Conclusion.

Rappelons que notre objectif était d'étudier l'intérêt de l'utilisation d'une approche objet pour la modélisation d'architectures matérielles en vue d'en évaluer les performances. Nous nous sommes très vite orienté vers une approche par composants autonomes qui permet une bonne réutilisation des modèles dans différents types d'architectures. De plus, l'utilisation d'un langage objet permet l'enrichissement incrémental du comportement des composants.

Nous avons alors montré les avantages d'un telle approche pendant les différentes étapes de la modélisation :

La définition d'un modèle générique des architectures cibles qui établit les règles de composition et les différentes interactions entre les composants rend possible la modélisation des architectures par une approche incrémentale.

En particulier, ce modèle générique permet la création d'un environnement de simulation non spécifique d'une architecture. Il est ainsi possible de disposer d'un outil de simulation qui permet non seulement la conception d'une architecture, mais aussi la mise au point des applications développées pour cette architecture. Le modèle réalisé permet une évaluation rapide des performances de l'architecture, il servira alors de modèle de référence pour la construction d'une description synthétisable.

L'autonomie des composants permet d'y intégrer des aspects génériques qui facilitent la modélisation.

C'est ainsi que nous avons d'abord défini une méthode pour construire le comportement des composants par combinaison de services élémentaires décrits en Java. Cette construction semi-automatique réduit la taille du code à écrire, ce qui améliore la réutilisation. Cette méthode est basée sur le modèle des méta-composants SEP.

De plus, il est possible d'étendre la notion de services aux modules, ce qui réduit la complexité du modèle réalisé et améliore la lisibilité. Cette notion de services de modules nous a aussi permis de définir des notions d'héritage de comportement.

Le composant décodeur générique prend en compte de façon simple le jeu d'instructions des architectures programmables modélisées et permet d'obtenir très rapidement un simulateur de jeu d'instructions.

L'autonomie des composants permet la modélisation hétérogène d'architectures. C'est ainsi que nous avons intégré dans un composant SEP une machine d'exécution du langage réactif synchrone Esterel. Ce mécanisme peut être étendu pour la prise en compte de tout modèle qui serait utile pour une description efficace d'un certain type de comportement. Par exemple, il est possible de définir un composant SEP qui joue une machine d'états décrite par un automate d'états finis ou par une représentation avec un StateCharts ou un SyncCharts.

Enfin, la généricité des données manipulées laisse au concepteur une grande liberté dans sa conception. En particulier, la taille des composants et le nombre de bits nécessaires au codage d'une instruction ou d'un code opération ne sont plus les problèmes abordés au premier abord. Le concepteur s'intéresse alors directement aux problèmes primordiaux.

Dans une deuxième partie, nous montrons comment, grâce au modèle objet générique, certains mécanismes de validation ont été mis en place afin d'augmenter la confiance que le concepteur peut avoir dans les modèles effectués.

Tout d'abord, un premier mécanisme valide la composition des composants à partir des types utilisés dans la description de leur comportement. Ainsi, certaines connexions dangereuses sont interdites.

La deuxième étape de validation permet de s'assurer que les ressources suffisantes à la réalisation de services de modules ou de schémas d'instructions sont présentes. Le mécanisme de spécification active mis en place est une aide à la création du jeu d'instructions et des modèles.

La troisième étape valide par des comparaisons d'expressions algébriques la fonction réalisée par une combinaison de services. Ainsi, il est possible de détecter les chemins de données qui manquent ou les composants séquentiels mal placés. Cette étape est rendue aisée grâce au mécanisme de liaison dynamique qui a été introduit.

Enfin, l'intégration d'une machine d'exécution Esterel permet la validation de propriétés sur les parties synchrones du modèle grâce aux outils disponibles dans l'environnement des langages synchrones.

L'utilisation de l'outil qui a été créé et l'utilisation de cette méthode de conception nous ont permis de modéliser avec succès plusieurs architectures industrielles. En effet, toutes les applications réalisées pour valider la méthode ont permis d'obtenir les résultats attendus, c'est-à-dire le même résultat que celui qui est obtenu avec les architectures réelles. Il s'agissait essentiellement d'obtenir un modèle qui fournit les mêmes résultats d'exécution et les mêmes temps d'exécution en termes de nombre de cycles. De plus, sur une architecture simplifiée nous avons mis en évidence des améliorations possibles.

Enfin, la modélisation de l'arroseur automatique illustre l'utilisation de SEP pour la modélisation d'architectures logicielles hétérogènes.

Ce travail dans le cadre d'une collaboration avec VLSI Technology – filiale de Philips Semiconductors – a donné lieu à un brevet industriel **[DMB00]**.

Perspectives.

L'utilisation d'un modèle objet et d'une technique objet aussi bien pour la conception que pour la réalisation de l'environnement de simulation permet de nombreuses extensions qu'il serait intéressant d'exploiter. En particulier, seule une vue de la composition structurelle est actuellement présentée graphiquement par notre environnement ; or, il semble évident que les services tiennent un rôle essentiel dans la modélisation des composants autonomes. Leur manipulation pourrait être facilitée et enrichie par la définition d'une deuxième vue graphique. Cette vue permettrait en particulier de manipuler très facilement la notion d'héritage de services et plus généralement tout schéma de modélisation qui fait intervenir un ensemble de composants, leurs services et leurs attributs. Le langage UML est sans aucun doute le langage graphique adapté à la représentation dans cette seconde vue des différents acteurs.

D'autre part, il peut aussi être intéressant d'étudier une méthode formelle qui permette d'utiliser les propriétés obtenues sur les composants Esterel afin d'en déduire des propriétés sur l'ensemble du module. La limitation principale dans une telle approche vient de la nature très hétérogène et très complexe des données manipulées. En effet, les langages synchrones ne sont pas adaptés pour la modélisation de structures de données complexes.

Dans bien des cas, la description de SEP semble permettre la génération automatique d'un squelette d'une description synthétisable. En particulier, les descriptions structurelles peuvent être en grande partie générées si nous considérons qu'il suffit de choisir les descriptions synthétisables adaptées des composants élémentaires simples (Multiplexeur, ALU, registres). Un tel processus ne peut être que semi-automatique, en particulier, le concepteur devra choisir tous les protocoles de communication des bus ainsi que les tailles des instructions ou des adresses.

En outre, il est intéressant et non trivial d'étudier comment du code synthétisable peut être produit à partir d'une description du jeu d'instructions en SEP-ISDL. D'autant plus, qu'à partir d'une telle description, il semble aussi possible de générer l'assembleur qui permet la traduction d'un programme assembleur vers le langage machine.

L'utilisation de SEP pour la modélisation d'architectures distribuées est aussi un défit intéressant. Une étude préliminaire d'un système composé de processeurs 68000 qui communiquent à travers un réseau de terrain de type CAN ne nous a pas permis d'entrevoir les limitations de SEP pour la modélisation de telles architectures. Cependant, une étude en cours, dans le cadre d'un *challenge* international, lancé par l'industriel Mercedes-Benz, pour la modélisation d'un siège de voiture contenant plusieurs moteurs et capteurs qui communiquent à travers un réseau CAN peut nous permettre de mettre en évidence l'intérêt de notre approche pour la modélisation de tels systèmes.

D'autre part, nous prévoyons l'étude d'un protocole de routage dynamique sur réseau ATM appelé P-NNI. Cette étude aurait pour objectif d'étudier le comportement d'un système distribué géré par cet algorithme de routage dynamique lors de pannes sur les nœuds du réseau ou lors de modifications dynamiques de la topologie.