8

Conclusions et Perspectives

Ce chapitre conclut la thèse en donnant un bilan du travail effectué et les perspectives envisageables au terme de cette recherche. Nous rappelons tout d'abord les principales caractéristiques de la méthodologie de co-design développée sur la base de la méthodologie MCSE et de son modèle de performance. Ce modèle de performance est utilisé pour l'évaluation des performances dynamiques avant et après le partitionnement matériel/logiciel. Cette évaluation des performances faite par une co-simulation repose sur la transcription du modèle de performance en un modèle VHDL simulable. Nous résumons donc également les caractéristiques essentielles de la technique de génération de code utilisée pour transcrire le modèle de performance en VHDL comportemental. Les perspectives envisageables au terme de cette thèse sont diverses. Nous commentons l'enrichissement du modèle de performance, l'estimation des performances statiques d'un système, l'estimation du temps d'exécution des opérations élémentaires et l'exploitation d'un niveau de granularité plus fine pour le partitionnement matériel/logiciel.

8.1 BILAN DE LA THESE

Dans le court terme, le problème du co-design concerne la définition et l'utilisation d'une méthodologie appropriée, cohérente, complète et efficace et d'outils associés pour aider les concepteurs à transformer le besoin du client en un produit opérationnel. A ce titre, l'objectif premier de cette thèse a été de montrer l'apport de la méthodologie MCSE et de son modèle de performance à la problématique du co-design et en particulier aux problèmes du partitionnement matériel/logiciel et de la co-simulation. La méthode de partitionnement retenue s'appuie sur une démarche itérative guidée par le concepteur et par une analyse des

performances dynamiques du système résultant d'un partitionnement. Cette analyse des performances est réalisée par une technique de co-simulation qui consiste à traduire le modèle de performance du système en un modèle VHDL simulable. Après avoir défini les règles de transcription, nous avons développé un principe générique de génération de code et un générateur de code VHDL. Les concepts du modèle de performance, les règles de transcription et le générateur de code VHDL ont été validés à l'aide de deux exemples: un serveur vidéo temps réel fourni par le CCETT de Rennes et un système de communication distribué.

-A- La méthodologie de co-design préconisée

Pour répondre à l'objectif initial, nous avons défini une nouvelle méthodologie de co-design basée sur la méthodologie MCSE. Cette méthodologie de co-design concerne principalement la conception des systèmes de contrôle/commande, des systèmes de communications et partiellement des systèmes de traitement. Ces systèmes dits dédiés sont généralement conçus pour répondre à un besoin spécifique et entrent dans la catégorie des systèmes électroniques embarqués et temps-réels (Real Time Embedded Systems). La méthodologie de co-design développée est caractérisée par une approche *système*, une modélisation selon 3 vues (fonctionnelle, comportementale et architecturale), une *architecture cible hétérogène* et non imposée, une méthode de *partitionnement matériel/logiciel interactive et itérative* basée sur une *évaluation des performances dynamiques* par co-simulation et une technique de *co-simulation macroscopique* et *non-interprétée*.

La technique de co-simulation repose sur la simulation d'un modèle de performance qui représente à la fois la partie matérielle et la partie logicielle du système. Cette solution n'est pas limitée par la complexité du système et permet d'extraire un ensemble de résultats de performances plus riche que les approches analytiques.

Pour aboutir au produit, une fois le partitionnement matériel/logiciel terminé, les descriptions fonctionnelles sont à transformer en code machine pour une implantation en logiciel et en un ensemble de portes logiques et de bistables pour une implantation en matériel. C'est le rôle de la phase de co-synthèse qui concerne la synthèse des parties matérielles et logicielles et la synthèse des interfaces matériel/logiciel. Actuellement, l'équipe MCSE travaille activement sur la génération de code C exécutable, de code VHDL synthétisable et sur la synthèse des interfaces logiciel/matériel.

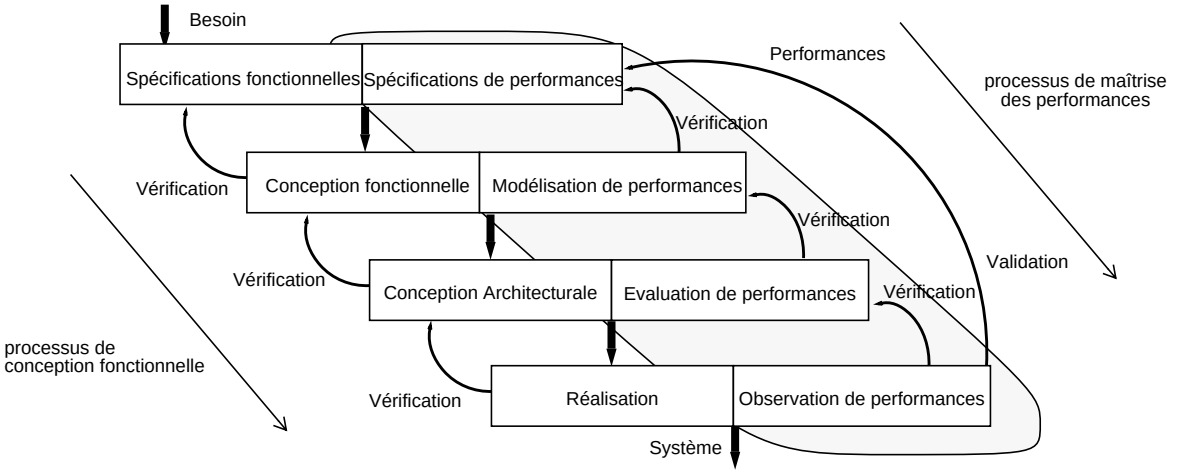
Notre approche permet d'exploiter à nouveau le modèle de performance après synthèse. En effet, l'affinement des temps des opérations avec leur rétro-annotation dans le modèle de performance permet d'évaluer le système sans utiliser obligatoirement des techniques de simulation hétérogène (IPC d'unix, fond de paniers de simulateurs, interface via le bus du microprocesseur, etc.). Ce principe de co-simulation qui utilise un niveau d'abstraction des modèles plus élevé que celui utilisé habituellement a l'avantage de réduire sensiblement les temps de simulation tout en permettant une évaluation assez fine des performances dynamiques du système. De plus, il ne se limite pas nécessairement à l'évaluation des performances. En effet, en remplaçant le temps des opérations élémentaires par une description algorithmique, on obtient alors un modèle interprété qui permet de faire une vérification fonctionnelle du système.

-B- Le modèle de performance de MCSE

Le travail effectué lors de cette thèse a permis d'enrichir et valider les concepts du modèle de performance utilisé pour l'estimation des propriétés d'un partitionnement matériel/logiciel.

Ce modèle est macroscopique, non-interprété, évolutif, générique et paramétrable. Il est basée sur l'emploi du modèle fonctionnel et du modèle exécutif de la méthodologie MCSE et d'un modèle de composition d'*activités dynamiques* qui décrit le comportement des fonctions. Les deux modèles sont enrichis avec des *attributs* pour spécifier les caractéristiques de chaque élément. L'un des attributs clef est le degré de concurrence d'un élément actif car il permet de simuler un processeur logiciel (processeur au degré de concurrence limité) et donc de faire de la co-simulation.

L'intégration du modèle de performance à la méthodologie de conception MCSE sur laquelle repose notre méthodologie de co-design offre une démarche descendante de développement intégrant au mieux la maîtrise des performances simultanément à la maîtrise des fonctionnalités. Comme le montre la figure 8.1, en utilisant le modèle de performance, le concepteur dérive par transformations et enrichissements successifs à partir des spécifications, une solution de conception puis une solution d'implantation et simultanément décrit et évalue les propriétés de performances à chaque stade. A partir de l'étape de conception fonctionnelle, l'*unicité* du modèle facilite aussi la conception sans erreur (pas de déformation ou perte d'information liées à une transcription de modèle) et améliore la *traçabilité*.



-Figure 8.1- Démarche de conception avec maîtrise des performances.

-C- Les règles de transcription en VHDL

Le choix du langage VHDL se justifie par sa standardisation, sa portabilité, la disponibilité des nombreux outils sur le marché et ses propriétés intrinsèques (modèle hiérarchique, paramétrable, parallélisme inhérent, instanciation multiple, etc.).

Cependant, comme le langage VHDL ne dispose pas de mécanisme de suspension de process, nous avons du décrire explicitement un composant ordonnanceur et une procédure spécifique de gestion des temps d'attente. Le manque de généricité pour la déclaration des types et les constructions telles que l'attente conditionnelle, l'achèvement forcé d'activité et certains cas de l'instanciation multiple ont également posé des difficultés de transcription. Pour ces cas, la complexité de la solution d'implantation en VHDL se paie malheureusement au niveau de la génération de code et de la lisibilité et de l'efficacité du code VHDL produit.

Malgré le niveau d'abstraction du modèle, la simulation des programmes VHDL obtenus à partir des modèles de performance du serveur vidéo temps-réel et du système de communication a permis de constater la nécessité de ressources importantes (puissance de calcul et mémoire) et une durée de simulation un peu trop longue pour trouver rapidement la

solution optimale recherchée. Ces constatations justifient pleinement l'intérêt d'une méthode de simulation en C++ que l'équipe développe.

-D- Génération du modèle VHDL

Le générateur de code VHDL a été réalisé sur un principe générique de développement de générateurs de code ou d'outils de transformation de textes qui permet de transcrire facilement le modèle MCSE (ou tout autre langage source) vers d'autres langages cibles. Un générateur est alors le résultat de la définition des grammaires des langages source et cible afin d'obtenir les analyseurs syntaxiques associés, la définition d'un ou plusieurs fichiers template et l'écriture d'un script. Ce script sert de point d'entrée au générateur de générateurs de code ou méta-générateur nommé MetaGen qui permet d'interpréter le script ou de le transcrire en code JAVA. L'écriture du script du générateur de VHDL comportemental a permis de constater que tout script est décomposable en une partie analyse du modèle source commune à tous les générateurs de la plate-forme MCSE et une partie génération de code spécifique.

Notre solution permet d'obtenir des outils multi plate-formes, basés sur une architecture générique commune, configurables et plus faciles à développer et à enrichir.

Le travail effectué sur la génération de code a aussi amené l'équipe MCSE à revoir sa stratégie de développement des outils comme support pour la méthodologie MCSE. En effet, l'expérience de génération de code effectuée lors des travaux de cette thèse a permis d'appréhender les concepts de générateurs d'analyseurs syntaxiques et de méta-structure sur lesquels repose entièrement la nouvelle "philosophie" de développement des outils MCSE.

8.2 PERSPECTIVES

Les perspectives envisageables en prolongement direct de cette thèse concernent 3 objectifs importants: l'enrichissement de l'estimation des propriétés des systèmes par une estimation du temps d'exécution des opérations élémentaires et des performances statiques d'un système, l'enrichissement des concepts du modèle de performance pour l'analyse de la sûreté de fonctionnement et de la tolérance aux fautes des systèmes, le changement du niveau de granularité du partitionnement matériel/logiciel.

-A- Enrichissement de l'estimation des propriétés des systèmes

Notre technique d'estimation des propriétés d'un système par co-simulation s'applique efficacement aux performances dynamiques du système. Il est utile de compléter cette évaluation par une estimation du temps d'exécution des opérations (activités élémentaires) et une analyse des performances dites statiques des propriétés des constituants et du système résultant.

Avec le modèle de performance de MCSE, les estimations de performances dynamiques s'obtiennent à partir des temps d'exécution supposés pour les opérations. Ces temps peuvent se déduire plus précisément à partir d'une description VHDL ou C de l'algorithme de l'activité élémentaire. Il s'agit avant tout d'utiliser les travaux réalisés ou actuels sur cet aspect (Program Analysis): technique de profiling, utilisation d'un modèle ISA [KNUDSEN-95] [BALBONI-95] [ROSE-96] ou transformation de la description en un graphe flot de contrôle/donnée (CDFG) et analyse du graphe obtenu [NARAYAN-92b] [MALIK-95] [GUPTA-95].

Une estimation des performances que l'on peut considérer plutôt statiques des systèmes nous apparaît utile d'une part pour aider le concepteur lors du partitionnement système et d'autre part pour réduire l'espace des solutions possibles à parcourir lors du partitionnement

matériel/logiciel. L'objectif est alors d'obtenir une zone plus réduite des solutions appropriées pour tendre vers une solution globalement optimale et d'appliquer ensuite sur cette zone notre technique d'estimation des performances dynamiques afin d'obtenir la solution optimale vis à vis des contraintes imposées. Pour cela, nous recommandons autant que possible l'emploi des techniques et outils existants et des méthodes d'évaluation analytique. Les performances statiques de chaque processeur matériel (surface de silicium occupée, puissance consommée, nombre de broches) dérivent des techniques de synthèse comportementale [NARAYAN-92b] [VAHID-92] ou d'estimations à priori et celles des processeurs logiciels (taille du code, taille de la mémoire nécessaire) des techniques de compilation.

Le concept d'attributs utilisé dans le modèle de performance est très intéressant pour calculer une estimation globale du système à partir des estimations de chaque constituant, obtenir une estimation grossière très tôt dans le cycle de développement (loi empirique pondérée par un ensemble d'attributs), laisser la liberté aux concepteurs d'utiliser des formules d'estimation qui lui sont propres ou encore faire une estimation des coûts. En effet, à partir des paramètres obtenus par une estimation des performances statiques du système et représentés sous forme d'attributs ('Area, 'Pin, 'CodeSize, 'MemoryUsed, etc.), une approximation du coût des composants et du temps de conception du système est calculable [MADISETTI-95].

-B- Enrichissement du modèle de performance pour la sûreté de fonctionnement

Parmi les contraintes que doit satisfaire un système électronique embarqué, la sûreté de fonctionnement est souvent une des exigences non-fonctionnelles non évidentes formulées dans le cahier des charges du produit à concevoir. Aussi, il apparaît intéressant et opportun d'enrichir le modèle de performance avec des concepts dédiés à l'analyse de la sûreté de fonctionnement et de la tolérance aux fautes d'un système. Pour la tolérance aux fautes, un concepteur peut saisir un modèle de performance qui représente un système avec détection et correction d'erreur. Compte tenu de notre modèle, pour l'instant, il doit explicitement définir le principe de détection et correction d'erreur retenu. Pour l'aspect sûreté de fonctionnement, il faut être capable de modéliser le dysfonctionnement éventuel d'un processeur ou des éléments de communication inter-processeurs (mémoire commune, interruption et noeud de communication). Tout comme pour la tolérance aux fautes, le modèle doit faciliter la création, la détection et la correction d'un dysfonctionnement d'un constituant de l'architecture matérielle. Pour cela, des attributs représentant par exemple le taux de dysfonctionnement et le degré de dédoublement d'un élément peuvent être rajoutés.

-C- Changement du niveau de granularité pour le partitionnement matériel/logiciel

Le degré de granularité du partitionnement matériel/logiciel de notre méthodologie est du niveau tâche car l'unité d'allocation est la fonction. Le comportement de chaque fonction est caractérisé par une composition d'activités dynamiques. Or, il y a actuellement un développement incontestable en recherche sur les architectures reconfigurables dynamiquement et les méthodologies associées pour leur exploitation (reconfigurable computing). Il nous apparaît possible et opportun de choisir comme unité d'allocation l'activité dynamique. Ainsi, les activités d'une fonction considérée jusqu'à présent indivisible, pourraient être allouées sur plusieurs processeurs ou réalisées par un même processeur matériel à des instants différents. Les générateurs de code utilisés durant la phase de synthèse devront exploiter le fait qu'une activité à une durée de vie limitée dans le temps pour permettre la reconfiguration dynamique et temps-réel des processeurs matériels au cours du fonctionnement

du système. Le partitionnement devient alors principalement un problème d’ordonnancement de tâches dynamiques dans un système distribué.