G-SCOP
SCIENCES POUR LA CONCEPTION, L'OPTIMISATION ET LA PRODUCTION

Centre National de la Recherche Scientifique

Grenoble INP

Proposition de thèse CIFRE à G-SCOP:

Gestion des modèles de simulation en conception électronique avec des approches de type Système d'information et Machine Learning

1. Contexte

A partir d'un premier cahier de charges définissant les fonctions, les performances et la consommation d'une nouvelle génération de puces électroniques (vitesse de visualisation d'une vidéo, tenue de charge en heures…), les concepteurs électroniques déterminent la famille de puces à développer et en définissent les composants. Pour s'assurer que ces futures puces respectent le cahier des charges, leurs fonctions, performances et consommation sont testées très tôt, plusieurs années avant même qu'elles ne soient gravées sur le silicium, et suivies durant toute la durée de la conception. Cette capacité à correctement estimer les caractéristiques des puces le plus tôt possible est critique. Toute erreur de conception découverte après gravure du silicium compromet la mise sur le marché et a de lourdes conséquences pour Intel®. Les pertes financières s'élèvent à plusieurs millions de dollars, sans compter le fait que l'image de la marque est détériorée et que le champ est laissé libre aux produits concurrents.

Dans ce contexte les architectes power sont responsables du suive de la consommation des futures puces. Ils remontent ces données aux chefs de projet responsables de la tenue d'un budget consommation relié la tenue en heures de la batterie sur un panel de scenarios d'utilisation représentatifs des utilisateurs visés (gaming, bureautique,…). Les architectes power écrivent les modèles de consommation de tous les composants (mémoire, arbre d'alimentation, cœur,…) des puces et les agrègent pour obtenir le modèle de consommation totale de chaque puce. Ces modèles sont basés sur des données de conception (nombre de gates, types de gates,…) et de caractérisations issues de simulation partielles, par exemple des simulations précises mais sur quelques cycles seulement, ou inférées à partir de mesures sur les générations précédentes. Ils calculent alors la consommation totale des puces et remontent les résultats aux chefs de projet. En cas de dérives par rapport au budget, ils remontent également des recommandations de changements. A partir de ces informations, les chefs de projets peuvent demander une modification de la puce, par exemple décider de réduire la performance (limiter le temps passé en mode turbo, réduire la fréquence turbo, remplacer un composant performant mais coûteux par un autre plus économe,…) ou investir du temps pour

Laboratoire G-SCOP

46, avenue Félix Viallet

38031 GRENOBLE Cedex 1

Tél. : +33 4 76 57 43 20

Fax : + 33 4 76 57 46 95

optimiser un composant. Ils peuvent aussi renégocier le budget consommation si la performance est jugée plus critique que la consommation. En cas de choix d'une nouvelle solution le « device under design » est changé. L'expert met alors à jour son modèle afin de calculer le nouveau « device under design ». Cette boucle de rétroaction doit être la plus rapide possible pour pouvoir réagir rapidement à tout écart au cahier des charges. D'autant plus qu'au cours du projet les événements susceptibles d'impacter la conception des puces sont nombreux :

- le cahier des charges peut s'adapter à la demande, pour une plus grande autonomie des ordinateurs portables par exemple,
- les mesures de consommation faites sur un prototype d'un composant peuvent s'avérer différentes de ce qui était attendu,…

Le temps de mise à jour des modèles de consommation « power » est critique. La consommation n'est qu'un aspect du design des puces et pendant que les modèles de conso sont mis à jour le reste du design continue à avancer. Si les résultats arrivent trop tard, ils ne sont plus pertinents. Ou au contraire le reste du design pourrait être bloqué en attendant les résultats de consommation. Le temps de mise à jour des modèles impacte donc directement le temps de mise sur le marché des puces. Cette phase d'arbitrage dure environ 18 mois. Au début de cette phase, pour chaque puce, une centaine de variantes sont définies où chaque variante répond à un besoin précis (gaming, bureautique, graphique,…). A chaque arbitrage la pertinence de chaque variante est réévaluée et les moins intéressantes sont supprimées pour qu'il n'en reste qu'une dizaine correspondant aux produits finalement mis sur le marché.

Pour faire face à ces défis, le site de Moirans a mis en place une application web nommée Intel® Docea™ Power Analytics. Cette application collaborative propose un mécanisme de suivi de versions et de gestion des dépendances. Elle fournit une base de données de modèles de consommation commune à tout Intel®. Les modèles sont écrits par les architectes power et sont composés d'équations, du type *P=UxI* pour les plus simples d'entre elles, et d'imports de sous-modèles, jusqu'à plusieurs milliers, dont les consommations viennent s'ajouter à celle du modèle parent. Quand un modèle est mis à jour, les architectes qui l'utilisent sont avertis. Ils peuvent alors facilement récupérer la nouvelle version pour mettre à jour leur modèle et automatiquement avertir les autres équipes

La taille typique des modèles présents dans l'outil est de 1 million d'équations et cette taille va croissant. Avec cette augmentation de complexité de nouveaux besoins apparaissent et des opérations qui étaient jusqu'à présent faisables à la main ou partiellement automatisée nécessitent d'être complétement automatisées. Par exemple, un modèle de processeur peut-être dupliqué pour simuler plusieurs variantes (nombre de cœur, quantité de mémoire,…). Quand une variable impactant toutes les configurations du processeur, comme la valeur de la

fréquence turbo, est modifiée, toutes les copies du modèle (variantes), plusieurs centaines à certains stades du projet, doivent être mises à jour. Cette opération est aujourd'hui encore longue, laborieuse et donc source d'erreur de conception. Elle peut être faite directement à la main, en récupérant pour chaque copie les mises à jour nécessaire, ou scriptée par les architectes eux-mêmes grâce à l'API qu'expose l'outil. Il manque à l'outil un mécanisme de gestion automatisée de dérivatifs de modèles. Un autre exemple concerne les rapports envoyés aux chefs de projets. Les résultats fournis par l'application ne satisfont pas directement ce besoin. Les architectes power les extraient et les réorganisent avant de les remonter aux chefs de projets. Par exemple, on leur demande de fournir la consommation par variante d'une puce. Comme la notion de variante n'existe pas dans l'outil, ce résultat ne peut pas être fournit directement. Les architectes, qui seuls savent quels modèle sont les variantes demandées, doivent écrire des scripts qui produisent les résultats voulus. Cette opération, qui n'est pas leur cœur de métier, leur prend du temps et leur temps est précieux. Il manque les objets du PLM et les résultats associés. Ces deux situations ne sont que deux exemples parmi d'autres des limitations connues de l'outil (tagging de version,…).

Ces limitations de Intel® Docea™ Power Analytics ont un impact négatif sur la productivité des architectes power et donc sur la vitesse de mise sur le marché des puces. L'équipe en charge de développer l'outil lui ajoute régulièrement des fonctionnalités pour combler ces manques. Mais développer une nouvelle fonctionnalité prend du temps et n'est pas sans risque : si elle n'est pas adaptée aux besoins réels des architectes power elle sera ignorée. Dans ce cas ce sont des semaines, voire des mois, de développement à jeter à la poubelle. Pire, une mauvaise fonctionnalité peut s'avérer plus gênante qu'utile et mettre en danger l'intégration de l'outil dans le flot de conception des puces. Là aussi le coût est énorme : la migration du flot de conception vers de nouveaux outils est longue et risquée et compromet la mise sur le marché des puces en cours de développement. Pour éviter de tels revers, les besoins réels des architectes power doivent être connus. Mais ils sont très occupés et peu disponibles pour parler de l'outil. Les ingénieurs qui développent Intel® Docea™ Power Analytics doivent donc souvent se contenter d'information partielle et en déduire des besoins supposés. Pour reprendre un des exemples précédents, les rapports personnalisés construits par les architectes power sont mal connus, ce qui les rend difficiles à implémenter. Il manque actuellement des outils pour automatiser la collecte d'information en provenance des utilisateurs. Ces informations permettraient aux développeurs de Intel® Docea™ Power Analytics de mieux comprendre comment l'outil est utilisé, modéliser le métier des architectes power, mieux identifier ces limitations et ainsi réduire, voire supprimer, l'écart entre les besoins supposés et réels des utilisateurs.

2. **Objectifs de la thèse**

2.1 Position du problème

Le besoin de système de contrôle de version (VCS : Version Control Systems en anglais) dédié aux concepteurs n'est pas nouveau. SI on regarde les disciplines connexes ou une problématique « similaire » a été abordée avec un certain succès différentes approches ont donnés différents outils. Parmi les plus diffusés (et utiliser) on peut citer les environnements de type PLM fort utilisés en conception mécanique (CAO) , les environnement très utilisés dans le domaine de la CAO électronique (autour de Cadence par exemple). Dans le cas de développement de code informatique les outils standard sont CVS,SVN et plus récemment git.

La caractéristique commune de ces environnements est qu'ils gèrent des données (géométrie, matériaux , layout …).Les données que nous sommes ici amenés à gérer sont des modèles d'équations , les équations elles-mêmes ainsi que les métadata nécessaires en particulier pour produire les différents rapports de performances.. La gestion des modifications concurrente ou non est difficile. Les modèles évoluent (comme vu dans le process) constamment. UN grand nombre de variantes peut (doit) être gérée par la même équipe.

Compte tenu de la montée en puissance de l'utilisation il faut impérativement définir un cadre formel de modélisation adapté à cette activité de création/modélisation/simulation/présentation des résultats de ces outils POWER.

Les outils que nous devons ici développer doivent s'inspirer d'après nous des deux mondes de system de version. D'un côté les outils de l'informatique comme git qui excellent dans la gestion du code par comparaison en mode texte de lignes de code (line-oriented). Cette vison est cependant mal adaptée à la représentation des modèles POWER plutôt représentée via des graphes car ce type de représentation est particulièrement efficace dans le cas de la simulation de ces modèles.
Les systèmes de versioning sont cependant plus adaptés au versioning de code que de modèles (d'équation) enrichis par des méta informations. A l'opposé les système d'information industriels sont bien adaptés à une gestion d'entités de type géométriques pouvant être classiquement enrichi par des annotations. L'utilisation de ces informations par d'autres que des humains (lecture des annotations) est difficile.

2.2 Recherche

L'objectif de la thèse est de définir un outil qui reprends les idées fondatrices du PLM dans sa version PDM tout en incorporant tout ce qui est nécessaire pour la gestion de conflits, les merge, … en s'inspirant de ce qui est efficace dans le monde du logiciel (code). Il sera surement utile de modéliser et définir un processus de refactoring et d'extraction de données dans les commentaires en lien avec l'ontologie définie.

On fera appel aux deux concepts suivants :

- Les ontologies avec un aspect modélisation du métier à partir des analyses faites avec les experts.
    - Modélisation du système actuel pour commencer à construire une ontologie du domaine de la simulation POWER chez INTEL. Construire le méta modèle associé à cette ontologie.
    - Faire un test modèle reel1 -> ontologies -> model reel 2 par application sur le système en prenant une entité dans une liste d'entité de l'ontologie préalablement définie.
    - Tester sur une base réelle de données
    - Définir les notions similaires aux concepts en système d'information
        - Fusion (merge)
        - Détection/résolution de conflit (semi-automatique, manuelle)
        - Dérivation de modèles
        - …
    - Extraction de concepts à partir des données dans les commentaires avec des patterns, converger vers une définition du corpus à utiliser
- Les techniques de type Machine Learning pour utiliser les techniques de classification lorsque la modélisation est imparfaite.

3. Lieux de travail. Entreprise. Contrat CIFRE

L'entreprise est une entreprise de la microélectronique ayant un département de développement en Isère.Le candidat effectuera ses travaux en se répartissant entre un centre de recherche développement de la région grenobloise et le laboratoire G-SCOP (Laboratoire Science de la conception, de l'optimisation et la production).

4. Compétences souhaitées

Compétences minimales en modélisation de systèmes d'information

Compétences en développement informatique

La compréhension des modèles de type circuit électrique et/ou de thermique est un plus (niveau prépa ou équivalent L3)

5. Contacts universitaires

Le premier contact se fera avec les universitaires de G-SCOP responsable de l'encadrement

Jean BIGEON, Directeur de recherche CNRS, jean.bigeon@grenoble-inp.fr

Lilia GZARA, maître de conférence, HdR, lilia.gzara@grenoble-inp.fr