOUI	Co-encadrant	Maître de Conférences à l'Université de Savoie

préparée au sein du LISTIC
Laboratoire d'Informatique, Systèmes, Traitement de l'Information et de la Connaissance
ESIA – Université de Savoie

RESUME

Nous nous sommes intéressés dans cette thèse aux problèmes liés à l'insuffisance de support informatique pour le raffinement d'architectures logicielles. Le peu de langages de description d'architectures (ADL) comme SADL et Rapide qui abordent ces problèmes le font de manière très restreinte. En effet, le support de SADL ne permet qu'un raffinement structurel et exige des preuves manuelles pour les constructions de mise en correspondance entre un modèle abstrait et un modèle architectural plus concret. Dans Rapide, la mise en correspondance d'événements entre des architectures individuelles est bien supportée grâce à un sous-langage exécutable, mais par contre, seul le raffinement comportemental y est possible. Notre contribution consiste en la fourniture d'un environnement logiciel que nous avons nommé *Refiner*, fondé sur la logique de réécriture. Le choix de cette dernière est motivé par le fait qu'elle soit bien adaptée à la description de systèmes concurrents, leur raffinement par le mécanisme de réécriture et également par le fait qu'elle soit bien supportée par des outils logiciels. Nous avons construit *Refiner* autour du langage de raffinement d'architectures *ArchWare ARL* de manière à ce qu'il permette à ses utilisateurs :

- de raffiner progressivement des éléments architecturaux (composants, connecteurs, ports, etc.) depuis des descriptions abstraites vers des descriptions concrètes, et ce à travers des niveaux d'abstraction multiples. A chaque étape de raffinement, peut être appliquée une action qui fournit par construction, sous certaines obligations de preuves, une transformation architecturale correcte. Ceci est possible de par le fait que les *pré-conditions* et *post-conditions* rattachées aux actions de raffinement dans *ArchWare ARL,* soient formalisées et implémentées dans le langage formel qu'est la logique de réécriture. Par conséquent, par construction, une action de raffinement transforme une architecture satisfaisant les pré-conditions en une architecture moins abstraite satisfaisant les post-conditions. Le modèle de transformation architecturale est correct s'il satisfait les pré et les post-conditions ainsi que les obligations de preuve.
- de raffiner plus que la structure et le comportement d'un élément architectural. En effet, l'approche *ArchWare ARL* permet au cours du processus de raffinement d'établir des relations de raffinement portant sur des comportements, des ports, des structures, et des données d'éléments architecturaux.
- de construire au cours d'une étape de raffinement, aussi bien une description architecturale plus concrète traduisant une architecture possible qu'une description simplifiée traduisant une architecture réduite. En effet, le premier type de raffinement est possible de par le principe de sous-spécification de *ArchWare ARL* (à un haut niveau d'abstraction, on spécifie un élément architectural tout en laissant certains aspects non spécifiés). La diminution de cette sous-spécification passe par l'établissement de relations de raffinement des comportements, des ports, des structures, et des données des éléments architecturaux. Le second type de raffinement (réduction ou simplification) est également rendu possible, grâce notamment aux actions de raffinement d'*explosion* et d'*implosion*.
- de supporter, au niveau le plus concret du raffinement architectural, la génération des applications dans des langages de programmation cibles. Ceci se fait par le biais du générateur d'applications *Sigma* fondé sur les concepts de *mapping* et de *patterns de synthèse*.

L'environnement *Refiner* proposé permet aux architectes de raffiner correctement des éléments architecturaux, puis de recomposer les éléments raffinés pour la construction d'architectures concrètes. Le choix de la logique de réécriture comme fondement à la fois de la sémantique formelle et de la sémantique opérationnelle de l'environnement permet de naturellement décrire des architectures logicielles, leur raffinement par le mécanisme de réécriture ainsi que leur "exécution" grâce aux outils proposés pour cette logique.