Thèse d'Ali Koudri, Lille 1, 2010

Processus de Conception Conjointe Logiciel Matériel Dirigés par les Modèles

Thèse

Présentée à

L'Université des Sciences et Technologies de Lille

Pour obtenir le titre de

Docteur en Informatique

Par

Ali KOUDRI

Soutenue le 13 Juillet 2010
Devant la commission d'examen composée de :

Jean-Luc Dekeyser	Directeur	LIFL
François Terrier	Rapporteur	CEA LIST
Pierre-Alain Muller	Rapporteur	Université de Haute Alsace
Joël Champeau	Examinateur	ENSIETA
Denis Aulagnier	Examinateur	THALES
Éric Senn	Président	LabSticc

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Remerciements

Je remercie mon directeur, Jean-Luc Dekeyser, qui m'a donné l'opportunité de faire cette thèse dans une région lointaine, avec toutes les difficultés que cela entraîne pour gérer l'encadrement. Mais c'était sans compter sur le support de mes deux tuteurs de choix, Denis Aulagnier et Joël Champeau, qui m'ont tant apportés. Un grand merci à vous.

Je remercie les équipes qui m'ont accueilli au sein de THALES et de l'ENSIETA, je n'oublierai jamais les débats enflammés qui m'ont fait douter et qui ont permis de faire avancer la réflexion. Je remercie particulièrement Florence Cornic, responsable du laboratoire BTN à THALES, pour son accueil et sa bienveillance. Merci à Philippe Dhaussy, responsable du laboratoire DTN à l'ENSIETA pour son accueil et ses remarques pertinentes.

Je remercie tous les collaborateurs du projet MoPCoM : Philippe Soulard, Jean-Christophe Le Lann, Pierre-Laurent Lagalaye, Stéphane Lecomte, Frédéric Le Roy, Pierre Leray, Christophe Moy, Guy Gogniat, Jorgiano Vidal, Florent de Lamotte, Didier Vojtisek, Vincent Leilde et Onil Goubier. Merci pour toutes ces réunions riches en enseignements et qui sont pour beaucoup dans le contenu de cette thèse.

Je remercie mes rapporteurs, Pierre-Alain Muller et François Terrier, pour leurs remarques pertinentes qui m'ont permis d'améliorer la qualité de cette thèse.

Je remercie enfin ma famille et mes amis qui m'ont encouragé le long de cette thèse, en particulier mon épouse Yasmina, qui a eu le courage de supporter 3 longues années de séparation et mon fils Avicenne qui a su me tenir éveillé toutes les nuits durant les derniers mois de la rédaction.

J'adresse une pensée particulière à Joël Bohée et Bruno Bogaert sans qui rien de tout cela n'aurait pu arriver, à ma sœur Habiba et à ma tante Aïcha pour toutes ces années de galère traversées ensemble. Merci à toutes les autres personnes que je n'ai pas cité et qui se reconnaîtront dans ces lignes.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

À ma mère, Fahima

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Résumé

Aujourd'hui, les industries développant des systèmes sur puce doivent gérer le gap de productivité résultant de l'évolution rapide des technologies et de la pression croissante du temps de mise sur le marché. Dans ce contexte, la maîtrise des processus depuis la gestion des exigences jusque l'implantation du produit final est une nécessité pour favoriser la compétitivité. Les solutions pour traiter ce problème sont nombreuses et variées, et peuvent être de fait difficile à intégrer dans un processus. Dans cette thèse, nous proposons un processus formalisé de conception de système sur puce qui utilise les modèles pour mieux organiser les activités de l'ESL (Electronic System Level). Pour parvenir à ce résultat, nous proposons une contribution à la représentation des processus techniques ainsi qu'une contribution à la représentation des modèles de calculs, essentiel dans les activités de conception et d'analyse de l'ESL. Ces contributions sont validées par l'expérimentation sur un processus industriel de conception d'un système de "Cognitive radio" implanté sur une plateforme FPGA (Field Programmable Gate Array).

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Table des matières

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Table des figures

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Chapitre 1

Introduction

Les entreprises qui développent des systèmes sur puces sont de plus en plus confrontées à des problèmes de productivité liés à l'évolution rapide des technologies et la forte pression des marchés. Aujourd'hui, la maîtrise des développements de systèmes sur puce n'est majoritairement assurée que de manière ad-hoc : le savoir-faire de l'entreprise est capitalisé dans des documents textuels représentant le référentiel de l'entreprise, lequel accumule tous les savoirs des domaines transverses de l'entreprise (métier, gestion, qualité, etc.). L'ensemble du référentiel représente de fait une somme de documents importante, difficile à mettre à jour et à en garantir la cohérence face aux évolutions. En outre, la question du strict respect de ce référentiel, assurant la qualité des produits délivrés aux clients, reste difficile à établir du fait de la nature informelle des documents. Au delà du référentiel, le nécessaire respect de certains standards ou normes ajoute davantage de difficultés à la conduite des processus de développement de système en général et de systèmes sur puce en particulier. Ce type d'approche est aujourd'hui remis en cause dans les entreprises. Dans le but de relever le niveau des conceptions au niveau du système, plusieurs challenges scientifiques et techniques ont été posés. Parmi ces challenges, on retiendra en particulier la gestion de la complexité croissante et du changement d'échelle dans la conception mais aussi la nécessité de contrôler finement l'élicitation des processus.

C'est dans ce cadre que j'ai effectué ma thèse CIFRE au sein de l'entreprise Thales. Mon rôle a été d'investiguer la pertinence de l'utilisation de modèles pour améliorer la maîtrise des processus de conception de système sur puce et de proposer des solutions permettant d'améliorer la formalisation entre les parties hautes (exigences du système) et la partie basse des pro-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

cessus (implantation sur puce). Ce travail devait entre autres permettre de décharger au maximum les ingénieurs de tâches répétitives afin qu'ils puissent se focaliser davantage sur les problèmes d'analyses. Aussi, après une étude de l'état de l'art sur les propositions des communautés de l'ESL, de l'IDM (Ingénierie Dirigée par les Modéles) et de l'ingénierie des processus, j'ai pu identifier quelques points bloquants pouvant gêner l'adoption d'une approche IDM au sein des entreprises développant des systèmes sur puce.

1.1 Problématique

Ma première tâche lorsque je suis arrivé dans l'entreprise Thales a été d'observer les pratiques de conception d'applications du traitement du signal implanté sur des cartes FPGA (Plateforme reconfigurable matériellement). Ces activités constituent le cœur métier du laboratoire auquel j'étais rattaché.

La figure 1.1 illustre le type de développement matériel constaté sur le terrain. En bref, les ingénieurs matériels, à partir des exigences reçues, appliquent de manière ad-hoc le savoir-faire du référentiel pour transformer manuellement les exigences en un produit fini. Le référentiel impose un travail fastidieux de documentation ou de mise à jour de matrices d'exigences devant être approuvés par les paires du domaine lors des activités de revue.

La mise en œuvre des spécifications du référentiel sont très laborieuses, sources de beaucoup d'incompréhensions et d'erreurs. Les difficultés rencontrées sont essentiellement dues au cloisonnement fort entre les activités de l'ingénierie matériel et l'ingénierie système. Finalement, cette approche dirigée par le code et la documentation laisse peu de place aux activités d'analyse. À cela, il faut ajouter le problème du respect des normes et des standards qui rend d'autant plus difficile le travail des ingénieurs. Par exemple, l'impact de l'application les normes DO-178B et DO-256 dans le domaine de l'avionique est considérable. Le respect de ces normes impose une lourdeur qui augmente de manière significative les temps et les coûts de développement.

Aussi, les principaux problèmes des approches classiquement employés dans l'industrie sont :

- Les processus appliqués impliquent un trop grand gap entre les exigences et les produits délivrés, rendant la gestion des exigences très difficile,
- L'application des processus du référentiel est prise en charge par les ingénieurs de développement alors qu'ils ne devraient en être que les

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

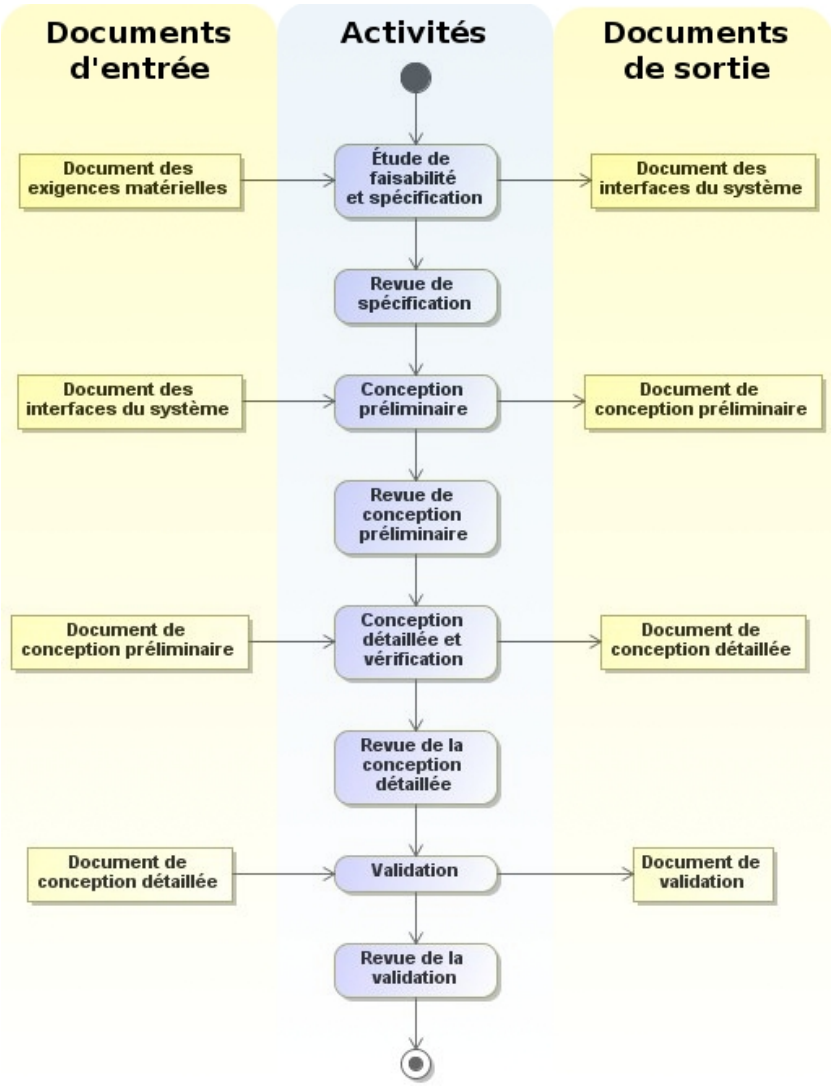


FIGURE 1.1 – Approche industrielle

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

acteurs,
- La maîtrise des artefacts de développement ne peut être assurée que de manière ad-hoc, conformément aux recommandations du référentiel,
- Il n'existe pas de séparation claire entre les aspects intentionnels des processus de leurs réalisations, ce qui rend le référentiel globalement dépendant des technologies et d'autant plus difficile à maintenir.

D'autres problèmes existent, comme par exemple celui du traitement des obsolescences, mais ils sont généralement considérés comme des effets de bords des problèmes mentionnés ci-dessus.

Il existe dans l'état de l'art un certain nombre de solutions traitant les problèmes mentionnés. À vrai dire, il existe même un grand nombre de solutions apportées par différentes communautés de recherche, quelles soient de l'ESL, de l'IDM ou de l'ingénierie des processus. Dans tout ce foisonnement de solutions, les choix sont difficiles car même si ces solutions se révèlent être efficaces pour traiter des problèmes précis, leur intégration au sein d'un processus de grande échelle peut poser des problèmes. L'interfaçage des métiers ou des activités du processus font partie des problèmes posés.

Dans un premier temps, j'ai proposé un processus qui fait la synthèse de techniques de l'ESL et de l'IDM. Dans cette proposition, je préconise l'utilisation de modèles en lieu et place des artefacts du processus décrit par le référentiel. Ce travail m'a permis de collaborer à la définition du profil UML MARTE pour la conception et l'analyse de systèmes temps réel embarqués, standardisé par l'OMG. Après avoir validé mon approche par l'expérimentation, je me suis posé la question de la capitalisation de mes propositions. Cette réflexion m'a amené à m'intéresser à la formalisation des processus.

Pour résumer, mes propositions consistent en un processus formalisé, basé sur l'utilisation des profils SPEM (System and Software Process Engineering Modeling) et MARTE (Modeling and Analysis of Real Time Embedded Systems) afin de mieux organiser les activités et les artefacts de la nébuleuse ESL (figure 1.2). Mes travaux de synthèse et de modélisation ont fait apparaître des manques comme la représentation des MoCs (Models of Computation) ou la gestion fine du cycle de vie des processus (en rouge sur la figure 1.2) dans la mise en œuvre des solutions existantes. En effet, l'utilisation des langages SPEM et MARTE, dans leur spécification actuelle, ne convient pas à une bonne organisation des activités du processus et de ses artefacts. Pour y remédier, il est nécessaire de :

- étendre SPEM pour favoriser l'intégration des techniques IDM,

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

– étendre MARTE pour améliorer son adéquation aux concepts de l'ESL.

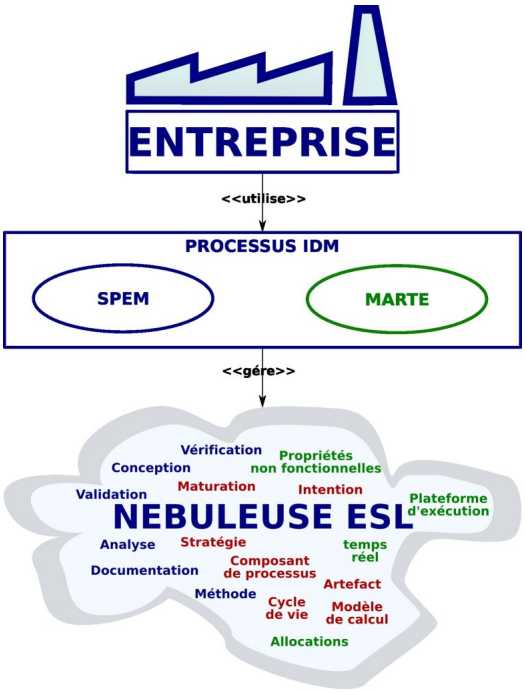


FIGURE 1.2 – Nébuleuse ESL gérée par les langages SPEM et MARTE

1.2 Solutions proposées

Les solutions que je propose visent à améliorer l'organisation des activités et des artefacts des processus de conception de systèmes sur puce par l'utilisation de modèles. Elles consistent en des extensions des langages SPEM et MARTE pour :

– Une meilleure intégration de l'ESL dans l'IDM,
– Une meilleure analyse des processus de codesign,
– Une meilleure intégration de l'IDM dans les modèles de processus.

Nous verrons dans la suite de ce document de quelle manière ces extensions permettent de mieux traiter les problèmes mentionnées (figure 1.3). En particulier, nous verrons dans le chapitre 4 de quelle manière l'extension COMETA complète le langage MARTE pour répondre au premier point. Nous verrons également dans ce chapitre comment l'extension MODAL complète le langage SPEM pour répondre au second et au troisième point. En-

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

fin, nous proposons dans le chapitre 5 un modèle de processus formalisé, nommé MoPCoM SoC / SoPC, permettant de valider par l'expérimentation ces propositions. Le modèle de processus MoPCoM SoC / SoPC constitue une contribution originale validée par une expérimentation industrielle servant de référence aux futurs développements de FPGA au sein de l'entreprise Thalès. Ce modèle explicite le cheminement des conceptions de système sur puce depuis un ensemble d'exigences formalisées jusqu'à l'implantation finale du système. Aussi, sa formalisation et son outillage contribuent à garantir que les bons concepts seront utilisés au bon moment et que le code ou la documentation seront générés conformément aux règles métiers de l'entreprise ou aux règles imposées par les standards et les normes.

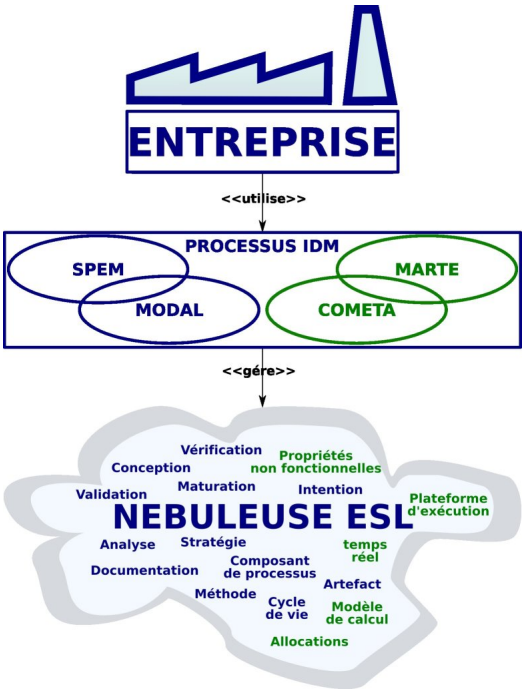


FIGURE 1.3 – Nébuleuse ESL gérée par les extensions MODAL et COMETA

1.3 Organisation du document

Le chapitre 2 présente de manière détaillée le contexte de cette thèse et les motivations de mes contributions. Le chapitre 3 présente l'état de l'art sur lequel je me suis basé pour proposer ces extensions. Ce chapitre présente un

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

rapide panorama des techniques IDM appliquées à la maîtrise des processus système et logiciel. Les chapitres 4 et 5 présentent les contributions de ma thèse qui sont :

- L'extension MODAL pour la représentation des processus,
- L'extension COMETA pour la représentation des modèles de calculs,
- Le modèle de processus MoPCoM SoC / SoPC dédié à la conception de systèmes sur puce.

Ces contributions font l'objet d'une analyse établissant la pertinence du travail réalisé dans le chapitre 5. Cette analyse permet de tirer un bilan de mes propositions et sur l'utilisation du profil MARTE pour le développement de systèmes sur puce. Enfin, la conclusion résume mes contributions, fait un point critique et ouvre de nouvelles perspectives sur le travail réalisé.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Chapitre 2

Contexte et motivations

Sommaire

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

2.1 Introduction

Je présente dans ce chapitre le contexte des travaux de la thèse et les définitions sur lesquelles je me suis basé pour construire ma réflexion. Dans un premier temps, je montre que la maîtrise de processus de conception sur puce représente un enjeu crucial. À cette fin, je présente un flot typique de conception de SoC qui permet de faire ressortir les problèmes traités dans cette thèse. Ensuite, je fais un tour d'horizon des solutions apportées par les communautés de l'ESL, de l'ingénierie des processus et de l'IDM à la maîtrise des processus. Cela me permettra d'établir, à la lumière des problèmes exposés et des solutions proposées, les défis à relever dans cette thèse.

2.2 De la difficulté de concevoir des SoCs

2.2.1 Les systèmes sur puce

Depuis quelques années, le marché des systèmes sur puce (SoC, System-on-Chip) connait une forte croissance. Il est prévu par ailleurs que ce marché pèse la somme de 56 milliards de dollars en 2012, ce qui représente une croissance moyenne annuelle de 24%.

FIGURE 2.1 – Exemples d'application des SoCs

Le domaine des SoCs touche un nombre de secteurs croissant (figure 2.1) comme l'automobile, l'aérospatial, la téléphonie mobile ou même l'électroménager.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Il existe dans la littérature plusieurs définitions des SoCs (System-On-Chip – Systèmes sur puce). La définition que je choisis dans le cadre de cette thèse est celle proposée dans [NN03] :

> "A SOC is a system on an IC that integrates software and hardware Intellectual Property (IP)"

Les systèmes sur puce sont des circuits intégrés qui implantent des systèmes complets. Ils sont constitués d'une hiérarchie de blocs hétérogènes, matériels ou logiciels, analogiques ou numériques, et qui représentent autant de sous-systèmes. Les blocs qui constituent la hiérarchie du système sont généralement désignés sous le terme de propriétés intellectuelles (IP – Intellectual Property).

Dans le détail, un SoC est un circuit intégré typiquement constitué de ressources de (figure 2.2) :

- calcul (processeur, FPGA, DSP ou microcontrôleur),
- mémorisation (RAM, ROM ou Flash),
- communication (bus AMBA, PCI, ...)
- temps (horloges),
- conversion analogiques/numériques,
- entrées / sorties (Ethernet, FireWire, etc.),
- gestion de l'énergie.

Les FPGAs sont des cas particuliers de SoC qui intègrent des parties reprogrammables matériellement. Ils sont constitués d'un ensemble de blocs logiques organisés en matrice. Chacun de ces blocs contient des éléments de calcul, des registres et des mécanismes supportant la reprogrammation (LUT – Look-Up Table) (figure 2.3). La logique des programmes exécutés sur les FPGA est dite "câblée" car les programmes qui y sont implantés peuvent être représentés sous la forme d'un plan de blocs fonctionnels reliés par des fils.

L'implantation de systèmes sur du matériel reprogrammable présente un certain nombre d'avantages (minimisation des risques de développement, possibilité de faire du prototypage rapide, etc.) et d'inconvénients (consommation énergétique accrue, rapidité d'exécution plus limitée que celle des ASICs – Application Specific Integrated Circuit, etc.). De par leurs propriétés, les FPGAs sont généralement réservés à des marchés de niche ne nécessitant pas de grosses productions, comme le domaine militaire ou l'aérospatial.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

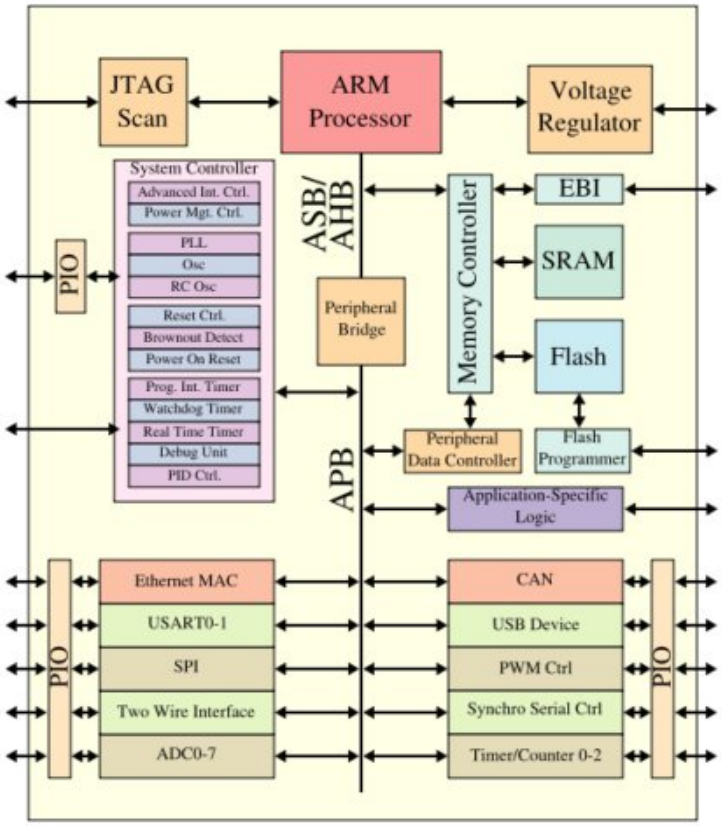


FIGURE 2.2 – Structure typique d'un SoC

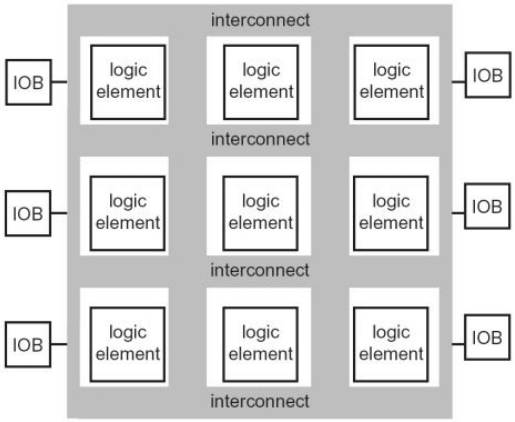


FIGURE 2.3 – Structure d'un FPGA

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

2.2.2 Les pratiques actuelles de conception de SoCs

Dans les pratiques actuelles de conception de systèmes sur puce, les équipes système font l'interface entre les clients et les équipes de développement. Ils établissent avec le client le cahier des charges et les exigences du système (exigences de haut niveau). Les exigences peuvent être classées en trois catégories :

- Les exigences fonctionnelles, relatives aux fonctions du système ("détecter une cible"),
- Les exigences non-fonctionnelles, relatives aux contraintes du système ("le système doit consommer moins de 5kW/h"),
- les exigences de processus portant sur le processus de développement ("le système doit être développé dans un an et livré avec son guide d'utilisation").

À partir des exigences fonctionnelles, les équipes d'ingénierie système établissent les interfaces du système qu'ils valident avec le client. À partir des exigences non-fonctionnelles, ils discutent avec les équipes d'ingénierie logicielle ou matérielle pour établir des exigences dérivées (exigences de bas niveau) conformément à l'état de l'art des technologies du moment. L'activité qui consiste à faire un choix d'implantation en logiciel ou matériel pour chaque fonctionnalité du système est généralement désignée sous le terme de "partitionnement". Le choix du partitionnement dépend de plusieurs facteurs : la qualité de service, la sûreté de fonctionnement, le besoin en performance ou en flexibilité, etc.

Les exigences matérielles sont transformées en architecture RTL (Register Transfert Level) constituée d'une hiérarchie de blocs matériels, et identifiés par leurs interfaces et leurs protocoles de communication. Pour réaliser ces activités, l'ingénierie matérielle utilise un ou plusieurs langages de description matérielle (VHDL, Verilog) supportés par différents outils de conception, de simulation, de vérification et de synthèse. Les exigences logicielles sont quant à elles généralement traduites en artefacts de langages de haut niveau (C, C++) décrivant des architectures logicielles qui seront ensuite compilées pour les plateformes d'exécution retenues, y compris les plateformes d'abstraction matérielles telles que les systèmes d'exploitation (VxWorks, Linux) ou les middlewares (CORBA). À l'issu de l'implantation du système, un ensemble de tests (tests unitaires, tests d'intégration, tests de robustesse) sont exécutés afin de vérifier le bon comportement du logiciel et du matériel, et la correction

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

fonctionnelle. Dans la pratique, la partie test peut prendre plus de 70% du temps de développement. Enfin, des activités de validation ont pour but d'établir les équivalences entre les exigences et le produit.

Avec l'évolution des technologies et l'accroissement exponentiel du nombre de ressources disponibles, la validation de tels systèmes au bit près et au cycle près (cycle accurate bit accurate) est devenue très laborieuse et consommatrice de temps. Et si les analyses conséquentes aux choix d'un partitionnement ne sont pas jugés satisfaisantes, il faut choisir un autre partitionnement et s'engager dans un nouveau processus de vérification / validation consommateur en temps et en ressources.

2.2.3 Les problèmes posés par les pratiques actuelles

Conformément à la loi de Moore, l'évolution rapide des technologies a permis aux fournisseurs de technologies de disposer d'un nombre croissant de ressources matérielles, toujours meilleur marché et plus rapide. Cette évolution a permis de proposer un plus grand nombre de fonctionnalités, toujours plus complexes.

Face à une telle évolution, les méthodologies classiques ne sont plus adaptées car elles n'ont su gérer convenablement la nécessaire montée en abstraction et il en résulte aujourd'hui une crise liée au gap de productivité (cf. figure 2.4) accentuée par la pression due à la forte compétitivité des entreprises [ITR07].

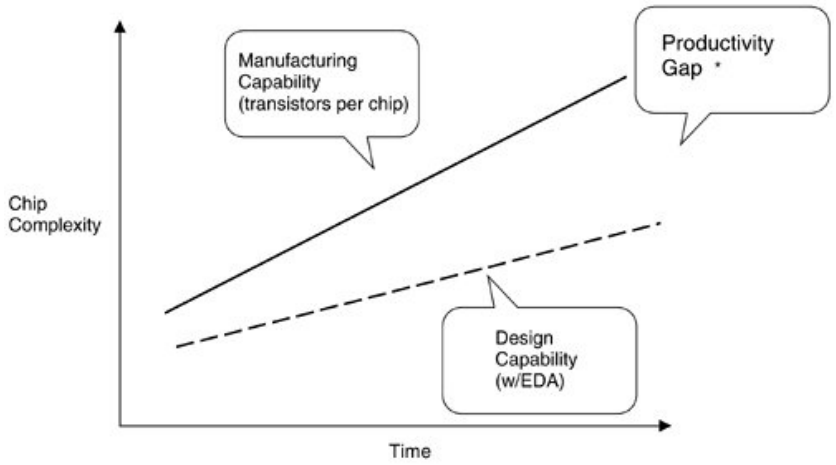


FIGURE 2.4 – Gap de productivité

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

La conception de SoCs nécessite l'exécution d'un ensemble d'activités de conception, d'analyse, de tests ou de certification depuis la capture des exigences du client jusqu'à la livraison, la maintenance du produit et son retrait. L'exécution de ces activités implique des interactions entre un ensemble de métiers, de compétences, de cultures, d'outils ou de langages qui rendent la conception très complexe. À cela, il faut ajouter le respect des standards et des normes de développement ou de gestion. Par exemple, dans le domaine de l'avionique, le problème de l'application des normes de certification provient de la lourdeur et le coût de leur mise en place. En effet, on estime que la mise en place d'un processus conforme à ces normes implique un surcoût financier de 75 à 150 % et des durées de développement d'autant plus longues. Il résulte de tous ces facteurs de grandes difficultés pour gérer de manière rationnelle les développements de SoCs.

Pour traiter ces problèmes, les ingénieurs appliquent généralement le référentiel d'entreprise dans lequel sont capitalisés les savoirs acquis au fil des expériences ou des formations. Ces référentiels sont constitués de guides, de méthodes ou de tutoriels spécifiés de manière informelle dans des documents textes. De fait, ces approches essentiellement dirigée par les documents ("Document Driven Approach") sont très lourdes à mettre en œuvre et à faire évoluer.

Afin de surmonter toutes ces difficultés, il est nécessaire d'appliquer des méthodologies de développements plus rigoureuses qui permettent de concevoir des SoCs conformes aux exigences du client dans les temps et les budgets requis. La définition de telles méthodologies doit prendre en compte les facteurs d'échecs récurrents afin de les anticiper. Dans ce contexte, l'adoption d'un processus formalisé [M.S03] permettrait de mieux gérer les activités de conception et d'analyse.

2.3 Vers une meilleure maîtrise des processus

2.3.1 Les réponses de l'ESL aux pratiques actuelles

Avant de présenter les propositions de la communauté ESL pour améliorer la maîtrise des processus de codesign, je me propose de choisir une définition de l'ESL (Electronic System Level). Dans [BGA07], l'ESL est définit comme suit :

"Utilization of appropriate abstractions in order to increase com-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

> prehension about a system, and enhance the probability of a successful implementation of functionality in a cost effective manner, while meeting necessary constraints"

L'ESL se définit par l'utilisation d'abstractions appropriées permettant d'améliorer la compréhension des systèmes et d'augmenter la probabilité d'implanter correctement leurs fonctionnalités au regard des contraintes de conception imposées. J'ai choisi cette définition car elle fait bien ressortir les relations entre la notion de système et les niveaux d'abstractions permettant de mieux gérer la complexité et les risques de conception.

L'ESL propose des techniques permettant de traiter le problème du gap de productivité et d'accélérer les développements conjoints de logiciel / matériel. Dans les paragraphes suivant, je présente les principales techniques de l'ESL qui sont exploitées dans l'approche IDM que je propose dans cette thèse.

Montée en abstraction

Les approches classiques de développement de SoC possèdent de nombreux inconvénients, en particulier dans la gestion des exigences. En effet, la compréhension du besoin et la validation des exigences dans ces approches ne se peut se faire qu'à travers les implantations du système. Aussi, dans la mesure où les choix de conception sont très vastes et les temps de vérification très long pour valider LA solution, les approches ESL se justifient par la proposition de techniques permettant d'explorer rapidement l'arbre des possibles afin d'éliminer un maximum de solutions inaptes à satisfaire complètement les exigences, tant fonctionnelles que non-fonctionnelles, dans les temps et les budgets requis.

Dans cet esprit, il est intéressant de pouvoir construire des systèmes par diverses représentations qui intègrent de plus en plus d'exigences, tant au niveau structurel qu'au niveau comportemental. Cette approche incrémentale nécessite l'utilisation de langages dédiés à chaque niveau d'abstraction. Elle permet en outre de fournir aux clients des modèles exécutables afin de mieux maîtriser les risques et d'éviter des dépenses inutiles liées à l'inadéquation ou la non-faisabilité des exigences [Kra03].

La figure 2.5 représente un flot typique ESL. Elle illustre bien les évolutions qui ont été apportées par la communauté de l'ESL. Entre autre, cette figure illustre un des principes de base de l'ESL qui est de retarder au maximum les choix du partitionnement. Cela permet d'une part d'éviter un effet

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

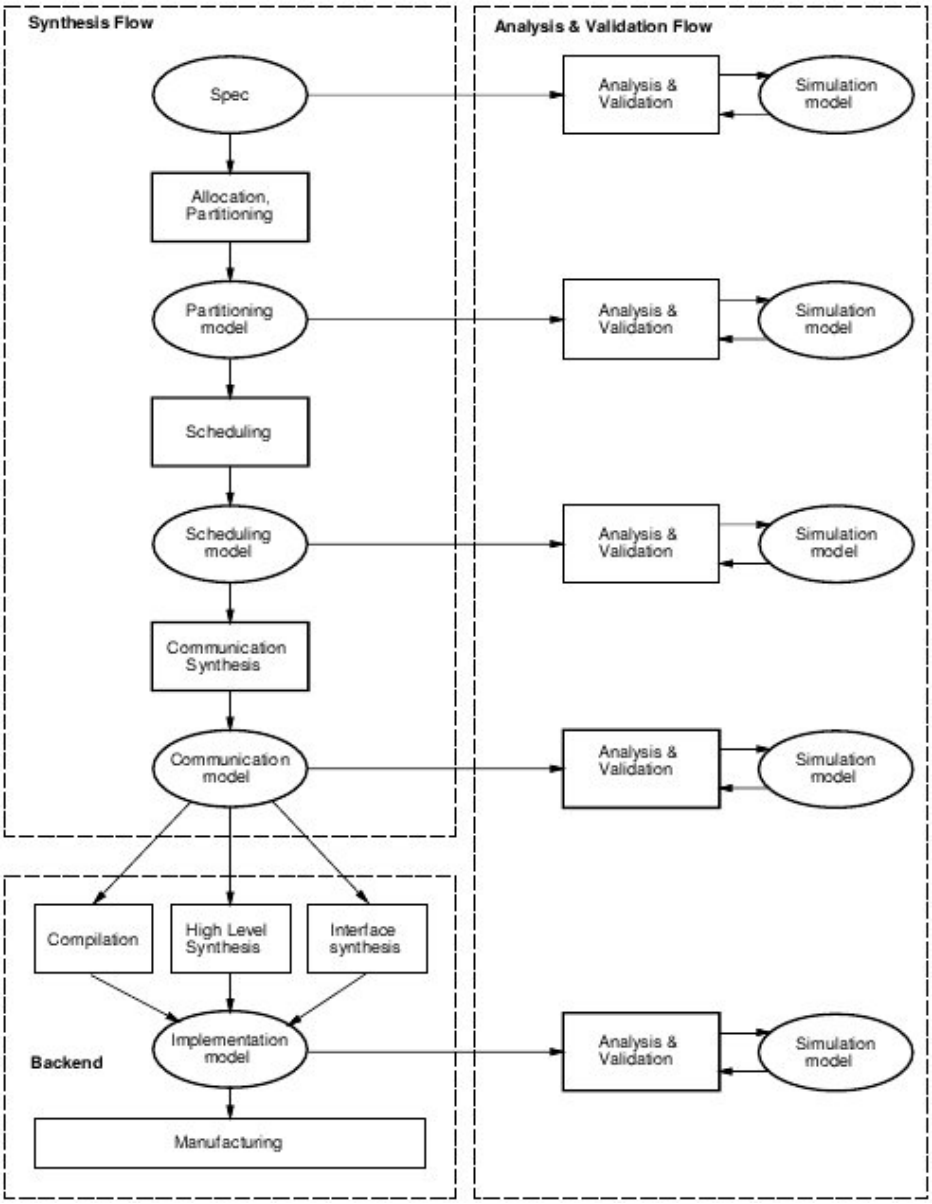


FIGURE 2.5 – Évolution des flots de codesign

boule de neige dû aux mauvais choix de partitionnement ; et d'autre part d'unifier le développement conjoint de logiciel/matériel et de lever les verrous de communications entre les équipes de développement matériel et logiciel.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Orthogonalisation des préoccupations

Pour décrire un système, il existe plusieurs manières de procéder :
- on peut le décrire de manière informelle, par la parole ou par des descriptions textuelles,
- par des schémas ou des croquis,
- par un langage informatique compréhensible par une machine.

Loin de s'exclure, ces représentations sont en fait complémentaires : un homme pourra difficilement deviner l'architecture d'un système en regardant du code VHDL et il est difficile de faire interpréter un schéma griffonné sur un bout de table à un ordinateur.

Que ce soit dans l'IDM ou dans l'ESL, la description d'un système sert avant tout son analyse. Par ailleurs, le terme "analyse" signifie étymologiquement décomposition. On constate en effet dans la pratique qu'il est plus facile de gérer la complexité d'un système à travers sa décomposition. On peut constater également que les produits complexes vendus sur le marché ne sont finalement jamais que des assemblages de systèmes plus simples. Aussi, les problèmes rencontrés dans la construction de systèmes complexes proviennent essentiellement de la gestion de l'hétérogénéité des sous-systèmes et de leurs interactions. Par exemple, les retards pris sur la construction de l'airbus A380 étaient essentiellement dues à des problèmes d'interfaçage.

La gestion de l'hétérogénéité est un des plus importants problèmes identifiés par la communauté de l'ESL. Afin de favoriser l'analyse de systèmes complexes qui mêlent des sous-systèmes hétérogènes, il est nécessaire d'identifier les détails qui caractérisent l'hétérogénéité et d'orthogonaliser les préoccupations. Dans l'ESL, les détails relatifs à la caractérisation des MoCs sont liés à la représentation des données, du temps, du calcul ou des communications. Ces caractéristiques définissent ce que l'on nomme "le modèle de calcul".

Dans [Jan04], l'auteur propose un méta-modèle qui sert à caractériser les modèles de calcul suivant les axes du calcul, de la communication, des données et du temps. Ce méta-modèle, nommé rugby, est schématisé par la figure 2.6. Les critères d'orthogonalisation choisis permettent d'établir une taxonomie pertinente des modèles de calcul et des langages qui servent à les exprimer. Le principe de ce modèle consiste à considérer que durant le développement d'un système, depuis l'idée jusqu'à sa réalisation, un certain nombre de modèles exprimés par différents langages sont utilisés à différents niveaux d'abstraction. Le raffinement des modèles, qui est guidé par les besoins en analyses,

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

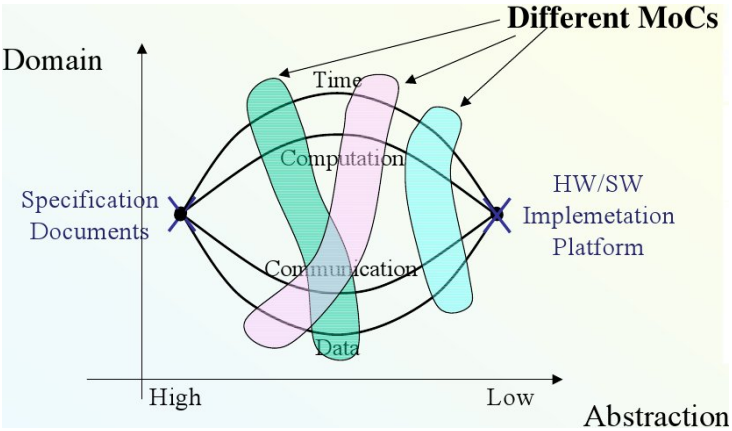


FIGURE 2.6 – Métamodèle Rugby

revient alors à détailler leurs caractéristiques suivant chacun de ces axes (domaines).

Le domaine du calcul, aussi désigné sous le terme fonction ou comportement, consiste à établir des relations entre les entrées et les sorties du système. Les relations peuvent être par exemple des fonctions mathématiques, des équations booléennes, une suite d'instructions machines (code assembleur) ou un algorithme.

Le domaine des données traite la représentation des données physiques ou logiques. Par exemple, au niveau du système, on parlera davantage de données physiques (20 kilogrammes, 36 km/h, 3 ms). Au niveau logique, on parle de types de donnée abstraits (entiers, réels sur 32 ou 64 bits). Au niveau matériel, on parle de vecteurs de bits représentant basiquement le nombre de fils nécessaires pour faire transiter les commandes et les données.

Le domaine du temps traite la représentation du temps, caractéristique dominante des systèmes embarqués temps réel. La forme la plus simple du temps est le temps causal qui établit des relations de précédence entre les événements. On distingue ensuite le temps discret (temps cadencé) par lequel les comportements du système sont reliés à des horloges de référence communes (synchrone) ou non (asynchrone) mesurant la progression du temps à intervalle régulier. Enfin, on distingue le temps réel au sens physique du terme et dont la définition dépasse tout simplement le cadre de cette thèse et du codesign en général. Cependant, on peut mentionner que certains langages, comme le profil UML MARTE ou MATLAB, choisissent de considérer le temps continu comme un temps discret dense.

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Le domaine de la communication traite les mécanismes permettant de faire interagir plusieurs entités. Suivant le niveau d'abstraction où l'on se place, les communications revêtent différentes formes. Par exemple, au niveau logiciel, les communications se traduisent en appels de méthodes alors qu'au niveau matériel, elles se matérialisent sous la forme de bus de communication caractérisées par des propriétés physiques (longueur, dissipation énergétique, etc.) avec des accès gérés par des protocoles souvent complexes nécessitant des politiques d'arbitrage.

L'approche "Platform Based Design"

Une des manières de répondre aux besoins de productivité est de favoriser au maximum la réutilisation de blocs de nature ou de niveaux d'abstraction différents (systèmes, logiciels ou matériels). La notion de bloc réutilisable est généralement désignée sous le terme de "Intellectual Property". Les propriétés intellectuelles (IP) constituent des solutions sur étagères pouvant être réutilisées, vendues ou échangées.

Outre les aspects légaux associés à la réutilisation, l'assemblage d'IPs pose un certain nombre de problèmes techniques. En effet, les blocs IP ne peuvent être assemblés que si :

- ils sont de même nature (physique ou logique),
- ils sont définis au même niveau d'abstraction,
- ils possèdent des interfaces et des protocoles de communication compatibles.

Ces problèmes soulèvent la nécessité de standardiser les niveaux d'abstraction [Lah06], les interfaces ou les protocoles. Dans cette idée, il existe aujourd'hui 3 consortiums à but non-lucratif qui travaillent dans ce sens :

- Open SystemC Initiative (OSCI) [osc] œuvre à la standardisation des niveaux d'abstraction et est à l'origine du langage SystemC,
- Open Core Protocol - International Partnership (OCP-IP) [ocp] œuvre à la définition un bus d'interconnexion matériel standard,
- SPIRIT [spi] - Structure for Packaging, Integrating and Reusing IP within Tool-flows (SPIRIT) œuvre à la standardisation de la description des IPs afin de faciliter leur échange, leur configuration et leur intégration dans les outils (IP-XACT).

Dans [CCH+99], les auteurs poussent la réflexion sur la réutilisation plus loin et suggèrent la réutilisation de plateformes vérifiées et documentées com-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

plètes : c'est l'approche "Platform Based Design". Une définition plus précise de cette approche est donnée dans [Bai05] :

> "Integration oriented design approach emphasising systematic reuse, for developing complex products based upon platforms and compatible hardware and software virtual components, intended to reduce development risks, costs and time to market"

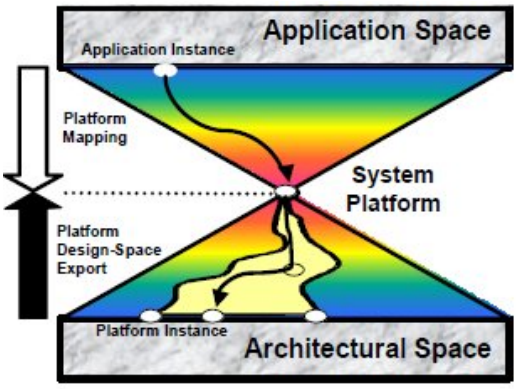


FIGURE 2.7 – Approche Platform Based Design

Comme le souligne cette définition, la réutilisation de plateformes et de composants matériels et logiciels vérifiés permet de limiter les risques et d'augmenter la productivité. De fait, une telle approche ne peut ni être de type "top-down" car les blocs matériels et logiciels existent déjà, ni même de type "bottom-up" car la plateforme d'exécution existe également. Dans ce cas, les méthodologies qui supportent cette approche sont de type "Meet in the middle" (figure 2.7).

Le principe de cette approche consiste essentiellement à rechercher une allocation spatiale et temporelle optimale des blocs logiciels ou matériels sur une plateforme contenant des composants reprogrammables en logiciel ou en matériel. Le choix d'une plateforme de déploiement est déterminé par le domaine d'application, les services et les qualités de service offerts par la plateforme et d'autres critères non fonctionnels tels que le prix, le poids ou la consommation. Le marché visé a également son importance. Par exemple, le marché militaire justifie largement le choix de plateformes FPGA pour ses applications. Outre les bénéfices apportés par cette approche, son application correspond dans les faits à une pratique industrielle bien ancrée.

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

2.3.2 Apports de l'ingénierie des processus

Le Standish Group fournit dans son rapport de 2006 [sta06] des statistiques détaillées sur les résultats de la conduite de projets dans les entreprises.

Il ressort de ces études que :

- 31,1% des projets échouent durant leur exécution,
- 52,7% des projets coûtent en moyenne 189% plus cher que prévu,
- 16,2% des projets tiennent le budget et les délais mais seulement 42% des fonctionnalités demandées sont effectivement implantées.

Bien évidemment, ces statistiques sont à prendre avec précaution. La réalité montre que, généralement, plus la taille des organisations est grande, plus les risques sont élevés.

Toujours selon ce rapport, les principales raisons expliquant ces échecs sont :

- Un manque d'implication du client dans les processus,
- Une mauvaise vision des processus impliquant de mauvaises décisions,
- Une mauvaise gestion des exigences ou l'acceptation d'exigences irréalisables,
- Un manque de réalisme sur l'adéquation entre les ressources disponibles et la taille des projets.

À la lumière de ces études, il apparait que seul le choix d'une méthodologie claire et cohérente permettrait de garantir la qualité du produit, c'est à dire sa conformité aux spécifications établies avec le client, dans un temps et à un coût raisonnable. Une telle méthodologie permettrait non seulement de guider et de borner le cycle de développement du produit, mais également de gérer les risques de manière efficace et d'éviter ainsi les échecs liés à de mauvais choix comme cela se produit encore bien souvent.

Dans le cadre du codesign, le développement de produits complexes mêlant du matériel et du logiciel est un processus généralement long et coûteux faisant interagir un certain nombre de compétences, de langages ou d'outils. De fait, cela représente une activité compliquée, qui, si elle est mal gérée, peut amener d'une part à devoir renégocier les délais et les budgets avec le donneur d'ordre, et peut d'autre part générer des défauts de conception potentiellement catastrophiques (crashs aériens par exemple).

Les difficultés rencontrées dans l'élicitation des processus de codesign proviennent essentiellement de la gestion des interactions entre les parties prenantes du processus. En effet, la profusion des métiers, des langages et des

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

outils utilisés dans les développements de tels systèmes est source de beaucoup d'incompréhensions et d'erreurs. Aussi, une image classiquement utilisée pour illustrer ce propos est l'image de la tour de Babel (figure 2.8) : la tradition judéo-chrétienne raconte que le roi Nemrod voulait construire une tour assez haute pour côtoyer le divin mais que Dieu, ne le voyant pas du même œil, introduisit plusieurs langues afin que les hommes ne se comprissent plus et que la construction soit abandonnée. Dans le domaine de l'ingénierie des processus, cette allégorie souligne le risque de voir un projet échouer quand les différents métiers impliqués ne parlent que le seul jargon de leur discipline.

FIGURE 2.8 – Allégorie de la tour de Babel

La compréhension des attentes de chacun favorise les bonnes prises de décision et il est important de fluidifier les communications à travers une culture d'entreprise commune. Dans les pratiques actuelles, les communications entre les différents métiers se matérialisent généralement par des échanges d'artéfacts de différentes natures (documentation, code sources, schémas) et les efforts déployés pour maintenir de l'ensemble des données est généralement considérable et coûteux.

Ainsi, les efforts pour améliorer la productivité et la qualité des processus ont été à la source de l'avènement d'une nouvelle discipline : l'ingénierie des processus. Cette discipline a pour objectif de rationaliser ce qui était auparavant de l'ordre de l'intuition et permettre ainsi de répondre aux problèmes posés par la maîtrise des processus. Le but recherché par les travaux autour

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

de la formalisation des processus est leur rationalisation. À travers ce terme, il faut entendre l'ajout de propriétés garantissant de manière sûre et reproductible la construction de produits conformes aux attentes des clients. Par ailleurs, la satisfaction de ces propriétés est au cœur de la maturité des processus telle que définie par le CMMI [CKS03] et constitue un pari à relever par l'approche dirigée par les modèles.

La rationalisation des processus passent par leurs formalisations à travers des notations adaptées possédant une sémantique clairement définie. Bien évidemment, les détails de la capture des processus de développement dépendent du domaine visé. Par exemple, dans le domaine du développement logiciel, l'auteur de [Mad91] expose les détails pertinents devant obligatoirement être représentés dans les modèles de processus de développement logiciel :

- Les étapes du processus,
- Les responsabilités,
- Les entrées/sorties des activités,
- Les conditions d'activation/terminaison d'une activité (pré/post conditions),
- L'état d'un artéfact avant, pendant et après l'exécution d'une activité,
- L'ordonnancement des activités (séquencement, parallélisation),
- etc.

Ainsi, ces représentations permettent :

- Une meilleure maîtrise des processus intellectuels de conception et d'analyse par l'identification claire des activités, des artefacts d'entrée/sortie ou des responsabilités,
- L'assurance de la qualité des produits en adéquations avec l'amélioration des processus,
- Une gestion optimisée de l'utilisation des ressources dans l'espace et dans le temps comprenant un ordonnancement optimal des activités,
- Une amélioration continue des processus de développement par des analyses statiques ou dynamique grâce à l'exploitation des métriques collectées lors de précédentes exécutions,
- La vérification systématique de la cohérence et de la correction des processus au regard des objectifs à atteindre,
- L'automatisation des tâches routinières par leurs expressions en langage compréhensible par les machines,
- Une meilleure maîtrise de la complexité et du passage à l'échelle.

Les apports de l'ingénierie des processus ont surtout été démontrés dans

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

le domaine du développement logiciel [GMP+94, Mad91] et je suis convaincu que l'ingénierie des processus peut être étendue au processus de codesign avec le même succès. C'est ce que je m'efforcerai de montrer dans le reste de ce document.

2.3.3 Apports de l'IDM

L'ingénierie des modèles (IDM) est un paradigme basé sur l'utilisation intensive de modèles et promouvant la séparation des préoccupations afin de mieux faire face à la complexité de la conception et de l'analyse des systèmes d'information. Dans l'IDM, les modèles sont au cœur des processus de développement car ils offrent des facilités pour la représentation et l'échange d'informations sur les caractéristiques pertinentes des systèmes étudiés. L'utilisation de modèles permet notamment de :

- Améliorer de manière significative la communication entre les différents métiers impliqués dans les développements,
- Fournir différents points de vue des systèmes étudiés (SUS – System Under Study) suivant les préoccupations adressées,
- Favoriser les processus d'analyses par la représentation des SUS à différents niveaux d'abstraction,
- Traiter les problèmes de portabilité et d'interopérabilité par la séparation des aspects métiers et technologiques,
- Capitaliser les expertises à travers les transformations de modèles et réduire les points de rupture technologiques,
- Automatiser un maximum de tâches routinières (analyses, codage, documentation, etc.).

L'idée principale des approches IDM est que l'on peut utiliser des modèles pour représenter les systèmes étudiés ainsi que tous les aspects liés au développement d'un système, tant techniques que managériales. Par exemple, la difficulté de la mise en place d'un processus formalisé prenant en charge le respect des standards et la conformité aux différentes normes provient de la lourdeur et du surcoût que cela engendre. En adoptant une approche IDM, on peut envisager d'automatiser l'ensemble du processus par l'identification claire de toutes les étapes de conception et d'analyse, des méta-modèles et des règles de transformations pour passer d'une étape à une autre.

En effet, dans la mesure où les modèles et les transformations sont au cœur de la définition de l'ingénierie des modèles, ils sont à même de capitaliser les

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

savoir-faire de l'entreprise. Mieux, ils peuvent être utilisés pour :

- gérer les interactions entre les parties prenantes du processus,
- gérer le cycle de vie des produits,
- évaluer les risques et de mettre en œuvre les actions de correction quand cela est nécessaire.

Par ailleurs, selon l'auteur de [Rol98], un processus de développement d'un système d'information peut être vu comme un chemin allant d'un point initial (expression des exigences) jusqu'au point final (produit final) en passant par les différentes positions intermédiaires (produits intermédiaires). Le passage d'un point à un autre correspond alors à une transformation.

Ainsi, l'approche IDM répond très clairement aux problématiques des processus de développement certifiables par le passage d'un monde informel, basé sur des documents de référence décrivant les règles permettant de gérer le processus de développement, à un monde formel où on réifie ces règles pour les incorporer dans des outils par l'utilisation de langages permettant de capturer les caractéristiques pertinentes du système en étude. Mais alors, les langages choisis doivent réifier les concepts nécessaires à la bonne conduite des activités de conception ou d'analyse du processus de développement. Dans les sections suivantes, nous présentons succinctement les langages de modélisation SPEM et MARTE que nous avons utilisés dans nos expérimentations et qui ont servi de base à nos contributions.

SPEM

SPEM est un langage permettant de décrire des processus de développement système ou logiciel par un ensemble de notations graphiques. Il est standardisé par l'OMG sous la forme d'un méta-modèle et d'un profil UML. La définition de ce langage est basée sur le principe qu'un processus de développement est régi par la collaboration d'entités abstraites (*process roles*) qui effectuent des opérations (*activities*) sur des entités concrètes (*work products*) (figure 2.9). À la manière de la norme ISO 12207 dont elle s'inspire, cette spécification a été écrite pour supporter un large éventail de processus de développement système ou logiciel.

En bref, le langage SPEM est organisé autour de 7 paquetages dont nous donnons une rapide description dans les lignes qui suivent. Le paquetage *Core* regroupe les concepts abstraits de base du langage. Il offre notamment aux méthodologistes les mécanismes d'extension permettant l'addition de nou-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

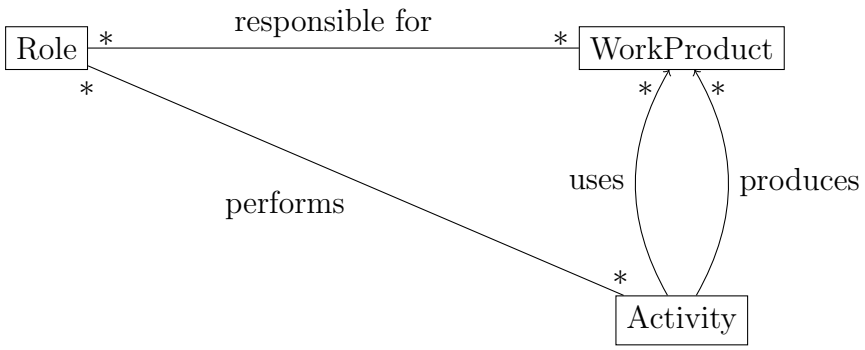


FIGURE 2.9 – SPEM, concepts de base

veaux types. Le paquetage *Process Structure* regroupe les définitions de base pour la représentation des processus (activités, rôles, artéfacts). Le paquetage *Process Behavior* offre les mécanismes de mise en relation des activités et permet leur spécification comportementale par la réutilisation des définitions des concepts d'UML. Le paquetage *Managed Content* définit les mécanismes permettant la gestion et la documentation des processus. La justification de ce paquetage est que, bien souvent dans la pratique, il est plus important de posséder une méthodologie bien documentée qui capitalise l'ensemble des bonnes pratiques qu'un modèle précis. Le paquetage *Method Content* permet de construire des bases de connaissances réutilisables et indépendantes du cycle de vie des processus. Il participe à la séparation entre les notions de méthodologie et de processus. Le paquetage *Process With Methods* fait le lien entre les méthodes et les processus. Enfin, le paquetage *Method Plugin* traite la gestion des processus à travers la notion de librairie de processus configurable mais aussi, et surtout, à travers la notion de composant de processus.

Les principaux cas d'utilisation du langage SPEM sont :

- Le support pour la représentation et la gestion de librairies de méthodologies réutilisables permettant de capitaliser les bonnes pratiques à l'intention des futurs développements. Cela englobe la définition de guides méthodologiques ou de procédures internes. Tous ces éléments représentent autant de bases de connaissances dont la représentation standard facilite la réutilisation ou l'échange.
- Le support pour la gestion des processus offre les mécanismes d'application des méthodes aux processus de développement incluant de

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

possibles adaptations.
- Le support de la configuration des méthodes et des processus fournit la prise en compte du contexte des organisations et des affaires dans la configuration des méthodes et des processus.
- Le support pour l'élicitation des processus de développement guidant la mise en œuvre des modèles et son impact sur les pratiques quotidiennes dans la vie de l'entreprise.

Enfin, les principaux concepts définis par SPEM sont résumés dans le tableau 2.1.

Concept	**Description**
process performer	permet de définir un responsable pour une activité donnée
guidance	fournit des informations plus détaillées à l'usage des acteurs. Il peut s'agir de guides, de référentiels, de techniques, de métriques, de méta-modèles, ...
work product and work product kind	artefact pouvant être produit, consommé, transformé par un élément actif et son type.
work definition	description d'un travail réalisé pendant le processus. Il peut s'agir de la description d'une phase, d'une itération, d'une activité décomposée en étapes,...
process component	description d'une partie consistante d'un processus pouvant être réutilisée.

TABLE 2.1 – Principaux concepts de SPEM

MARTE

Le profil UML MARTE est un profil qui traite la modélisation et l'analyse de systèmes embarqués temps réel. Le langage est né suite à une RFP (Request For Proposal) de l'OMG. Le profil MARTE avait notamment pour objectif de combler les lacunes du profil standard SPT (Scheduling, Performance and Time) mais aussi de réutiliser et de simplifier les concepts présents dans d'autres profils. Par exemple, les notions de qualité de service présents dans MARTE sont issues du profil standard QoS&FT (Quality of Service and Fault Tolerance). Le profil MARTE est construit autour de 3 paquetages (2.10) favorisant la séparation des préoccupations :

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

– le paquetage “foundation” fournit les bases du langage qui traitent la modélisation des propriétés non-fonctionnelles, du temps, des ressources génériques et des allocations,
– le paquetage “design” permet de modéliser des plateformes d'exécution matérielles ou logicielles à différents niveaux d'abstraction,
– le paquetage “analysis” fournit les mécanismes permettant d'annoter les modèles à des fins d'analyse. Il fournit les concepts génériques pour couvrir tout type d'analyse et couvre nativement des analyses spécifiques comme les analyses d'ordonnançabilité ou de performance.

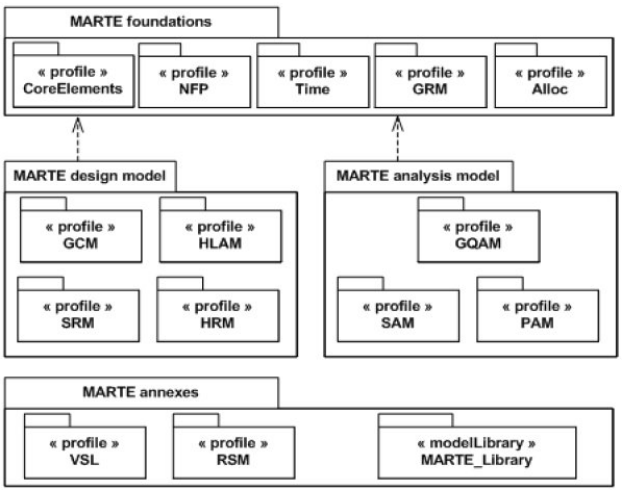


FIGURE 2.10 – Architecture du langage MARTE

En bref, le profil MARTE permet :
– une modélisation du temps plus fine (temps causal, continu ou discret),
– une modélisation des plateformes et leurs services à différents niveaux d'abstraction,
– une modélisation des allocations spatiales ou temporelles de modèles métiers sur des modèles de plateforme.

Enfin, le profil MARTE offre des mécanismes permettant d'annoter les modèles à des fins d'analyse par l'introduction, entre autres, d'un langage de spécification de valeur (VSL – Value Specification Language). Le langage VSL fournit une syntaxe concrète claire permettant de saisir les propriétés non-fonctionnelles des systèmes (poids, utilisation des ressources, consommation d'énergie, etc.) nécessaires aux analyses.

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

2.4 Les défis à relever

Afin de rester compétitives, les entreprises du marché des SoCs ont pour objectifs de réduire le temps de mise sur le marché, des coûts de production et d'exploiter au maximum les apports de la technologie. Aussi, dans le contexte de la loi de Moore, plusieurs défis sont à relever :

- La maîtrise des processus de conception depuis la capture des exigences jusqu'au déploiement du produit, sa maintenance et son retrait,
- La réduction des temps de conception par l'adoption d'approches de plus haut niveau préconisant la séparation des préoccupations,
- La réduction des coûts liés aux outils, aux supports ou à l'application des standards,
- La gestion des risques liés à l'évolution du besoin, des technologies ou des standards.

Nous avons vu que pour traiter ces problèmes, les approches classiques ne peuvent être envisagées. Aussi, il existe plusieurs solutions proposées par la communauté de l'ESL. Bien que ces techniques apportent effectivement un gain de productivité, je montrerais dans la suite de ce document que l'utilisation de modèles permet une meilleure montée en abstraction et une meilleure séparation des préoccupations. En particulier, nous verrons que les modèles apportent plus de flexibilité dans la conception et l'analyse de SoCs.

La maîtrise des processus pourrait être grandement améliorée si leurs objectifs étaient clairement établis et s'il existait des mécanismes permettant d'encapsuler, de partager et d'appliquer de manière systématique les savoir-faire. Grâce aux grandes avancées faites dans le domaine de la modélisation ces dernières années, je pense que les modèles peuvent apporter des réponses aux problèmes de maîtrise et de maturation des processus.

Dans cette thèse, je propose d'utiliser les modèles afin de gérer les activités et les artefacts de l'ESL. Cette approche permet une meilleure maîtrise des processus de codesign et fournit aux développeurs un cadre formel les guidant dans leurs activités de conception et d'analyse. Entre autres, je préconise l'utilisation de modèles et de transformations afin de décharger les développeurs des activités fastidieuses telles que la documentation ou le codage. Par ailleurs, nous verrons que ce travail de formalisation est d'autant plus important quand les langages de modélisation utilisés possèdent parfois un champ d'application très large. Par exemple, le langage MARTE offre de nombreux concepts couvrant la conception et l'analyse des systèmes temps réel embar-

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

qués. En comptabilisant le nombre de concepts dans UML additionnés à ceux de MARTE, on arrive à plus de 400 concepts. Il est alors important de définir une méthodologie de conception qui indique très clairement quels concepts utiliser pour couvrir telle ou telle activité.

Aujourd'hui, avec l'avènement de l'ingénierie dirigée par les modèles, une étape supplémentaire peut être franchie par l'automatisation complète du processus de développement supporté par un seul et même outil. Le processus de développement, à la lumière de l'ingénierie dirigée par les modèles, est alors vu comme une suite de transformation de modèles qu'il est possible d'automatiser. La modélisation du processus permet ainsi ce qu'il était impossible de faire hier : générer des outils taillés sur mesure pour une organisation donnée.

2.5 Conclusion

Dans ce chapitre, j'ai évoqué les principales difficultés rencontrées aujourd'hui dans l'élicitation des processus de codesign. J'ai passé en revue quelques une des solutions proposées par l'ESL afin de gérer les risques liés aux développements de systèmes sur puce. Puis, j'ai fait une présentation rapide de l'ingénierie des modèles et de l'ingénierie des processus pour améliorer la définition des processus. Ces présentations nous permettront de mieux comprendre les apports de l'état de l'art présentés dans le chapitre suivant.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Chapitre 3

État de l'art

Sommaire

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

3.1 Introduction

Ce chapitre présente quelques travaux significatifs basés sur l'ingénierie des modèles pour résoudre les problèmes soulevés dans le chapitre précédent. Ce travail de synthèse me permet de positionner mes contributions dans l'état de l'art. Au préalable, je fais une rapide présentation de l'ingénierie des modèles et de ses principes. Cette présentation permettra de mieux comprendre les apports de l'IDM pour organiser les activités de l'ESL et pour gérer les processus de développement conjoint dans leur ensemble.

3.2 L'ingénierie Dirigée par les Modèles

L'ingénierie dirigée par les modèles est un paradigme basé sur l'utilisation intensive de modèles et promouvant la séparation des préoccupations afin de mieux faire face à la complexité de la conception et favoriser l'analyse des systèmes d'information.

La définition de l'IDM est basée sur trois notions clefs [mF04] : les notions de système, de modèle et de transformation. Il n'existe actuellement pas de consensus sur la définition de ces notions, ni même sur la nature de leur relation. Toutefois, je me propose ici de choisir une définition pour chacune de ces notions dans les paragraphes suivants.

Concernant la notion de système, la définition choisie est celle proposée par l'IEEE (Standard Dictionary of Electrical and Electronic Terms) :

> "A system is a combination of components that act together to perform a function not possible with any of the individual parts"

Cette définition me semble pertinente car elle permet de mieux appréhender la complexité des systèmes par une décomposition structurelle / comportementale du système en sous-systèmes moins complexes : c'est le principe même de l'analyse. Dans cette définition, le comportement global du système est réalisé par les interactions entre les comportements de ses sous-systèmes. Cette définition fait ressortir trois notions orthogonales importantes déjà mentionnée dans le chapitre précédent : la structure, le comportement et la communication.

Concernant la notion de modèle, la définition choisie est celle proposée dans [Jan04] :

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

> "A model is a simplification of another entity, which can be a physical thing or another model. The model contains exactly those characteristics and properties of the modeled entity that are relevant for a given task. A model is minimal with respect to a task if it does not contain any other characteristics than those relevant for a task"

Un modèle décrit un système quel qu'il soit, y compris un autre modèle. Par décrire, il faut entendre abstraire ou simplifier. Il ne s'agit pas de décrire de manière exhaustive un système complexe mais plutôt d'en faire ressortir les caractéristiques intéressantes pour mener des analyses pertinentes dans la résolution d'un problème donné. Il existe différents types de modèles (mathématique, physiques, fonctionnels, etc.) et chaque type de modèle permet de résoudre une famille de problèmes particuliers. Par ailleurs, il est généralement nécessaire d'avoir recours à plusieurs modèles (points de vue) pour permettre la construction correcte d'un système.

Le nombre et le type de modèle nécessaires à la réalisation d'un système, de même que la question de la pertinence d'un modèle sont difficiles à établir. Toutefois, dans [Sel03], des éléments de réponses sont donnés. Selon l'auteur, un modèle ne peut être pertinent que si il possède les caractéristiques suivantes :

- il supporte l'abstraction, dans le sens où il ne met en exergue que les aspects nécessaires et suffisants aux analyses du système étudié,
- il est exprimé dans une forme assez compréhensive pour être comprise de manière intuitive,
- il doit être précis, dans le sens où il représente fidèlement les aspects réel du système étudié et il proscrit les ambigüités,
- il doit être moins coûteux et plus facile à développer que le système qu'il représente.

Dans le cadre de l'ingénierie des modèles, l'expression d'un modèle passe par l'utilisation de langages permettant de capturer les caractéristiques pertinentes du système en étude. Les langages choisis doivent alors réifier les concepts nécessaires à la bonne conduite des activités de conception ou d'analyse des processus de développement.

Par exemple, le langage UML (Unified Modeling Language [OMG05b]) est le langage de modélisation généraliste et extensible proposé par l'OMG. Il représente la synthèse des techniques d'analyse et de conception de systèmes à logiciel prépondérant depuis les années 70. Il intègre des mécanismes apportés

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

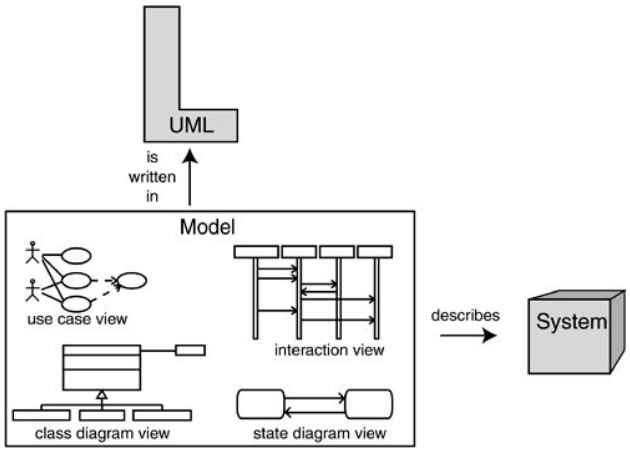


FIGURE 3.1 – Utilisation d'UML pour la modélisation de systèmes

par les approches objets (encapsulation, héritage, polymorphisme), et permet en particulier l'usage des patrons de conception [GHJV95] pour la résolution de problèmes récurrents.

Le langage UML est essentiellement utilisé dans les activités d'analyse permettant de mieux comprendre le besoin fonctionnel (figure 3.1). Il est ensuite complété par différents profils pour la gestion d'aspects spécifiques (exigences, qualité de service, propriétés non-fonctionnelles, plateformes, etc.). Par exemple, nous verrons dans le chapitre 5 de quelle manière le profil MARTE est utilisé dans la méthodologie de conception de SoC pour représenter les plateformes, les propriétés non-fonctionnelles ou les allocations.

Les avantages de l'utilisation de modèles dans les processus de développements logiciels sont nombreux et ont été démontrés à travers plusieurs expérimentations. Ils permettent notamment :

- D'améliorer de manière significative la communication entre les différents métiers impliqués dans les développements,
- De fournir différents points de vue des systèmes étudiés (SUS – System Under Study) suivant les préoccupations adressées,
- De favoriser les processus d'analyses par la représentation des SUS à différents niveaux d'abstraction,
- De traiter les problèmes de portabilité et d'interopérabilité par la séparation des aspects métiers et technologiques,

Enfin, alors que les modèles permettent de saisir les caractéristiques pertinentes des systèmes en étude, les transformations représentent un ensemble

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

de spécifications non-ambigües (règles de transformation) décrivant la manière de créer un modèle (source) à partir d'un autre (cible). Dans [KWB03], les auteurs définissent les transformations, les définitions de transformation et les règles de transformation comme respectivement :

> "A transformation is the automatic generation of a target model from a source model, according to a transformation definition."

> "A transformation definition is a set of transformation rules that together describe how a model in the source language can be transformed into a model in the target language."

> "A transformation rule is a description of how one or more constructs in the source language can be transformed into one or more constructs in the target language."

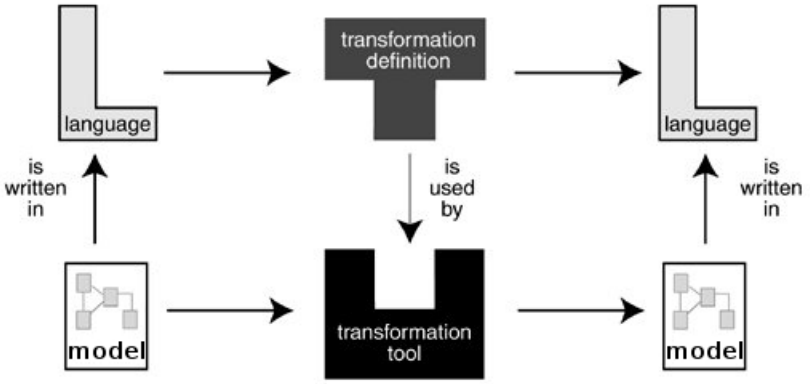


FIGURE 3.2 – Transformation de modèles

La figure 3.2 illustre les définitions données ci-dessus : une transformation de modèles est réalisée par un outil de transformation. Il prend en entrée un modèle exprimé dans un langage source et applique des règles de transformation, exprimées dans un langage dédié, afin de générer un nouveau modèle exprimé dans un langage cible. Une transformation peut être endogène (le langage source et le langage cible sont les mêmes) ou exogène (le langage source et le langage cible sont différents).

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Associé au standard de modélisation UML, on peut citer le langage QVT (Query View Transformation) [OMG05a], également standardisé par l'OMG, comme langage de référence pour exprimer des transformations, créer des requêtes pour interroger les modèles (à la manière de SQL pour les bases de données) ou créer des vues spécifiques. On retiendra principalement des transformations de modèles qu'elles permettent avant tout de :

- Capitaliser les expertises
- Réduire les points de rupture technologiques,
- Automatiser un maximum de tâches routinières (analyses, codage, documentation, etc.).

3.3 IDM et codesign

Dans le domaine des systèmes sur puce, la transformation des exigences en un système faisant les bons compromis entre les performances, les coûts ou les délais est un processus très long. Il est constitué de diverses activités de conception et d'analyse. Ce qui caractérise en premier lieu la conception des systèmes sur puce est l'hétérogénéité : hétérogénéité des sous-systèmes, hétérogénéité des activités, hétérogénéité des métiers impliqués, etc. La complexité engendrée par cette hétérogénéité accroît de manière exponentielle les risques. Aussi, l'introduction de niveaux d'abstraction doit permettre de mieux gérer cette complexité et les risques associés [GVNL94].

Ainsi, comme nous l'avons vu dans le chapitre 2, les apports de l'ESL vont dans ce sens et les évolutions des flots de développement proposées offrent des solutions permettant de :

- Relever le niveau d'abstraction afin d'accélérer les développements,
- Orthogonaliser les préoccupations en séparant les aspects fonctionnels des aspects techniques (concurrence ou communication),
- Construire le système par assemblage d'IPs vérifiés et documentés,
- Réutiliser des plateformes configurables,
- Comprendre et valider les exigences fonctionnelles du système par la construction de spécifications exécutables / vérifiables.

Dans ce chapitre, nous voyons de quelle manière l'ingénierie des modèles peut apporter des réponses aux questions soulevées par l'ESL. J'illustrerai mes propos par la présentation de quelques unes des approches IDM représentatives de la conception de systèmes sur puce. Il y a plusieurs manières de présenter ces apports : par niveau d'abstraction, par activité, par types de

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

modèle, etc. Dans les paragraphes suivants, je fais le choix de les présenter suivant les besoins identifiés en termes de modélisation et d'analyse car les travaux mentionnés dans les paragraphes suivants s'accordent au moins sur la nécessité de capturer (figure 3.3) :

- Un modèle métier comme expression du problème,
- Un modèle de plateforme comme expression du support de la solution,
- Un modèle d'allocation (ou modèle de décision) comme expression des choix d'implantation,
- Un modèle alloué, obtenu après transformation, comme expression de la solution issue des décisions d'implantation.

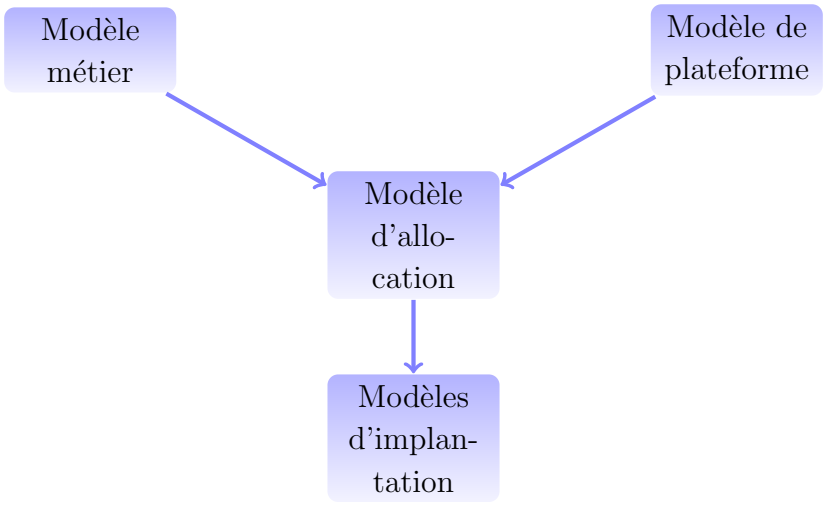


FIGURE 3.3 – Les modèles dans l'IDM

Dans les paragraphes suivant, nous nous intéressons à la modélisation des plateformes d'exécution et des choix d'allocation, la modélisation des systèmes étant très largement adressée par la littérature.

3.3.1 Modélisation des plateformes d'exécution

Un des aspects les plus intéressants de l'approche IDM est la séparation des aspects métiers des aspects technologiques. Cette séparation permet de faire évoluer de manière indépendante, mais pas gratuite, chacun de ces aspects. Dans la terminologie de l'ingénierie des modèles, la notion de technologie est généralement désignée sous le terme de plateforme, qui peut paraître comme un abus de langage.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Par exemple, dans [OMG03], la plateforme est définie comme :

> "A platform is a set of subsystems and technologies that provide a coherent set of functionality through interfaces and specified usage patterns, which any application supported by that platform can use without concern for the details of how the functionality provided by the platform is implemented"

La plateforme désigne donc le support d'exécution des applications métiers. On constate dans la littérature qu'il existe deux familles de langages de modélisation de plateformes. La première a pour principe de réifier les concepts de langages HDL (Hardware Description Language) existants. La seconde approche, plus abstraite, consiste à considérer les éléments de la plateforme d'un point de vue fonctionnel et non fonctionnel. Nous présentons dans les paragraphes suivant les avantages et les inconvénients de ces approches.

Les plateformes comme assemblage d'artefacts HDL

Dans le monde de l'ESL, il existe pléthore de langages de conception matérielle situés à différents niveaux d'abstraction. Parmi les plus outillés et les plus utilisés, on peut citer VHDL, Verilog, SystemC et SystemVerilog. Dans le monde de l'IDM, on constate qu'il existe plusieurs méthodologies se basant sur des langages de modélisation qui réifient les concepts des langages de l'ESL. Cette approche consiste généralement à raccrocher les concepts des langages choisis aux concepts du langage UML par le biais de profils. Par exemple :

- le profil "UML for SoC"[OMG06] standardisé par l'OMG réifie les concepts structurels du langage SystemC,
- le profil "UML for SystemC" [RSRB05] réifie également les concepts structurels et comportementaux du langage SystemC,
- le profil "UML for HDL" [SOD] réifie les concepts du langage VHDL.

La figure 3.4, extraite du profil "UML for SystemC" montre comment les concepts du langage SystemC sont raccrochés aux concepts du langage UML. Cette figure se focalise en particulier sur les concepts structurels du langage. Pour gérer les aspects comportementaux, les auteurs de ce profil étendent le langage d'action de UML pour introduire des actions spécifiques au matériel (par exemple, l'action "wait"), présents dans le langage SystemC.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

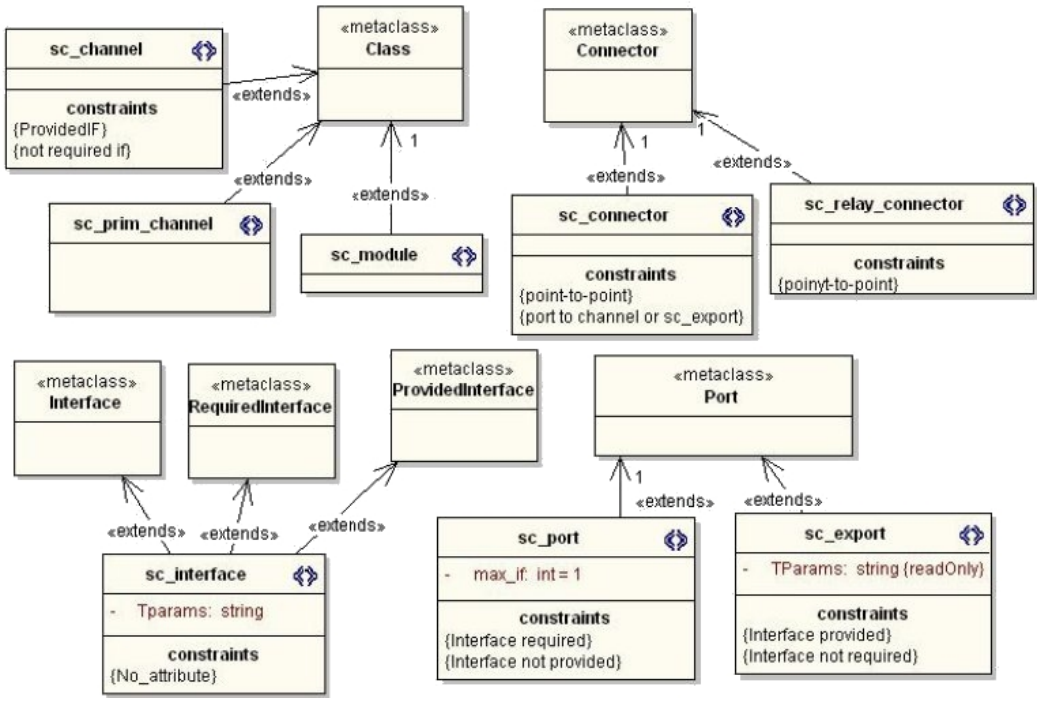


FIGURE 3.4 – Extrait du profil UML for SystemC

Selon les auteurs de [BP], le choix de tels langages se justifie d'une part par le manque de maturité des outils de modélisation pour gérer les activités de codesign, comparé aux outils de l'ESL, et d'autre part, par la difficulté de gérer la synchronisation entre le modèle et le code. Dans ce cadre, le principal avantage de ces approches est de pouvoir réutiliser les outils de simulation, d'analyse ou de synthèse qui ont été développés autours des langages de l'ESL.

Les méthodologies se basant sur ce type d'approche sont nombreuses [RSRB05, WZZ+06]. Ne pouvant toutes les décrire ici, nous choisirons de présenter en particulier la méthodologie UPES [RSRB05] (Unified Process for Embedded Systems), représentative de cet type d'approche. La figure 3.5 donne un aperçu de la méthodologie UPES qui repose essentiellement sur la proposition du profil "UML for SystemC".

La méthodologie UPES est une adaptation du processus unifié (UP – Unified Process) basé sur l'utilisation de modèles et de transformations. Ainsi, le principe de ce type d'approche est d'adapter une approche ESL en remplaçant les artefacts de l'ESL par des modèles. Plus en détail, la partie système

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

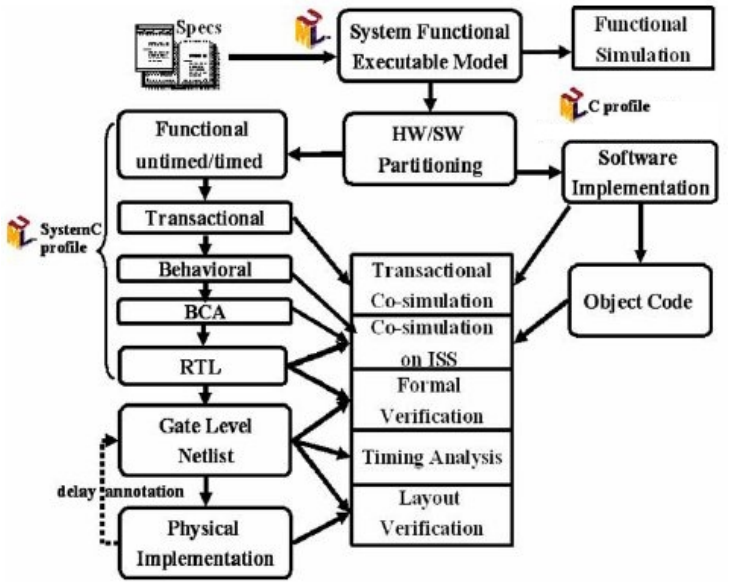


FIGURE 3.5 – Méthodologie UPES

applique l'approche préconisée par les spécifications du processus unifié : elle prend en entrée des exigences textuelles et produit une spécification système exécutable. Ensuite, l'activité de partitionnement détermine quels blocs systèmes seront implantés en matériel ou en logiciel. La spécification de la plateforme d'exécution utilise le profil "UML for SystemC" tandis que celle du logiciel utilise le profil "UML for C" qui réifie les concepts du langage C. L'avantage de cette méthodologie est qu'elle permet en effet de réutiliser un certain nombre d'outils fournis par l'ESL pour valider les choix de conception par la co-simulation. Elle permet en outre de bénéficier des outils de synthèse commerciaux prenant en entrée des architectures exprimées en SystemC. Cependant, ce type d'approche a l'inconvénient de son avantage puisqu'elle reste très dépendante des technologies, ce qui finalement va à l'encontre même des principes de l'IDM.

Les plateformes comme assemblage de blocs fonctionnels

Si les approches se basant sur des langages de modélisation qui réifient des concepts de langages ESL permettent des mieux réutiliser les outils existants, elles ne favorisent pas pour autant la montée en abstraction. Aussi, des approches de plus haut niveau doivent être adoptées pour permettre une meilleure séparation des préoccupations et une meilleure prise en compte

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

de l'hétérogénéité. Dans cette section, nous étudions les approches qui préconisent la représentation des plateformes du point de vue fonctionnel et non-fonctionnel. La figure 3.6 donne un exemple de modélisation de plateforme matériel : les blocs matériels sont abstraits pour n'en faire ressortir que les propriétés fonctionnelles, les autres caractéristiques étant occultées pour des besoins d'analyse. Les langages permettant une telle modélisation sont pléthores et parmi les plus utilisés, on peut citer AADL ou SPT.

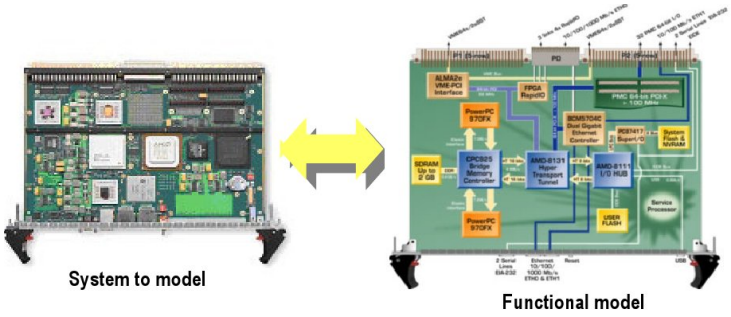


FIGURE 3.6 – Exemple de modèle fonctionnel d'un système complexe

En bref, dans ces approches, les plateformes sont abstraites à un ensemble de ressources interconnectées offrant des services caractérisés par des qualités de service. Finalement, modéliser une plateforme revient à modéliser un ensemble de ressources concurrentes interconnectées, généralement limitées, et leurs services, qui servent de support d'exécution à l'application métier. L'exemple de la figure 3.7, extraite de [CSL+], représente un cas intéressant de modélisation d'une plateforme et de ses couches d'abstraction (HAL – Hardware Abstraction Layer). On peut voir sur cette figure que chaque couche d'abstraction utilise les services fournit par la couche sous-jacente, représenté par la notion d'interface. Les services sont annotés par des caractéristiques relatives aux qualités de service.

Dans [Tah08], l'auteur fait une étude comparative des langages permettant une telle modélisation de plateforme. Aussi, l'auteur de cette étude propose un langage de description de ressources matériel qui fait la synthèse des approches étudiées. Ce langage permet, en plus des propriétés fonctionnelles des plateformes d'exécution matérielles, de décrire les propriétés non fonctionnelles nécessaires à la validation de certaines exigences (poids, encombrement, layout, prix, etc.) en utilisant des mécanismes d'annotation. Cette proposition à laquelle j'ai participé pour la description des plateformes

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

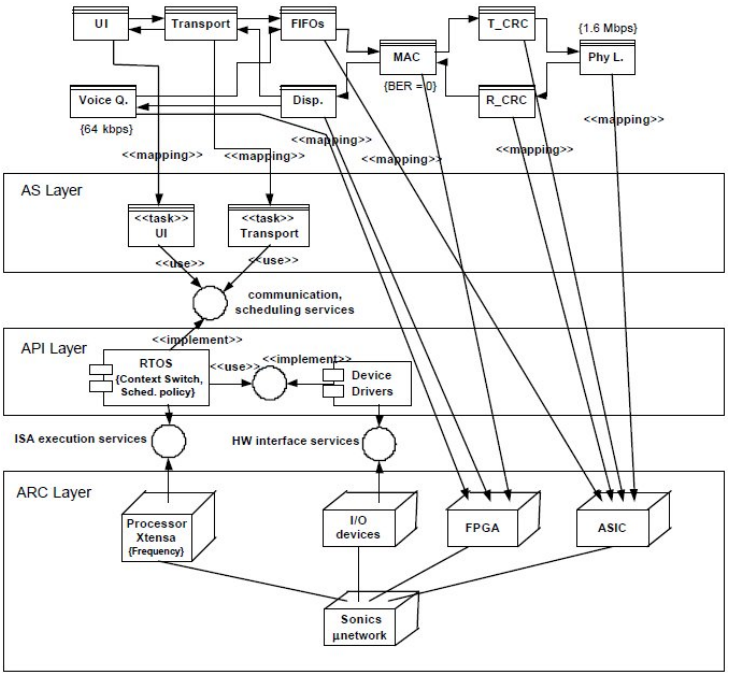


FIGURE 3.7 – Exemple de modélisation de plateforme et d'allocation

reconfigurables, a été depuis intégrée au profil MARTE dans le paquetage HRM (Hardware Resource Modeling).

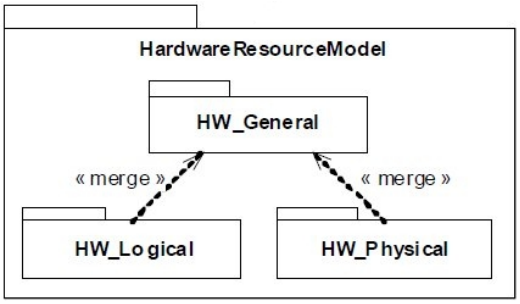


FIGURE 3.8 – Structuration du paquetage HRM de MARTE

La figure 3.8 montre la structure du paquetage HRM contenant l'ensemble des concepts de MARTE pour la description de plateformes matérielles. Cette structuration reflète le besoin de séparer les préoccupations en fournissant aux utilisateurs la possibilité de représenter le matériel selon les points de vue fonctionnel ou physique. Dans cette proposition, le concept de ressource matérielle est un concept abstrait représentant une entité matériel comme

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

une unité d'exécution dont les services sont caractérisés par un ensemble de qualité de service. Le paquetage "Hw_Logical" a pour l'objectif de fournir une taxonomie pertinente des ressources suivant leurs caractéristiques fonctionnelles. En résumé, les ressources matérielles peuvent être classées en :

- Ressources de calcul, comprenant l'ensemble des ressources fournissant des services d'exécution comme les processeurs, les ASIC (Application Specific Integrated Circuit) ou les PLD (Programmable Logic Device),
- Ressources de mémorisation, comprenant l'ensemble des ressources fournissant des services de mémorisation (persistantes ou non) comme les mémoires RAM, les caches ou les disques durs,
- Ressources de communication, comprenant l'ensemble des ressources fournissant des services de transfert de donnée comme les bus ou les passerelles,
- Ressources de temps, comprenant l'ensemble des ressources fournissant des services d'accès au temps comme les horloges,
- Ressources auxiliaires, comprenant l'ensemble des ressources fournissant des services d'accès aux éléments extérieurs du système comme les ressources d'entrée / sortie ou les senseurs.

En plus de représenter les ressources matérielles et leurs services, le paquetage "HW_Logical" fournit des mécanismes d'accès aux ressources partagées comme par exemple les arbitres pour les bus de communication. Enfin, le paquetage "HW_Physical" fournit une vue complémentaire plus axés sur les caractéristiques physiques telles que le poids, l'encombrement, le pin-out ou même le prix, nécessaires au traçage des exigences relatives à ces caractéristiques..

Dans la modélisation des plateformes d'exécution, il est parfois difficile de caractériser de manière formelle et précise toutes les caractéristiques des plateformes. En effet, certaines caractéristiques ne peuvent être déterminées statiquement. Par exemple, les communications à travers des bus ou même par une interface air (wifi, wiMax) peuvent corrompre l'intégrité des données. Il est alors nécessaire d'intégrer ces caractéristiques non prédictibles à travers des modèles statistiques et probabilistes [Kou96]. Ce sont ces questions que les mécanismes d'annotation fourni par le paquetage "NFP" (Non-Functional Properties) traitent.

Les méthodologies de codesign basées sur ce type d'approche sont nombreuses. Nous présentons ici la méthodologie "Gaspard" [PAM+08], schématisée par la figure 3.9, représentative d'une méthodologie IDM de codesign.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

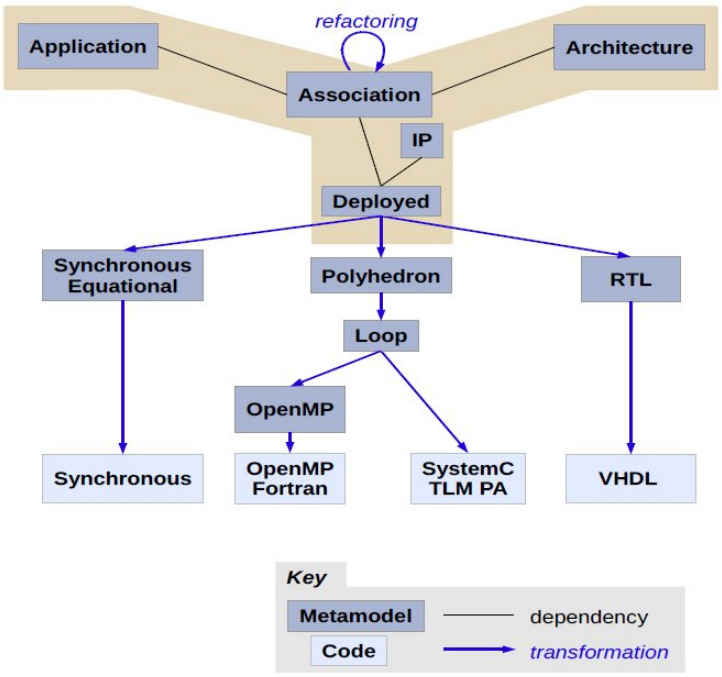


FIGURE 3.9 – Aperçu de la méthodologie Gaspard

Cette méthodologie, basée sur l'utilisation du profil MARTE, se focalise essentiellement sur des applications spécifiques nécessitant des architectures de type MPSoC (Multi-Processors-on-Chip) contenant des grilles de processeurs. De fait, les applications visées sont des applications axées sur un parallélisme de données et de traitements important. Aussi, pour modéliser ces architectures systèmes et matérielles particulières, la méthodologie "Gaspard" utilise les concepts fournit par le paquetage "RSM" (Repetitive Structure Modeling) de MARTE permettant de modéliser des structures dites "répétitives". L'outillage de cette méthodologie implante un certain nombre de transformations de modèles permettant de générer du code pour des outils d'analyses dédiés aux types d'application visés ainsi que pour des outils de simulation et de synthèse de l'ESL. La section 3.3.3 donne un exemple de modélisation de structures répétitives et d'allocation de cette méthodologies. Les travaux réalisés dans le cadre de la méthodologie GASPARD démontrent la pertinence de l'approche IDM pour gérer les activités de codesign. En particulier, elle démontre la nécessité de relever le niveau d'abstraction et de bien séparer les préoccupations.

Parmi les travaux important qui tentent de connecter l'ingénierie des mo-

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

dèles à l'ESL, on peut citer la thèse [Rev08] qui propose un profil UML nommé "UML for ESL" dont les objectifs sont, entre autres, de :

- fournir les concepts permettant de décrire des composants matériels en portant une attention particulière à la séparation interface / implantation,
- fournir une interopérabilité avec les formalismes de référence de l'ESL, et en particulier IP-XACT [spi],
- automatiser grâce à l'IDM le passage de modèles de spécification vers des modèles d'implantation exploitant des profils particuliers à un langage ou à un niveau d'abstraction donné.

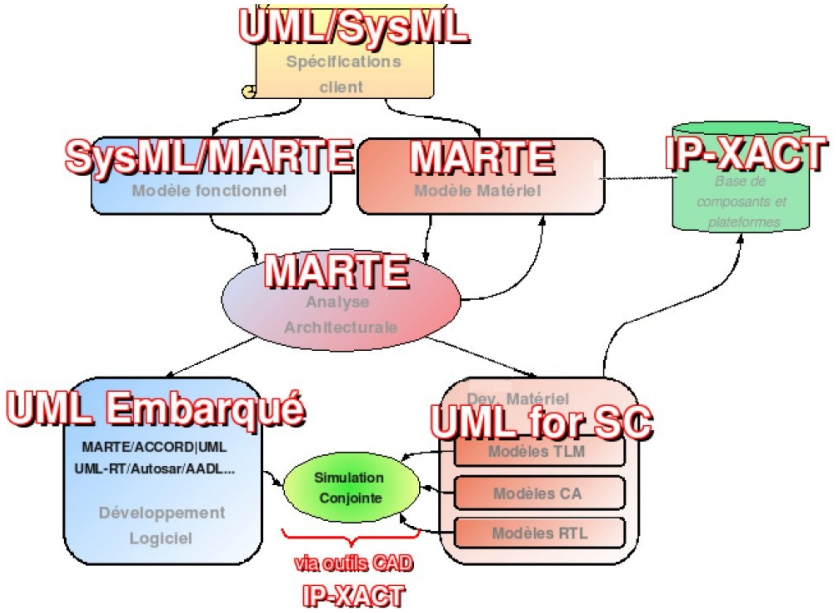


FIGURE 3.10 – Utilisation du profil UML for ESL

Le figure 3.10 résume la proposition d'un flot ESL basé sur l'utilisation de ce profil. La spécification des exigences précède le partitionnement matériel / logiciel. Cette activité doit être validée par des modèles d'analyse MARTE. Ensuite, des modèles de plus bas niveaux d'abstraction traitent l'intégration de composants logiciels / matériels dans les outils de co-simulation. À cet effet, le profil "UML for ESL" permet de décrire les composants matériels aux différents niveaux d'abstraction identifiés par l'ESL (TLM, CA, RTL), de les capitaliser et de les réutiliser dans d'autres conceptions.

Pour compléter cette étude, nous donnons dans les lignes suivantes quelques références à d'autres travaux relatifs sans rentrer dans le détail de leur des-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

cription. Le lecteur pourra trouver des études comparatives détaillées de méthodologies IDM pour l'ESL dans [Rev08]. Dans les travaux présentés dans [CSL+], les auteurs présentent la méthodologie "Metropolis"qui adapte l'approche "Platform Based Design" à l'approche IDM. Ces travaux mettent en exergue la nécessaire séparation des préoccupations pour mieux gérer l'hétérogénéité des systèmes à concevoir. Dans [EG03], les auteurs proposent la méthodologie HASoC (Hardware and Software Object on Chip) qui utilisent UML pour fournir des modèles exécutables du système à différents niveaux d'abstraction : le niveau "uncommited model" a pour objectif de fournir une spécification purement fonctionnelle alors que le niveau "commited model" s'attache davantage aux détails d'implantation logiciels / matériels. Pour finir, la méthodologie Harmony/ESW (Embedded System Workflow) présentée dans [Dou09] est une adaptation du RUP (Rational Unified Process) dédié au développement de systèmes embarqués temps réel. Cette méthodologie est basée sur l'utilisation d'UML et des profils SysML [OMG08] et SPT [OMG05c].

Nous avons dans ce chapitre essentiellement parlé de plateformes matérielles. Mais les plateformes regroupent également les couches d'abstractions (HAL – Hardware Abstraction Layer) permettant de faciliter la conception en cachant la complexité des couches sous-jacentes. On peut citer par exemple les systèmes d'exploitation (VxWorks, Linux, Windows), les middlewares (CORBA CCM, EJB, .NET), ou les machines virtuelles (JVM). Les plateformes logicielles sortent du cadre de cette thèse. Cependant, le lecteur intéressé par le sujet pourra trouver un état de l'art du sujet dans [Tho08].

3.3.2 Prise en compte de l'hétérogénéité des plateformes

Parmi les principaux problèmes adressés par la communauté ESL, les questions de la représentation des modèles de calcul reste un des challenges les plus importants de cette communauté. En effet, comme nous l'avons mentionné précédemment, la nécessité d'orthogonaliser les divers aspects relatifs aux modèles de calcul a été souligné par plusieurs travaux [CSL+, Jan04] et il semble que l'approche IDM soit un bon candidat pour traiter ces questions [SVSS+09].

Parmi les travaux les plus significatifs de l'ESL, on peut citer les travaux de l'université de Berkeley autour de l'outil Ptolemy [BHLM02] et ceux de l'université de Cantabria [HSV04] visant à utiliser le moteur à événements

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

discrets de SystemC pour simuler des systèmes hétérogènes.

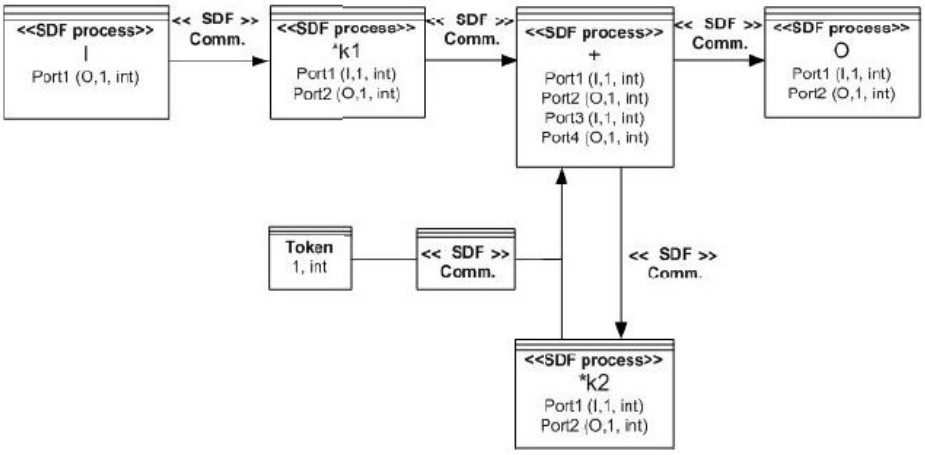


FIGURE 3.11 – Capture d'un MoC hétérogène avec "UML Platform Profile"

Malheureusement, la question de l'hétérogénéité des plateformes est encore peu adressée par l'ingénierie des modèles. Cependant, dans [CSL+], les auteurs proposent une adaptation IDM de la méthodologie "Platform Based Design", basée sur l'utilisation de profils UML. Après avoir constaté les manques d'UML pour représenter les choix d'implantation en termes de modèles de calcul, les auteurs proposent un profil spécifique nommé "UML Platform Profile". Ce profil propose un ensemble de stéréotypes permettant de modéliser des plateformes d'exécution à un haut niveau d'abstraction. La notion de plateforme est réifié par le stéréotype "netlist" qui représente un ensemble de ressources ("Resource") interconnectés par des médias de communication ("medium") à travers leurs ports ("Port"). Les ressources contiennent des processus concurrents ("Process") chargés d'effectuer des calculs. Enfin, les communications entre processus peuvent être spécifiées par l'application du stéréotype "communicate" sur leurs liens. En particulier, ce stéréotype est raffiné de manière à intégrer les mécanismes les plus usités dans la conception de SoC. Ainsi, on retrouve dans ce profil les stéréotypes "sdf" (Synchronous Data Flow), "kpn" (Kahn Process Network), "csp" (Communicating Sequential Process), etc. Par exemple, la figure 3.11 illustre l'utilisation de ce profil pour la modélisation d'un filtre FIR pour le traitement du signal.

Dans [Har08], l'auteur propose une approche par composition de modèles multi-paradigme nommée ModHel'X. Cette approche a pour but de :

– Spécifier la sémantique d'un langage de modélisation de manière exé-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

cutable à travers la spécialisation opérationnelle d'une sémantique abstraite pour les modèles de calcul,
- Spécifier explicitement les mécanismes de composition à utiliser entre des modèles hétérogènes via une structure de modélisation appelée bloc d'interface,
- Simuler le comportement global de modèles hétérogènes par un algorithme générique d'exécution.

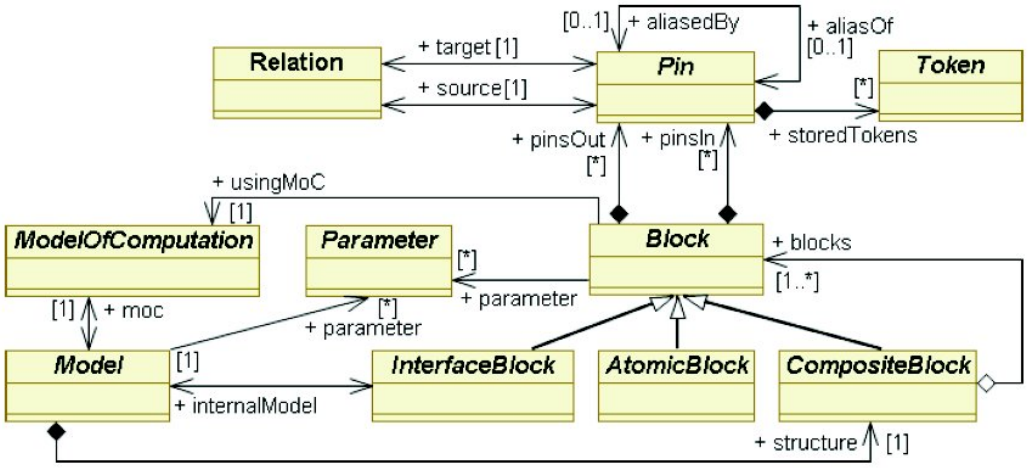


FIGURE 3.12 – Syntaxe abstraite de ModHel'X

La figure 3.12 représente la syntaxe abstraite du langage ModHel'X. L'utilisation de ce méta-modèle permet de mieux gérer l'hétérogénéité grâce l'emploi de représentations structurelles homogènes. Nous retiendrons de ce méta-modèle deux notions importantes pour le support de l'hétérogénéité :
- le bloc composite pour la représentation hiérarchique d'un système hétérogène,
- le bloc d'interface pour le support de l'interfaçage entre MoCs hétérogènes.

3.3.3 Modélisation des allocations et des analyses

Lorsque le modèle d'application a été défini et qu'une technologie d'implantation a été choisie, il faut décider des choix d'implantation qui devront être validés par différents types d'analyse. Ces choix consistent essentiellement à déterminer de quelle manière les fonctionnalités du système seront exécutées sur la plateforme. Les analyses qui découlent de ces choix doivent

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

permettre alors de juger de la bonne utilisation des ressources de la plateforme à travers le temps et l'espace. Par exemple, si on considère un ensemble de fonctions s'échangeant des données, les choix d'implantation pour une même plateforme peuvent être nombreux : doit on exécuter les fonctions de manière séquentielle ou en "pipeline" ? Sur une même unité de calcul ? Communiquent-elles par des variables partagées en mémoire ou utilisent-elles des canaux de communication dédiés ? etc. L'analyse de ces choix nécessite au préalable leur capture à travers un modèle d'allocation.

En cherchant une définition des modèles d'allocation, on se rend compte qu'il est difficile d'en trouver une qui satisfasse l'ensemble des communautés. Dans cette thèse, nous retiendrons la définition donnée dans [DEAB08] :

> "An Allocation can represent either a spatial placement or a temporal placement. Spatial placement is the allocation of computation to processing resources, of data to memory at specific range of addresses, and of dependencies between application components to communication resources. Temporal placement is the scheduling of a set of elements spatially allocated to the same platform resource."

En bref, une allocation définit :

- le placement des fonctions sur les ressources de calcul de la plateforme (CPU, FPGA),
- le placement des données sur les mémoires (SRAM, Flash),
- le placement des communications.

Dans le cas où plusieurs blocs applicatifs sont alloués sur la même ressource de calcul, il est nécessaire de préciser la temporalité des accès aux services de la ressource considérée. Par exemple, dans le cas des ressources reprogrammables matériellement, l'allocation définit alors la configuration de la ressource allouée à un instant t. La manière de représenter les allocations et les informations qui y sont associées dépend beaucoup de la manière de représenter les applications et les plateformes d'exécution ciblées.

La figure 3.13 extraite de [DEAB08] montre un exemple d'allocation d'une application de décodage de flux H263 sur une plateformes MPSoC. Cette figure illustre un exemple d'allocation d'une application constituée de tâches répétitives sur une plateforme constituée de ressources de calcul répétitives (grille de processeurs). On comprendra aisément qu'une telle concision dans la modélisation de l'application et de la plateforme implique l'usage de notations

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

adaptées pour représenter l'allocation. Par exemple, cette modélisation fait l'emploi du stéréotype "distribute" et associe un ensemble de tags précisant les contraintes d'allocation.

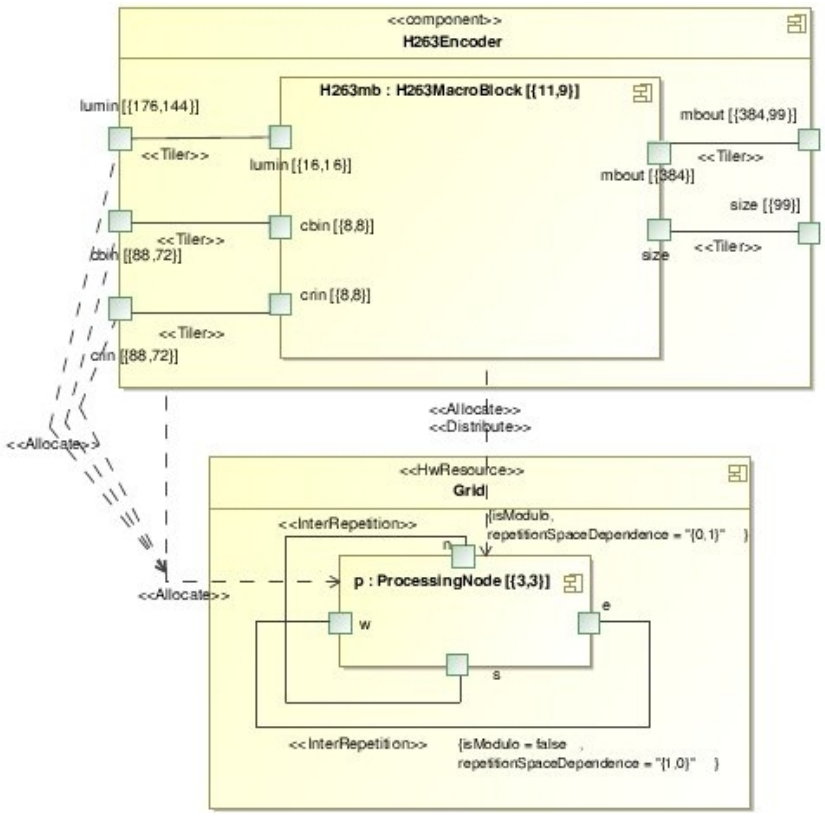


FIGURE 3.13 – Exemple d'allocation sur un MPSoC

Les choix d'implantation capturés par le modèle d'allocation sont avant tout guidés par les exigences non-fonctionnelles comme les exigences de performance ou de consommation. La validation de ces choix ne peut généralement pas être réalisée de manière formelle à cause des caractéristiques non-déterministes des ressources de la plateforme. Elle nécessite de passer par l'élaboration de scénarios d'analyses qui se focalisent sur des questions relatives aux propriétés non-fonctionnelles du système :

- Est-ce que les performances exigées seront toujours tenues ?
- De quelle manière évolue la consommation du système si les charges augmentent ?
- Comment mesurer la fiabilité du système si celui-ci entre dans un environnement hostile ?

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Pour répondre à ces questions, il est nécessaire de créer des contextes de simulation qui mettent en exergues toutes ces caractéristiques (scénarios d'analyse de performance, d'ordonnancement, de WCET (Worst Case Execution Time), etc.).

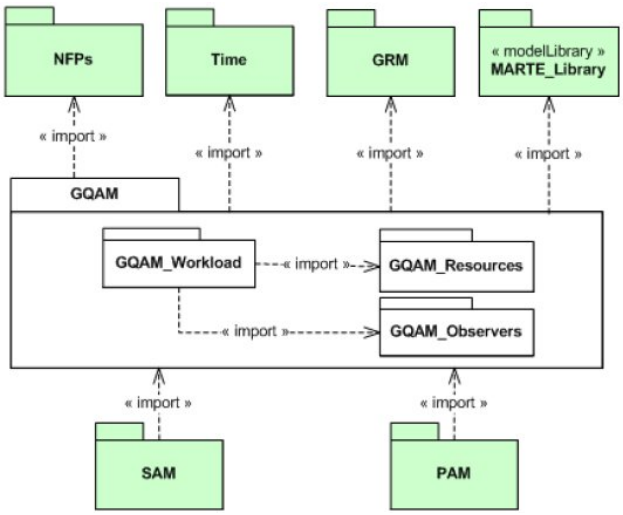


FIGURE 3.14 – Modélisation des analyses dans MARTE

À cet effet, le profil MARTE fournit dans le paquetage "GQAM" (Generic Quantitative Analysis Modeling – figure 3.14) les concepts nécessaires à l'élaboration de scénarios d'analyses exploitant un ensemble d'annotations dédiées. Ces annotations consistent à établir des relations entre des mesures statistiques portant sur diverses propriétés non fonctionnelles en entrée (nombre de requêtes, nombre d'exécution requis) et en sortie (latences, nombre de contraintes temporelles violées, quantité de ressources utilisées) du système.

En outre, le paquetage "GQAM" fournit des concepts génériques extensibles pour traiter tout type d'analyse. Il fournit en particulier deux extensions :

- Une pour les analyses d'ordonnancement (SAM – Schedulability Analysis Modeling),
- Une autre pour les analyses de performance (PAM – Performance Analysis Modeling).

Aujourd'hui, les technologies IDM adressant ces questions ne sont pas encore assez avancées. Aussi, il existe un certain nombre de passerelles vers des outils de validation ou d'optimisation de placement qui ont fait leurs preuves.

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Par exemple, le projet OpenEmbeDD [1] propose des transformations de modèles visant l'outil Syndex qui implante la méthodologie AAA (Algorithm Architecture Adequation) basée sur la théorie des graphes.

Les modèles d'allocation ne recherchent pas nécessairement une solution globale optimale au problème. L'idée dans un premier temps serait plutôt de trouver une solution satisfaisante dans un délai raisonnable pour répondre aux contraintes du marché.

3.4 IDM et processus de développement

La maîtrise des processus est le passage obligatoire pour qui veut maîtriser les délais et les budgets, ou augmenter les gains de productivité. Celle-ci passe nécessairement par l'application d'une méthodologie qui définit clairement l'ensemble des activités, des responsabilités ou des artéfacts produits ou consommés. En outre, la définition d'une méthodologie de développement doit tenir compte des contextes techniques, juridiques ou environnementaux des organisations en charge de la maitrise d'ouvrage dans l'élicitation des processus [Mek08].

Le succès d'un projet dépend généralement de la bonne exécution de plusieurs processus transverses tels que le processus de gestion des projets ou le processus de gestion des relations avec les clients. En particulier, dans les processus techniques, la construction d'un système conforme aux besoins des clients nécessitent la mise en œuvre d'un ensemble de méthodes de conception et d'analyse prenant en compte l'évolution des technologies ou les contraintes de temps et de coûts imposées par les lois du marché.

Dans la section suivante, je fournis quelques définitions de la littérature permettant de mieux comprendre les apports de l'ingénierie des processus. Nous voyons ensuite de quelle manière l'ingénierie des modèles permet d'améliorer la représentation et l'analyse des processus de développement.

3.4.1 Quelques définitions

Dans le domaine du développement logiciel, la nécessité d'établir un consensus sur l'univers du discours a été soulignée dans [Lon93]. Dans cet article, l'auteur propose un ensemble de définitions à partir de l'état de l'art du

1. http ://openembedd.org

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

moment. Par exemple, on constate dans la littérature que l'usage du terme "méthodologie" constitue généralement un abus de langage pour désigner une méthode décrivant un ensemble d'actions et de moyens mis en œuvre pour atteindre un but [Yin03]. Dans le cadre de cette thèse, je choisis la définition donnée dans [Cal90] et qui me semble pertinente :

> "Une méthodologie est un ensemble structuré et cohérent de méthodes, guides, outils permettant de déduire la manière de résoudre un problème."

Une méthodologie explicite le cheminement permettant de résoudre un problème dans un domaine particulier. Elle précise l'utilisation de compétences, de formalismes ou d'outils nécessaires à la résolution du problème posé.

Le terme processus est plus difficile à définir car il est utilisé dans un très grand nombre de domaines (chimie, biologie, physique, etc.). Cependant, je choisirai la définition générique trouvé dans le dictionnaire en ligne `http://dictionary.reference.com` :

> "A process is a systematic series of actions directed to some end."

Cette définition, bien qu'étant en effet très générale, me paraît très pertinente dans la mesure où elle fait ressortir l'aspect tangible, mesurable des processus.

Enfin, associées aux méthodologies et aux processus, il existe bien d'autres notions dont le nombre et la nature diffèrent suivant le domaine ou la communauté adressée. Par exemple, dans les domaines du traitement de l'information, on retrouve des notions récurrentes comme les notions de responsabilité, de produit ou d'activité. Ne pouvant énumérer ici toutes ces notions, je renvoie le lecteur désireux d'approfondir le sujet à la lecture de [Lon93].

3.4.2 Modélisation des processus

Si l'on s'en tient aux ingénieries système et logicielle, un processus de développement permet de gérer le cycle de développement d'un système. Dans la pratique, celui-ci est généralement décomposé en phases de conception, d'implantation, de validation, de mise en production, etc. Chaque phase peut ensuite être décomposée en activités, puis les activités en sous-activités, et ainsi de suite. Un processus de développement n'est pas nécessairement linéaire. En effet, certaines activités du processus peuvent être exécutées de manière concurrente et un des objectifs de l'ingénierie des processus est de

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

paralléliser au mieux les tâches du processus afin de réduire les délais de développements, et par voie de conséquence les coûts. La réalisation de tels objectifs peut faire apparaître des risques dont il faut assurer la maîtrise.

L'utilisation de modèles traditionnels ne peut pas être envisagée pour gérer la maîtrise des processus. En effet, les modèles traditionnels posent un sérieux frein à l'élicitation et la maturation continue des processus de développement car ils ne permettent au mieux que la description informelle des processus. Les principaux reproches qui leurs sont faits sont leur nature informelle et leur grosse granularité. Aujourd'hui, les processus ont atteint un tel niveau de complexité qu'il est nécessaire de proposer des approches de plus haut niveau facilitant la définition, l'élicitation et l'amélioration continue des processus. Des telles approches doivent permettre l'expression de modèles plus formels pour une meilleure capture et une meilleure évaluation des processus, et qui satisfassent les propriétés suivantes [B.C88] :

- non-ambigüité : la définition du cycle de vie ne doit permettre qu'une seule interprétation possible,
- complétude : toutes les informations pertinentes doivent figurer dans le modèle,
- vérifiabilité : toute information peut être vérifiée par une personne ou par un outil,
- cohérence : absence de conflit entre informations, cohérence et conformité des données,
- modificabilité : le cycle de vie doit pouvoir être modifiable de manière complète et cohérente,
- traçabilité : possibilité de déterminer l'origine de toutes les composantes du système,
- présentabilité : toute information doit pouvoir être retrouvée et affichée correctement.

La satisfaction de ces propriétés est au cœur des préoccupation de l'ingénierie des processus. Dans les faits, les apports de l'ingénierie des processus ont surtout été démontrés dans l'ingénierie logicielle. L'ingénierie des processus s'appuie sur des représentations formelles des processus faisant ressortir les caractéristiques pertinentes des processus pouvant fournir des moyens d'analyse efficaces [Ost09]. Ces représentations doivent alors utiliser des notations adaptées possédant une sémantique clairement définie [ZKJ09, GL09]. Il existe dans la littérature un grand nombre de langages permettant de représenter les processus. La définition de ces langages peut être basée sur

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

des ontologies, des schémas XML, des grammaires ou des méta-modèles. Des comparaisons pertinentes de ces différents formalismes sont fournies en détail dans [Bre02, Ben07].

Dans cette thèse, nous en tiendrons à la présentation de travaux relatifs au langage SPEM introduit dans le chapitre 2. Le méta-modèle SPEM a fait l'objet de plusieurs expérimentations mettant en avant ses avantages par rapports à d'autres formalismes mais également et surtout ses manques. Dans les sections suivantes, nous évoquons quelques travaux qui tentent de résoudre des problèmes important de l'ingénierie des processus à travers des propositions d'extension de ce langage. Les sections suivantes traitent du problème de l'élicitation des processus de développement à travers l'exécution de modèles puis de l'encapsulation des expertises dans les composants de processus.

3.4.3 Élicitation des processus à travers l'exécution de modèles

Comme le remarquent les auteurs de [ben09], l'utilisation de SPEM pour modéliser des processus de développement système et logiciel apporte incontestablement un plus en terme de formalisation. Seulement, si les modèles capturés restent dans le domaine contemplatif et à des fins documentaires, ce plus ne servirait finalement pas à grand chose compte-tenu de l'effort nécessaire à sa capture. Aussi est-il nécessaire d'aller plus loin dans l'orchestration des processus en proposant une élicitation des processus par une exécution totale ou au moins partielle de ces modèles.

Parmi les améliorations possibles du langage SPEM, une des propriétés les plus attendues était l'exécutabilité des modèles de processus. Elle faisait par ailleurs partie des exigences émises par l'OMG dans sa RFP pour la version 2.0 du langage. Dans [BCCG07], les auteurs soulignent le fait que cette exigence n'a finalement pas été couverte par la proposition malgré la forte pression des communautés en faveur de l'automatisation et la validation des processus.

Cette propriété est de prémière importance pour qui souhaite avoir un contrôle plus fin du cycle de vie des processus en adéquation avec la définition du modèle. Pour répondre à ce problème, les auteurs proposent une extension de SPEM, nommée xSPEM (eXécutable SPEM) adressant la spécification comportementale des processus cadrée par un ensemble de contraintes

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

temporelles ou d'ordonnancement. La principale contribution de xSPEM est l'introduction d'une sémantique opérationnelle par l'intégration de concepts permettant la communication entre activités du processus à base d'événements.

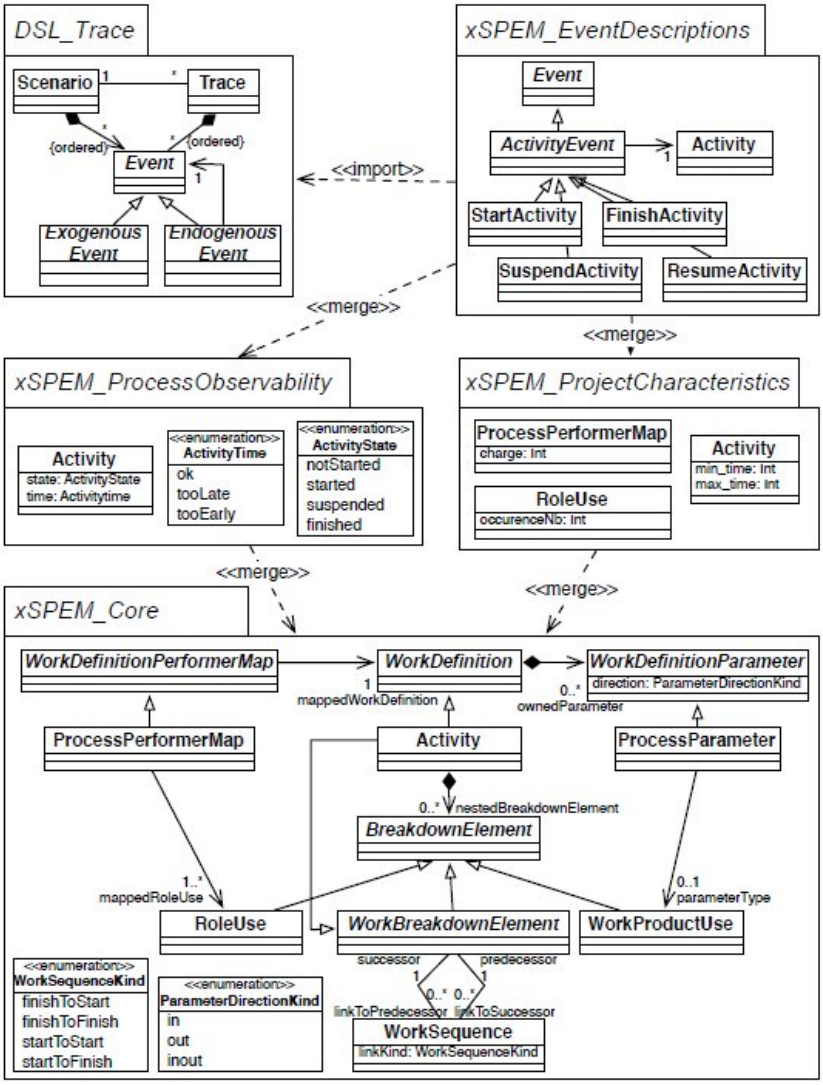


FIGURE 3.15 – Extension xSPEM

La figure 3.15 illustre la syntaxe abstraite du langage xSPEM. Dans cette approche, la simulation est basée sur l'observation des événements produisant des traces d'exécution. La sémantique sous-jacente est une sémantique translationnelle : la simulation est rendue possible par des transformations

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

de modèles vers des réseaux de Petri.

Dans les travaux présentés dans [ben09], les auteurs proposent d'utiliser le langage KerMeta afin d'attacher une sémantique opérationnel au langage UML4SPM [Ben07] afin d'exécuter les modèles de processus. Le principe de cette approche consiste également à exploiter des traces d'exécution afin d'améliorer la définition des processus de développement.

Malheureusement, dans ces deux approches, une telle exploitation ne peut être réalisée que manuellement car SPEM n'offre pas de concepts permettant la rétro-annotation et une gestion fine des données du processus. À cet effet, les auteurs de [LDM06] proposent une extension du langage SPEM permettant une meilleure gestion du cycle de vie des activités et des artefacts du processus par des mécanismes de rétro-annotation.

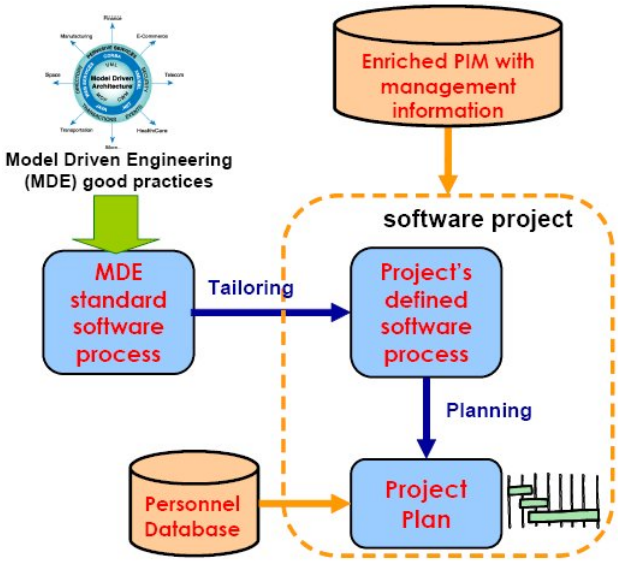


FIGURE 3.16 – Gestion des données et rétro-annotations

La figure 3.16 résume l'idée de cette approche qui propose un mécanisme d'annotation des produits des processus avec des informations de gestion. Les annotations correspondent en fait aux informations recueillies lors de l'élicitation des processus et sont automatiquement réinjectées dans le modèle de processus. Ce mécanisme semi-automatique permet :

- gérer finement le cycle de vie des activités et des artefacts du processus,
- corriger les actions au fur et à mesure des développements,
- améliorer la qualité des processus.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

3.4.4 Les composants de processus

La capitalisation et la réutilisation des savoir-faire dans l'ingénierie des processus comptent parmi les plus gros enjeux de l'ingénierie des processus. Dans ce cadre, plusieurs travaux ont montré la pertinence d'adapter les approches "composants logiciels" aux modèles de processus. Les arguments avancés sont les avantages apportés en termes d'analyse et de réutilisation [Mad91] mais aussi le possible franchissement d'un nouveau pas dans la course à la productivité [CCTDDTB01]. Dans [CR06, Kam00] les auteurs montrent que la gestion des processus de développement à travers des composants de processus permet de mieux traiter la question de l'amélioration continue des processus. En outre, dans [Mek08], l'auteur ajoute que l'adaptation des processus à différentes lignes de produits n'en serait que facilitée.

On constate que dans le langage SPEM, la définition des composants de processus est plutôt sommaire. Aussi, plusieurs définitions de composants de processus ont été proposées par la littérature. Nous voyons dans les paragraphes suivants les définitions qui m'ont semblé les plus pertinentes. Cette étude permet de mieux comprendre nos apports dans la modélisation des composants de processus.

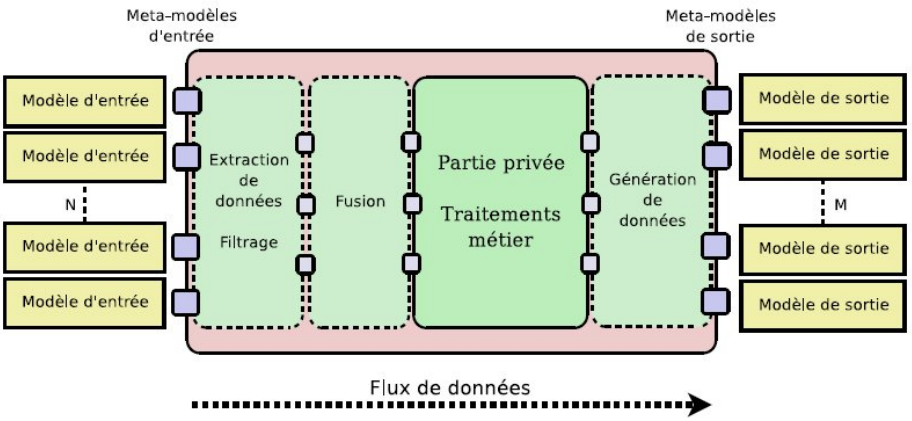


FIGURE 3.17 – Composants de processus

Dans [Mek08], le composant de processus est défini comme une unité d'encapsulation caractérisée par sa partie publique et sa partie privée. Alors que la partie publique du composant expose l'ensemble des services qu'il offre au monde extérieur, sa partie privée encapsule les artefacts nécessaires à la réalisation de ces services (figure 3.17).

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Dans cette approche, les services fournis ou requis par un composant de modélisation sont en relation directe avec le type de modèles que le composant manipule. Cela induit pour ce composant la connaissance liée aux méta-modèles correspondants et la mise en œuvre de services de base dans l'outillage. Ceux-ci sont liés à la gestion des modèles afin d'assurer la cohérence des processus. Par modèle, il faut entendre les modèles et leur méta-modèle, car à l'image des modèles que les processus manipulent, les méta-modèles sont également des produits ayant leur propre cycle de vie.

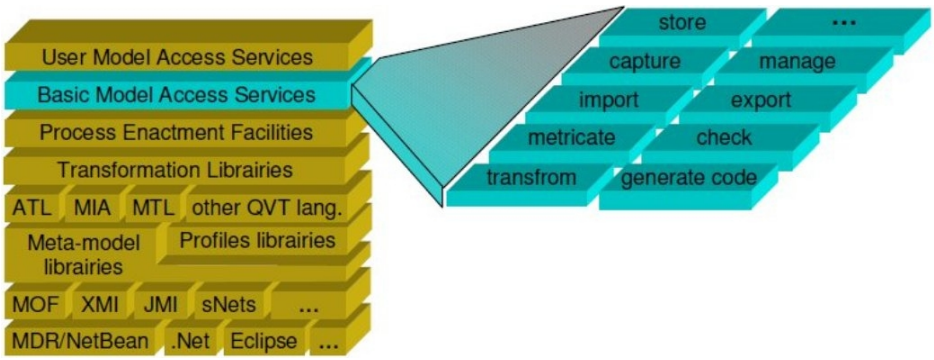


FIGURE 3.18 – Classification des services

Ainsi, parmi les caractéristiques importantes des composants de processus se trouve la notion de service, associée à la notion de (méta-)modèle. Dans [BG, BGML03], les auteurs fournissent une définition des composants de processus similaire. En outre, ils fournissent une taxonomie fine des services et des modèles associés, représentée par la figure 3.18. Cette définition s'inscrit dans le cadre d'une proposition de composant de processus nommé "MDA Tool Component" (figure 3.19) dont l'objectif est de gérer des chaînes d'outils. Le contexte de ces travaux est l'approche MDA préconisée par l'OMG. Ainsi, cette approche présume de l'alignement des représentations sur le MOF afin de faciliter la mise en œuvre des composants de processus.

Cet alignement se justifie par l'utilisation d'un bus dédié permettant les interactions et l'interopérabilité des composants puisque le cadre MDA présume de la conformité des méta-modèles au méta-modèle MOF. Basé sur cette hypothèse, les auteurs suggèrent un cadre d'exécution, appelé MIF (Model Interoperability Framework), permettant la mise en œuvre de chaines d'outils.

En relation avec ces travaux, la vérification des composants de proces-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

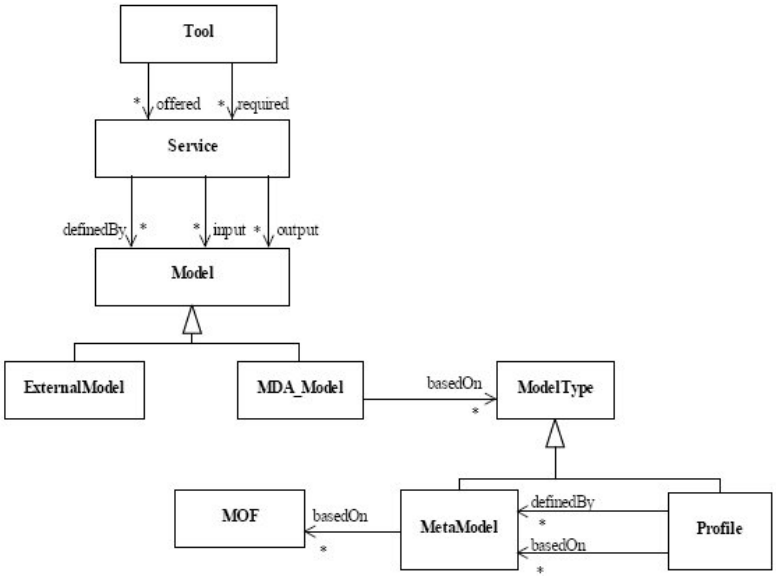


FIGURE 3.19 – Métamodèle "MDA Tool Component"

sus est adressée dans [MBLT06] où les auteurs établissent que l'aspect reproductible des transformations de modèles appliqués aux différentes activités du processus doit passer par des phases de validation, tout comme les composants logiciels. Dans cette proposition, les composants sont désignés comme des composants MDA réutilisables conçus par une approche de type "Design-by-Contract". Ainsi, chaque spécification de composant est traduite en contrat exécutable vérifié et validé. Dans cette approche, un composant MDA possède 3 facettes :

- La spécification, comprenant l'interface du composant, les invariants, les pre/post conditions et la documentation,
- l'implémentation,
- les vecteurs de tests pour la validation.

À travers les travaux présentés, on constate que les composants de processus possèdent deux forces : ils permettent d'une part de construire des processus de développement par assemblage ; et d'autre part, ils permettent d'échanger ou de vendre sous forme de boîtes noires les savoir-faire à d'autres entreprises [WLH05, Mek08] sans prendre le risque de voir ses compétences se dissoudre chez les concurrents.

Mieux encore, selon [Mek08], l'utilisation de composants de processus permettrait en plus de formaliser les interactions entre les différents maîtres

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

d'œuvre car il serait tout à fait envisageable de construire et d'orchestrer des processus à travers des composants répartis (figure 3.20).

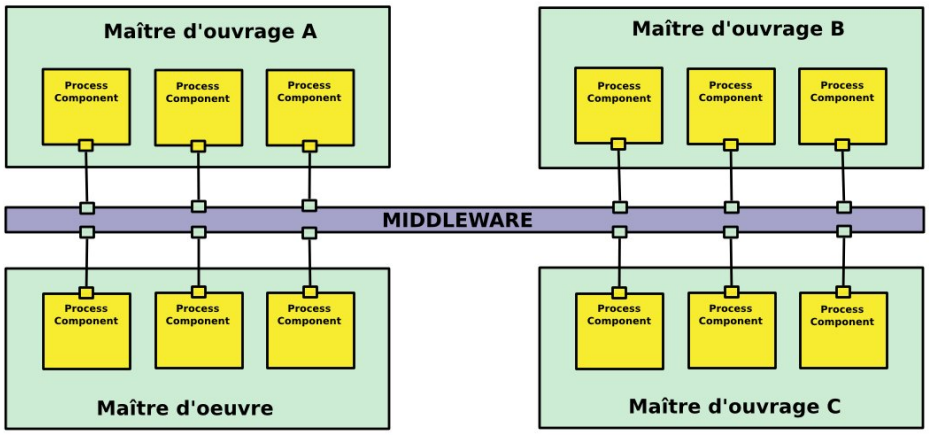


FIGURE 3.20 – formalisation des rapports maître d'œuvre / maîtres d'ouvrage

Enfin, les travaux présentés dans [BDG05, BDGM08] établissent un lien entre les composants de processus et les préoccupations de l'ingénierie des lignes de produits (PLE, Product Line Engineering). Ce lien permet, selon leurs auteurs, de traiter les manques des deux communautés, IDM et PLE, dans la capacité d'adapter les savoirs-faires dans les processus de développement logiciel. Pour ce faire, les auteurs ajoutent aux *MDA Components*, présentés ci-dessus, des interfaces de configuration permettant de spécifier la manière d'adapter la connaissance qui est encapsulée dans le composant. Cette adaptation permet de traiter la gestion de nouvelles lignes de produits.

3.5 Conclusion

Les travaux présentés ci-dessus contribuent à améliorer le contrôle des processus par leur automatisation. Dans ce cadre, l'utilisation de modèles et de transformations de modèles ne permettent de satisfaire cet objectif que si les activités et les artefacts des processus sont clairement définis et formalisés. Dans ce sens, il reste encore beaucoup d'efforts à fournir pour permettre une élicitation des processus à travers l'exécution de modèles. La satisfaction de cet objectif permettrait d'améliorer le contrôle et la qualité des processus de développement actuels et j'espère que cette modeste thèse y contribuera.

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Chapitre 4

Contrôle des processus par les modèles

Sommaire

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

4.1 Introduction

La présentation des problèmes de maîtrise de processus auxquelles sont confrontées les entreprises de conception conjointe logiciel / matériel (chapitre 2) et des solutions apportées par les différentes communautés de l'ESL, de l'ingénierie des processus et de l'IDM (chapitre 3) nous a permis de faire ressortir certains manques. En effet, si ces travaux permettent effectivement d'améliorer les processus de conception conjointe, les solutions apportées restent somme toute locales. Aussi, à travers l'étude de l'état de l'art, j'ai pu constater qu'il manquait des concepts essentiels à l'analyse de la causalité des processus, ainsi qu'à la représentation des artefacts de la conception conjointe nécessaires à une meilleure gestion des activités de l'ESL.

Dans la première partie de ce chapitre, je présente ma contribution à la représentation des processus de conception conjointe. Cette contribution permet de mieux expliciter la causalité des processus par une séparation des aspects intentionnels de leurs réalisations. Elle clarifie également la définition des artefacts des processus centrée sur la définition de modèle et leur associe un cycle de vie. Ce travail de clarification permet de raffiner la définition des activités, des contraintes et des composants de processus. Dans la seconde partie de ce chapitre, je présente ma contribution à la représentation des artefacts de processus de conception conjointe qui explicite le choix d'un modèle de calcul, potentiellement hétérogène, avant allocation du système sur une plateforme d'exécution.

Les contributions proposées ont fait l'objet de plusieurs publications : [KCA07], [KAC07], [KVS+08], [KCAS09], [KA09], [kou09], [KCLLA10], [KCLLL10], [KC10].

4.2 MODAL pour la modélisation des processus

Comme mentionné dans le chapitre 1, le premier axe de travail de la thèse consistait à fournir une méthodologie de conception conjointe qui soit plus en phase avec l'état de l'art. Quand cela fût fait, j'ai décidé d'aller plus loin dans la capitalisation et la maîtrise des processus de conception conjointe. Aussi, après avoir fait l'état de l'art sur les formalismes permettant de représenter les processus, mes choix se sont portés sur le langage SPEM standardisé

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

par l'OMG. Ce choix est justifié car, bien que dédié à la représentation des processus techniques de développement logiciel ou système, SPEM est un langage à large champs et extensible. Ce n'est que lorsque j'ai utilisé ce langage pour modéliser le processus proposé dans le but de l'opérationnaliser que j'ai fait le constat de certains manques.

Les contributions présentées dans les sections suivantes ont pour objectif de répondre à ces manques et faire converger les techniques de l'ingénierie des processus et ceux de l'IDM. À travers ce travail, je voulais :

- Définir les intentions qui guident la définition ou l'amélioration des processus,
- Définir les stratégies qui réalisent les intentions du processus par l'utilisation de compétences, d'outils ou de langages en relation avec un niveau d'abstraction donné,
- Raffiner la définition des artefacts du processus centré sur la définition de modèles conformes à des méta-modèles en leur associant un cycle de vie,
- Raffiner des contraintes de processus pour mieux cadrer les activités des processus par, d'une part, une formalisation des règles métiers ou des normes de développements et par, d'autre part, l'identification de sous-ensembles de méta-modèles applicables aux activités du processus,
- Raffiner la définition des composants et permettre ainsi une meilleure encapsulation des expertises.

4.2.1 Les intentions et les stratégies

La définition et l'amélioration de processus de développement de systèmes d'information ne sont pas des tâches aisées. Elles nécessitent une profonde compréhension de tous les tenants et les aboutissants des processus [IGI+05, BDK+08, OMR09] ainsi que l'établissement d'un consensus entre toutes les parties prenantes du développement [KRF09].

Même si le langage SPEM permet une représentation fine des processus, il ne permet pas de faire ressortir la causalité qui guide la définition des processus et qui cadre leur amélioration. En d'autres termes, il est possible avec SPEM de représenter le "comment" des choses sans vraiment jamais s'intéresser au "pourquoi" et au "quoi". Par exemple, on peut représenter une activité du processus avec ses entrées / sorties ainsi que les flots de données qui la caractérise, comprenant des structures de contrôle et un langage d'action dé-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

dié, mais sans jamais avoir accès aux raisons qui ont poussé à la définition de cette activité. De ce fait, les aspects intentionnels des processus se trouvent implicitement couplés aux choix liés aux évolutions technologiques. C'est un problème qui ajoute de la complexité à la maintenance et à l'évolution des processus [TLB+09].

Dans ce cadre, je pense qu'il est nécessaire de décorréler la définition des intentions de leur réalisation afin de permettre une meilleure capitalisation du savoir-faire et de disposer d'un meilleur cadre pour gérer les risques liés aux évolutions techniques. Il est en effet important de faire la distinction entre les objectifs que tentent d'atteindre les parties prenantes des processus de développements des plans qui sont mis en œuvre pour y parvenir. Dans notre extension du langage SPEM, MODAL, nous avons réifié les concepts d'objectif et de plan sous les termes, respectivement, d' "intention" et de "stratégie".

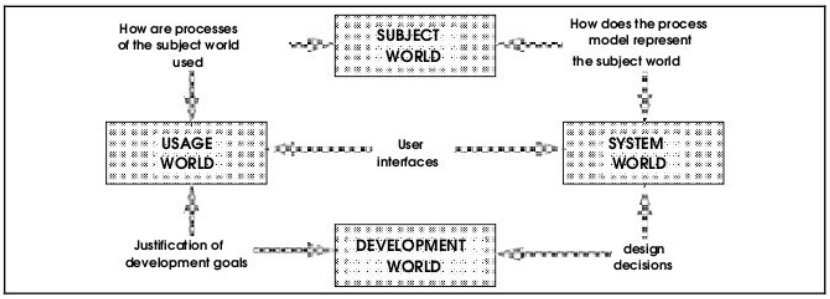


FIGURE 4.1 – Four-Worlds Framework

L'introduction de ces concepts a été inspirée par les travaux [Rol98] où les auteurs proposent un cadre intellectuel nommé "Four-Worlds Framework" permettant de mieux gérer la complexité des processus de développement. La figure 4.1 résume cette approche. Brièvement, dans cette approche, les objets du monde réel sont des sujets d'étude du monde du système. Le monde du système gère la complexité des objets du monde réel par des représentations à différents niveaux d'abstraction. Le monde des usages rassemble la définition des activités devant être exécutées par les parties prenantes du processus pour réaliser le système. Les parties prenantes exploitent les représentations du monde du système avec des objectifs à atteindre (intentions) en mettant en œuvre des plans (stratégies) permettant de satisfaire ces objectifs. Enfin, le monde du développement contient les entités concrètes du processus chargés d'exécuter les stratégies définies par le monde des usages.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

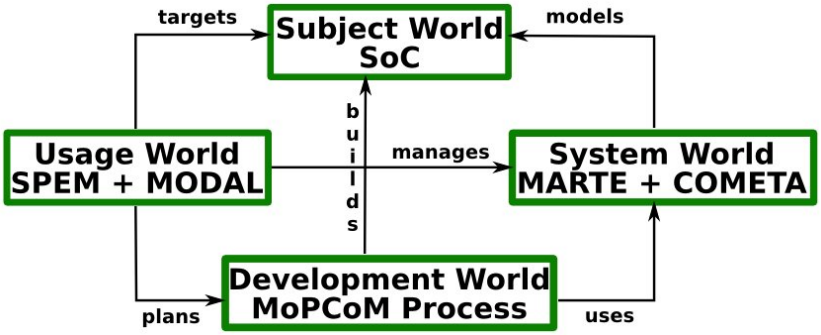


FIGURE 4.2 – Spécialisation du Four-Worlds Framework

Cette vision correspond parfaitement aux objectifs de maîtrise des processus que nous nous sommes donnés. On peut par ailleurs voir sur la figure 4.2 l'adaptation de ce cadre intellectuel davantage axé sur la réification de ces concepts et sur l'utilisation de l'approche IDM. Dans ce schéma, le monde réel représente le produit que l'on veut construire. Le monde du système contient des représentations de ce système à différents niveaux d'abstraction pour en favoriser l'analyse. Il est nécessaire de représenter dans ce monde tous les aspects pertinents menant à une construction correcte du système : il s'agit autant de représenter les aspects fonctionnels que les aspects non fonctionnels du système en développement. À ce niveau, nous verrons que l'emploi du langage MARTE complétée de notre extension COMETA convient à de telles représentations. Le monde des usages gère les modèles du monde du système conformément à un plan défini par l'ensemble des parties prenantes. Ce plan défini l'ensemble des activités identifiées par leurs entrées / sorties et des responsabilités. Ces activités de conception et d'analyse sont exécutées dans le monde du développement et utilisent les objets du monde du système afin de construire le produit final. Nous utilisons SPEM complémenté de notre extension MODAL afin de définir les objectifs et les plans des processus de développements ainsi que les activités de conception et d'analyse nécessaire à la construction du système. Dans les deux paragraphes suivants, je m'attache à donner des définitions plus précises des concepts d'intention et de stratégie.

Les intentions

Les processus sont l'essence de la réalisation d'un produit. Ils sont guidés par des choix motivés par des raisons. Ces raisons sont introduites sous

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

le concept d'intention. Il existe deux domaines qui s'intéressent à la notion d'intention : le domaine de l'intelligence artificielle et celui des systèmes d'information.

Le domaine de l'intelligence artificiel s'intéresse à la réalisation de programmes par des agents rationnels. Un agent est dit rationnel si il utilise sa perception du monde extérieur pour décider des actions à réaliser [Woo00]. Plus précisément, dans [Bra87] les auteurs proposent un modèle nommé BDI (Belief, Desire, Intention) qui utilise le processus de raisonnement qu'un être humain utilise tous les jours. La croyance (Belief) représente ce que l'agent croit connaitre du monde. Le désire (Desire) représente l'ensemble des états que l'agent essaie d'atteindre. Enfin, l'intention représente un but intermédiaire ou final que l'agent veut satisfaire.

Dans cette approche de l'intelligence artificielle, les intentions d'un agent dépend de sa perception de son environnement. Dans le cadre de la définition de méthodologies, l'environnement des organisations qui va mettre en œuvre les méthodologies n'est à priori pas connu et c'est la raison pour laquelle les méthodologies standards ne peuvent finalement qu'offrir un cadre de développement générique qui devront ensuite être adaptés par les organisations qui les appliquent.

Dans le domaine des système d'information, dans [CR99], l'intention est définie comme :

> "An intention is a goal, an objective that the application engineer has in mind at a given point of time"

Selon les auteurs de cette définition, un processus de développement peut être vu comme une carte d'intentions reliées par des stratégies. Dans cette représentation, les stratégies permettant de relier deux intentions peuvent être nombreuses, avec pour conséquence d'ajouter aux processus des propriétés de reconfiguration dynamique. Mais contrairement aux auteurs de cette définition, je ne pense pas que les intentions construisent les processus. je pense que les intentions offrent un cadre permettant de guider la définition ou l'amélioration des processus.

Enfin, dans [MFB09], les auteurs discutent des intentions dans la modélisation. Ils définissent l'intention comme suit :

> "The intention of a thing thus represents the reason why someone would be using that thing, in which context, and what are the expectations vs. that thing. It should be seen as a mix-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

> ture of requirements, behavior, properties, and constraints, either satisfied or maintained by the thing."

Dans cette définition, l'intention est davantage rattachée aux artefacts de la modélisation et représente en quelque sorte la raison d'être d'un modèle dans un processus de conception. Ces travaux établissent des relations entre les modèles dans le processus afin de mieux comprendre le cheminement intellectuel que les modèles servent. La modélisation intentionnelle apporte ainsi des réponses sur le qui et le quoi de la modélisation et non sur le comment. Contrairement aux auteurs de cette définition, je ne cherche pas à expliquer un raisonnement intellectuel à travers en ensemble de modèles mis en relation. Mon approche est davantage prescriptive dans la mesure où les modèles servent justement un raisonnement intellectuel préexistant.

La figure 4.3, extraite de ma contribution MODAL, définit les intentions comme un cadre de définition de méthodologie générique pour l'élaboration ou l'amélioration de processus de développement. Les intentions définissent des objectifs méthodologiques établis de concert par les parties prenantes du processus. Elles appliquent des guides méthodologiques génériques devant être formalisés par le choix de stratégies. On distingue parmi les règles méthodologiques les guides spécifiques à l'organisation (règles de codage, convention de nommage, etc.), les standards de développement ou les normes de certification. Associé aux intentions, des conditions de satisfaction devront faire l'objet d'une formalisation dans la définition des stratégies correspondantes.

Les liens de satisfaction définissent des liens de causalités dans les objectifs méthodologiques définis par les intentions. Par exemple, dans la figure 4.4 extraite du modèle de processus MoPCoM SoC / SoPC, l'intention "Design Coarse Grain Architecture" ne peut être satisfaite que sur la base de la satisfaction de l'intention "Define Fine Grain Architecture". De la même manière, les liens d'utilisation précise pour chaque intention quelles sont les compétences requises, quelles règles méthodologiques génériques vont cadrer la réalisation de l'intention ou quel type d'outil il sera nécessaire d'utiliser pour y parvenir.

En définitives, les cartes d'intentions décrivent de manière semi-formelle le cheminement intellectuel guidant la construction d'un produit. En plus de cadrer la définition des processus, elles permettent de générer automatiquement la documentation des guides méthodologiques, spécifiant les responsabilités, les règles méthodologiques, les contraintes de certification, la définition des outils ou des produits. Dans le paragraphe suivant, nous voyons de quelle

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

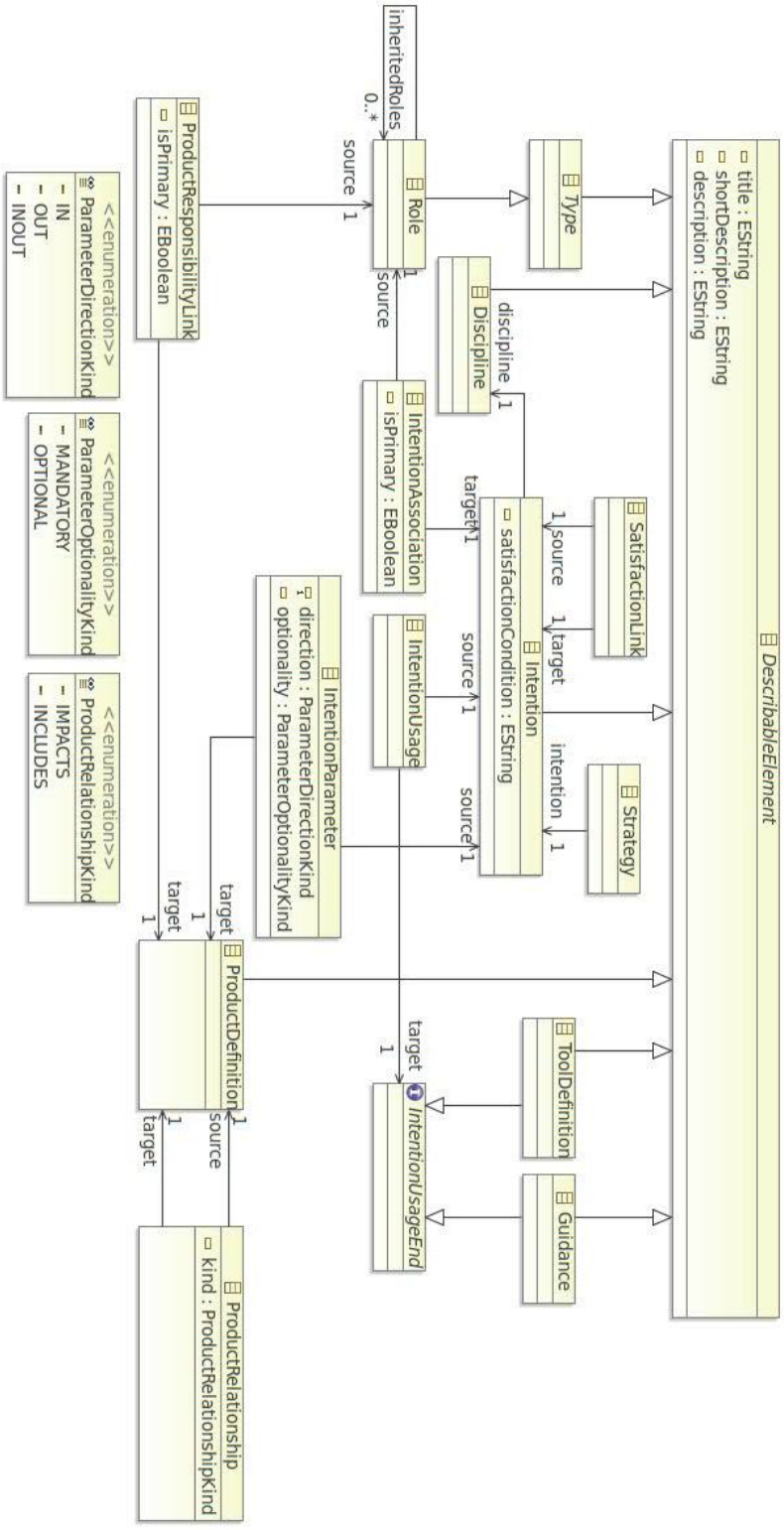


FIGURE 4.3 – Syntaxe abstraite de la notion d'intention

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

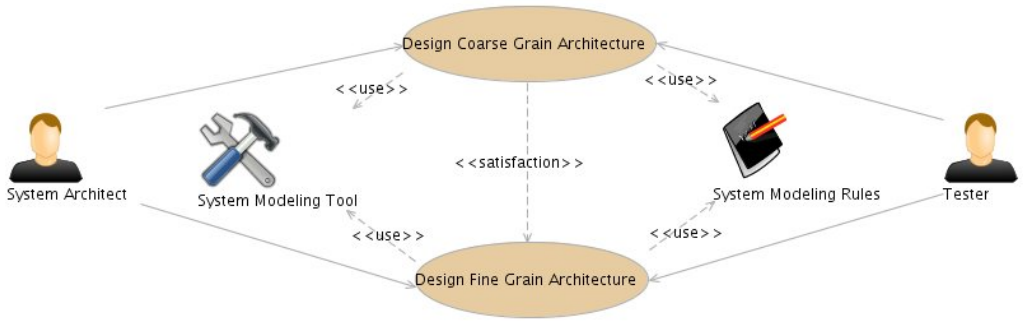


FIGURE 4.4 – Exemple de carte d'intentions

manière les stratégies mettent en œuvre les intentions.

Les stratégies

La définition d'une stratégie commence par l'analyse de l'intention qu'elle tente de satisfaire. Elle établie sur la base d'un consensus entre les parties prenantes un plan initial menant à la satisfaction de l'intention. Ce plan est en fait un synopsis représentant un ensemble de ressources ou d'activités interconnectés dont la collaboration vise la satisfaction de l'intention.

Alors que les intentions utilisent des définitions génériques les rendant indépendantes des technologies, les stratégies formalisent les intentions par des projections sur des espaces technologiques particuliers. Ainsi, les stratégies établissent un passage entre un monde informel, constitué d'entités génériques décrites de manière textuelle, à un monde formel, constitué d'entités compréhensibles et manipulables par les machines. Une telle projection permet d'une part une opérationnalisation des processus, et d'autre part, une formalisation des règles méthodologiques sous forme de contraintes de processus interprétables par la machine.

Par exemple, le tableau 4.1 montre quelques exemples de résolutions technologiques spécifiées par le choix de stratégies pour le passage d'un monde informel à un monde formel.

La figure 4.5 montre le diagramme de stratégie représentant le pendant du diagramme d'intention présentée à la figure 4.4. Chaque stratégie vise à satisfaire son intention par un ensemble de résolutions technologiques. Elle réutilise alors des savoir-faire existants encapsulés dans des composants de processus ou en définit de nouveaux à travers des diagrammes d'activités

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Artefact d'intention	**Projection technologique**
Outil de modélisation système	Papyrus
Outil de métamodélisation	KerMeta
Règle méthodologique	Contrainte OCL
Architecture système	Modèle SysML
Plateforme d'exécution	Modèle MARTE HRM

TABLE 4.1 – Exemples de résolutions technologiques

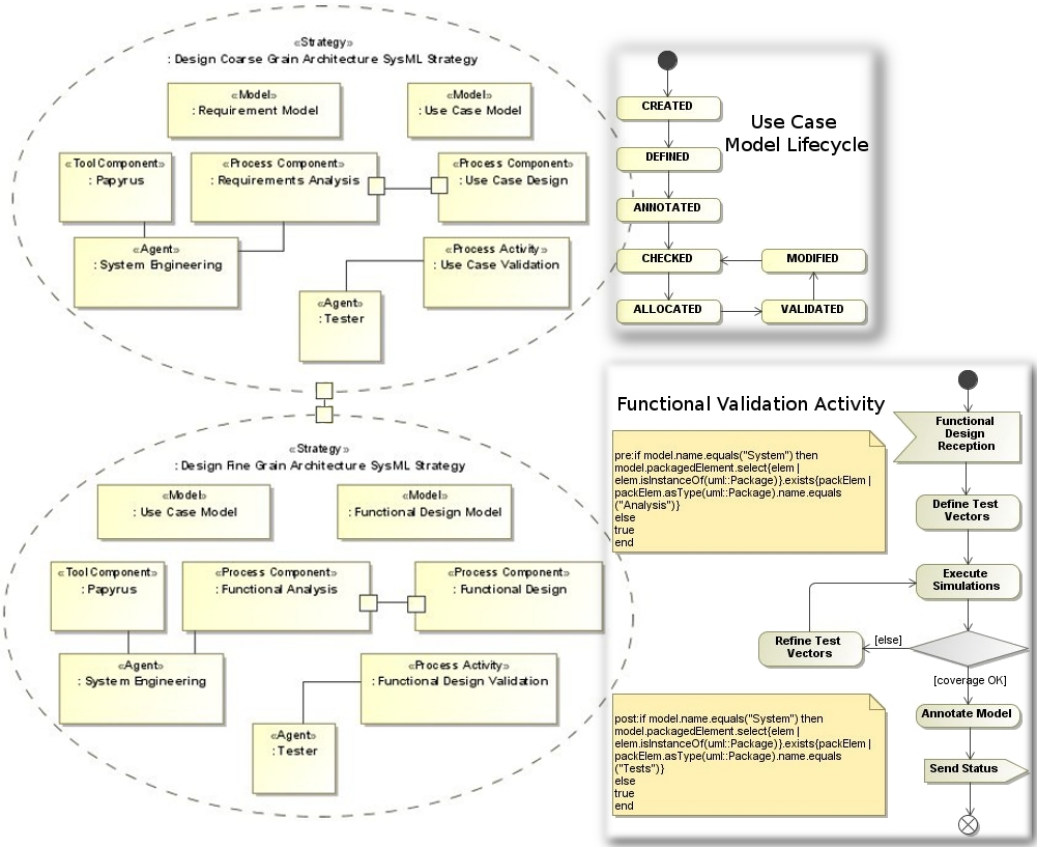


FIGURE 4.5 – Exemple de stratégie

manipulant des artefacts de modélisation. Dans l'état actuel des choses, la formalisation de la connaissance n'est pas toujours possible. En effet, certains de nos choix dans la réalité ne peuvent être faits que sur la base de nos expériences passés et de notre instinct. Dans ce cadre, le langage d'action utilisé dans les diagrammes d'activités doivent permettre des interactions

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

homme / machine.

4.2.2 Les modèles comme artefacts de processus

Suivant les communautés d'ingénierie des processus (Systèmes d'information, Workflow, etc.), les approches peuvent être centrées sur les produits comme sur les activités ou même la structure des organisations. Je pense qu'un processus de développement basé sur l'ingénierie des modèles doit avant tout être centré sur les modèles.

Dans le langage SPEM, la notion de produit est réifiée sous le concept de "WorkProductDefinition" dont la sémantique informelle est définie comme :

> "Work Products are in most cases tangible work products consumed, produced, or modified by Tasks. They may serve as a basis for defining reusable assets. Roles use Work Products to perform Tasks and produce Work Products in the course of performing Tasks. Work Products are the responsibility of Role Definitions, making responsibility easy to identify and understand, and promoting the idea that every piece of information produced in the method requires the appropriate set of skills. Even though one Role Definition might own a specific type of Work Product, other roles can still use the Work Product for their work, and perhaps even update them if the Role Definition instance has been given permission to do so."

Dans l'approche SPEM, un produit est un artefact du processus sous la responsabilité d'un rôle et qui peut être produit, consommé ou transformé par une activité. Il me semble plus cohérent que dans une approche IDM, cette définition devrait être raffinée et recentrée sur la notion de modèle. Mieux encore, je pense que l'association d'un cycle de vie aux méta-classes permettrait en plus un contrôle plus fin des activités du processus.

La figure 4.6, extraite de l'extension MODAL, illustre ma définition d'un artefact de processus. Les artefacts manipulés par les processus sont des modèles typés par des méta-classes [STE07]. Aux méta-classes sont associées un cycle de vie qui définit l'ensemble des états possibles dans lesquels un artefact conforme peut se trouver. Les transitions entre ces états peuvent alors être gardées par des contraintes formalisées.

La notion de "CombinedMetamodel" qui apparaît sur la figure est une notion dont nous ne donnons pas encore de définition précise parce qu'elle

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

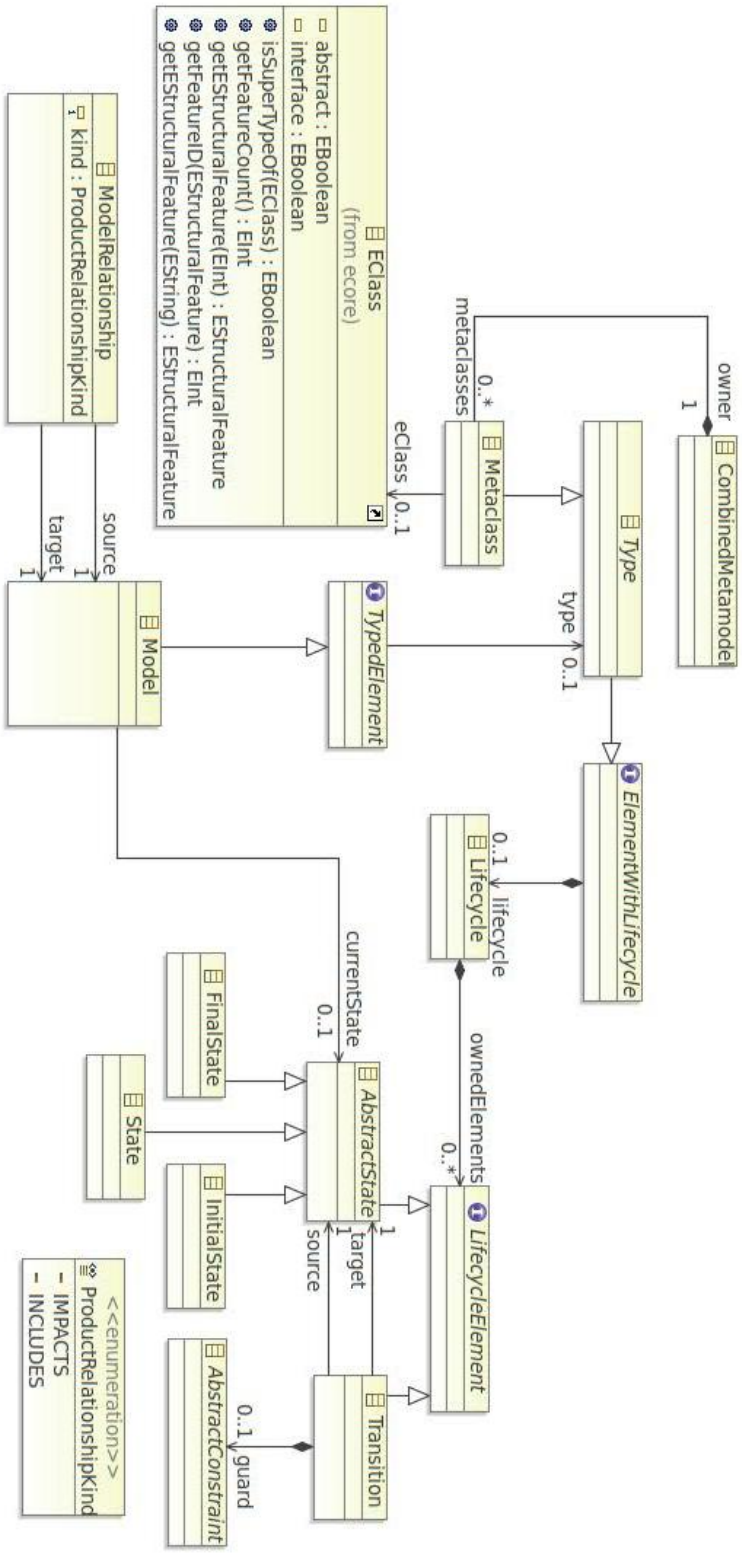


FIGURE 4.6 – Raffinement de la notion de produit

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

dépasse le cadre des travaux réalisés durant cette thèse. Je préciserai juste que cette notion a été introduite suite à des expérimentations qui ont fait apparaître le besoin d'identifier des sous-ensembles restreints de méta-modèles pour gérer différents niveaux d'abstraction. J'ai donc proposé de définir des méta-modèles spécifiques à des niveaux d'abstraction qui combinent des concepts provenant de plusieurs méta-modèles. Par exemple, pour gérer les activités du système, j'ai utilisé un sous-ensemble du langage SysML pour la gestion des exigences combiné à un sous-ensemble du langage MARTE pour la formalisation des propriétés non-fonctionnelles. Il y a aujourd'hui une thèse en cours à l'ENSIETA qui traite ce sujet.

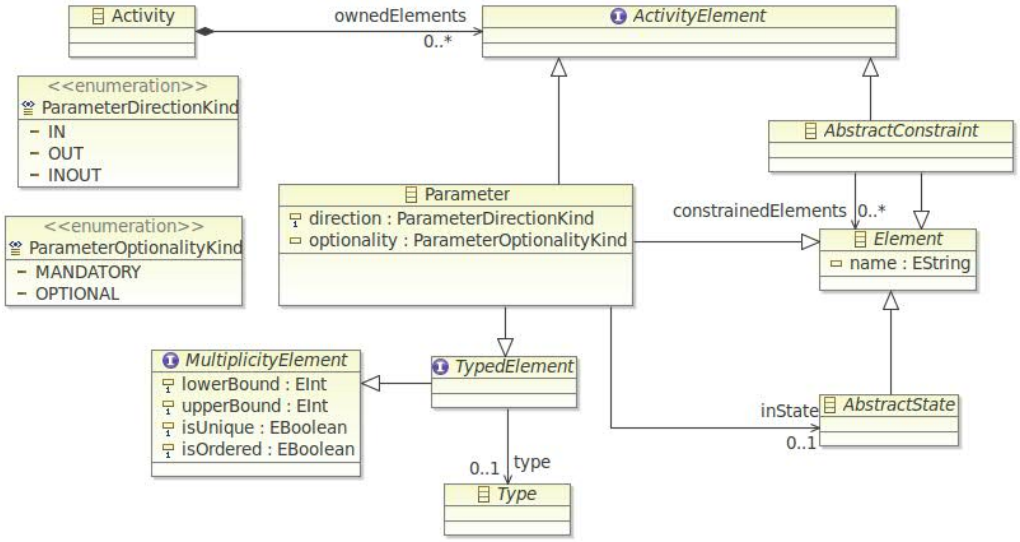


FIGURE 4.7 – Raffinements des paramètres des activités

Cette proposition permet également de raffiner la définition des activités de processus en rattachant la définition des paramètres des activités à la notion de modèle. La figure 4.7 illustre ce raffinement. Comme les produits, les paramètres des activités sont typés par des modèles. La propriété "inState" du paramètre spécifie que le modèle passé en lieu et place du paramètre devra se trouver dans l'état spécifié au runtime. Ce raffinement permet alors un contrôle plus fin des activités du processus.

Enfin, comme dans SPEM, il existe entre les modèles des liens de différentes natures permettant de mieux capturer et comprendre les relations qui

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

lient les différents artefacts de processus. On retrouve cette idée dans les travaux de [MFB09] présentés ci-dessus où les auteurs établissent un ensemble de relations entre les artefacts du processus sous la perspective de l'ingénierie des modèles.

4.2.3 Les contraintes de processus

Associée aux stratégies et aux produits, une attention particulière doit être accordée à la définition de contraintes de processus. En effet, les contraintes définies dans les stratégies ou les activités du processus participent à garantir la cohérence du processus. L'expression des contraintes de processus porte sur les éléments de décomposition du processus comme les produits ou les activités. Or, sans une définition claire de ces éléments, les seules contraintes que l'on est capable d'exprimer sont des contraintes exprimées en langage naturel, ce qui rend l'opérationnalisation des processus impossible.

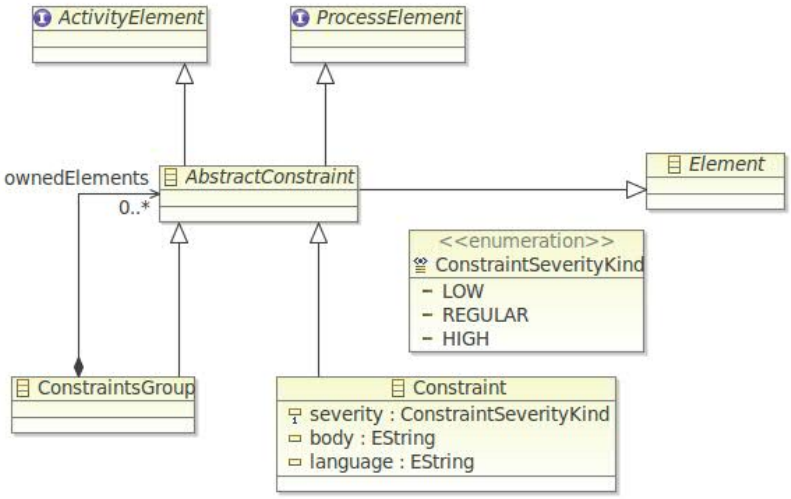


FIGURE 4.8 – Les contraintes de processus

La figure 4.8 extraite de l'extension MODAL illustre la définition des contraintes rattachées aux éléments du processus. Bien que les contraintes de processus permettent de cadrer les activités du processus, tout ne doit pas pour autant être gravé dans le marbre : il est parfois nécessaire de laisser aux ingénieurs quelques degrés de latitudes sur l'application de certaines contraintes. Ainsi, la propriété "severity" de la méta-classe "Constraint"

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

illustre cette idée de gradation qui exprime que certaines contraintes doivent absolument être respectées (severity=HIGH) alors que la violation de certaines autres (severity=LOW) pourra être tolérée.

La notion de "ConstraintsGroup", apparaissant dans la figure, réifie la notion de groupe de contraintes associées à un domaine ou à un niveau d'abstraction donné. La nécessité d'introduire cette notion correspond au besoin de restreindre l'utilisation des langages de modélisation qui est apparue lors de mes expérimentations. Un exemple typique d'une telle restriction concerne le méta-modèle UML dont l'emploi permet de couvrir un grand nombre d'activités de conception et d'analyse. Au vu du nombre de concepts présents dans UML, il nous est apparu essentiel d'identifier pour chaque activité du processus des sous-ensembles nécessaires et suffisants dont les ingénieurs ont besoin pour remplir leurs missions. Grâce aux groupes de contraintes, nous avons pu formaliser ces règles d'utilisation et nous sommes à même de vérifier de manière automatique que les règles méthodologiques sont bien respectées. Le tableau ci-dessous montre quelques exemples de contraintes liées à l'utilisation du langage MARTE spécifiées dans le cadre du projet MoPCoM : la première colonne spécifie la source de la règle méthodologique, la seconde sa spécification informelle, la troisième le concept de UML / MARTE permettant de la formaliser, la quatrième sa formalisation en langage KerMeta et enfin la dernière son niveau de sévérité.

Enfin, au delà de la restriction des langages, et comme je l'ai mentionné dans l'introduction, les contraintes de processus permettent de formaliser les contraintes de certification. C'est là un aspect très important pour les sociétés qui travaillent dans des secteurs sensibles comme le domaine militaire ou le domaine de l'avionique. J'ai mentionné dans le chapitre "Contexte et motivations" que l'application de normes de certification comme la norme DO-178B de l'avionique a pour conséquence des surcoûts en temps de conception allant de 75 à 150%. Ma proposition permet alors de diminuer les efforts consentis pour la vérification de ces contraintes.

En définitive, cette proposition permet de réaliser ce qui auparavant était impossible : formaliser les contraintes imposées par le cadre méthodologique et les normes de développement par un passage d'un monde informel à un monde formel.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Source	Contrainte	Contexte	Formalisation	Sévérité
Référentiel	Une plate-forme doit au moins posséder une horloge	HW Resource	self.ownedElements.exists(e| e.isInstance (Property) and e.type.stereotypes.contains (“HW_Clock”))	REGULAR
DO-258	Le corps des opérations ne doit pas être vide	Operation	self.body <> null	HIGH
Contrat	le système doit être livré dans un an	Process	self.duration <= (1, year)	HIGH

TABLE 4.2 – Formalisation des contraintes

4.2.4 Les composants de processus “IDM”

La clarification de la définition des artefacts de processus, alignés sur la définition de modèles, permet un meilleur contrôle des activités du processus, notamment par l'usage de contraintes formalisées. Dans cette section, je montre en quoi ce travail permet d'améliorer l'encapsulation et la diffusion de patrons de processus à travers des composants de processus.

La figure 4.9 illustre la définition des composants de processus selon le langage SPEM qui le définit comme :

> “SPEM 2.0 supports replaceable and reusable Process Components realizing the principles of encapsulation. Certain situations in a software development project might require that concrete realizations of parts of the process remain undecided or will be decided by the executing team itself (e.g., in outsourcing situations). Process components support an assembly mechanism that is based on the black box principle. At any point during a project, the component realization detailing the work can be added to the process. The component approach also allows that different

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

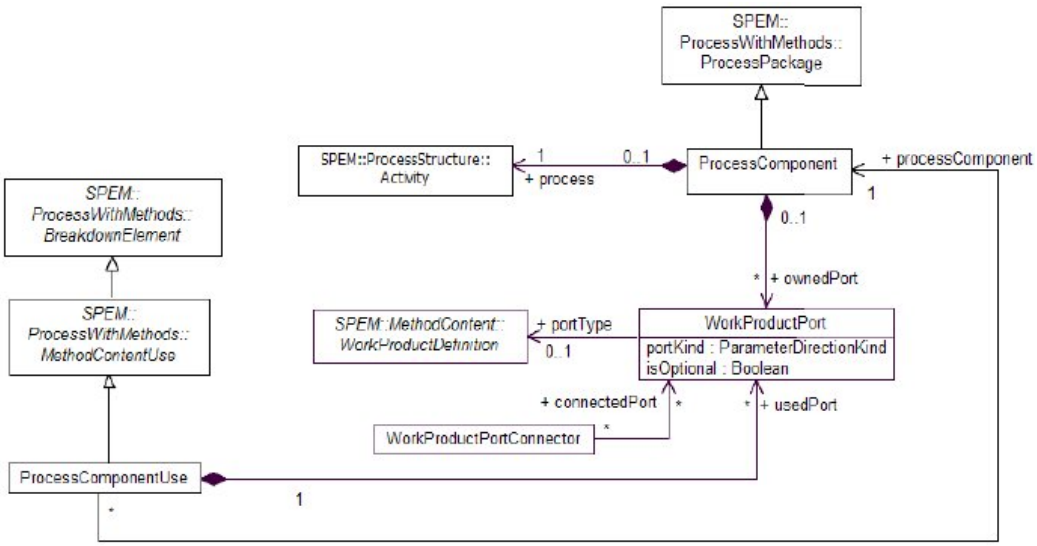


FIGURE 4.9 – Les composants de processus SPEM

> styles or techniques of doing work can be replaced with one another. For example, a software code output of a component could be produced with a model-driven development or a code-centric technique. The component concept encapsulates the actual work and lets the development team choose the appropriate technique and fill the components realization with their choice of Activities that produce the required outputs."

Dans cette définition, le composant de processus encapsule la définition d'une activité du processus et possède des ports caractérisés par le type des artefacts produits, consommés ou transformés. Dans cette vision, un processus peut être construit par un assemblage de composants de processus qui s'échangent des produits à travers des connecteurs dédiés ("WorkProductPortConnector").

Une telle définition manque de précision car elle ne dit rien sur l'environnement du composant, ni même sur l'utilisation des artefacts en entrée / sortie des composants de processus. Le problème qui se pose dans cette définition est qu'elle est essentiellement orientée vers la documentation et non sur l'élicitation des processus. On constate alors que le composant de processus de SPEM n'est finalement qu'un espace de nommage qui capitalise des définitions réutilisables. Dans ce cadre, je propose de raffiner cette notion

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

afin de l'orienter davantage sur l'élicitation des processus. Aussi, je profite des apports décrits dans les sections précédentes pour renforcer la sémantique des composants de processus dans le but de les opérationnaliser.

Dans la définition que je propose, illustrée par la figure 4.10, les composants de processus permettent d'encapsuler la définition de plusieurs activités du processus, et notamment celles définies par les stratégies en relation avec les intentions qu'elles réalisent. Les composants de processus MODAL offrent dans leurs parties publiques (interfaces) la définition des services qu'ils rendent au monde extérieur. Ces services prennent en paramètres des modèles ayant un cycle de vie et sur lesquels des contraintes peuvent être exprimées. La partie privée des composants de processus représente les moyens mis en œuvre pour l'exécution des services publiés aux interfaces. Ces moyens consistent essentiellement à déléguer la réalisation de ces services à des constituants du composant ("Part") ou à exécuter une activité du composant. Ce choix de réalisation peut être spécifié au niveau du port par la mise en relation entre un service et une activité ("ServiceBinding").

Plus précisément, la partie privée des composants de processus encapsulent la définition des activités en relation avec l'ensemble des outils ou des composants de processus requis pour réaliser les services publiés. Ainsi, les composants de processus permettent de gérer un niveau d'abstraction donné en créant un lien entre l'élicitation des processus, les intentions, les stratégies, les activités, les modèles et les contraintes. La figure 4.11 présente le composant de processus de plus haut niveau. Ce composant encapsule l'ensemble de la formalisation du processus MoPCoM présenté dans le chapitre suivant.

L'approche orientée sur les services sur laquelle est basée cette définition offre un certain nombre d'avantages en termes d'opérationnalisation, en particulier pour le déploiement des composants de processus. Ces avantages n'ont pas été exploités durant cette thèse mais d'autres travaux ont pris le relais dans ce sens. Par exemple, cette contribution constitue un bon point de départ pour définir un bus dédié à l'exécution de composants de processus dans un environnement réparti.

4.3 COMETA pour l'explicitation des MoCs

L'analyse de la concurrence et de la communication dans la conception de système sur puce représente une activité très importante. En particulier, elle traite les problèmes d'interblocage ou de famine liés aux choix de conception

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

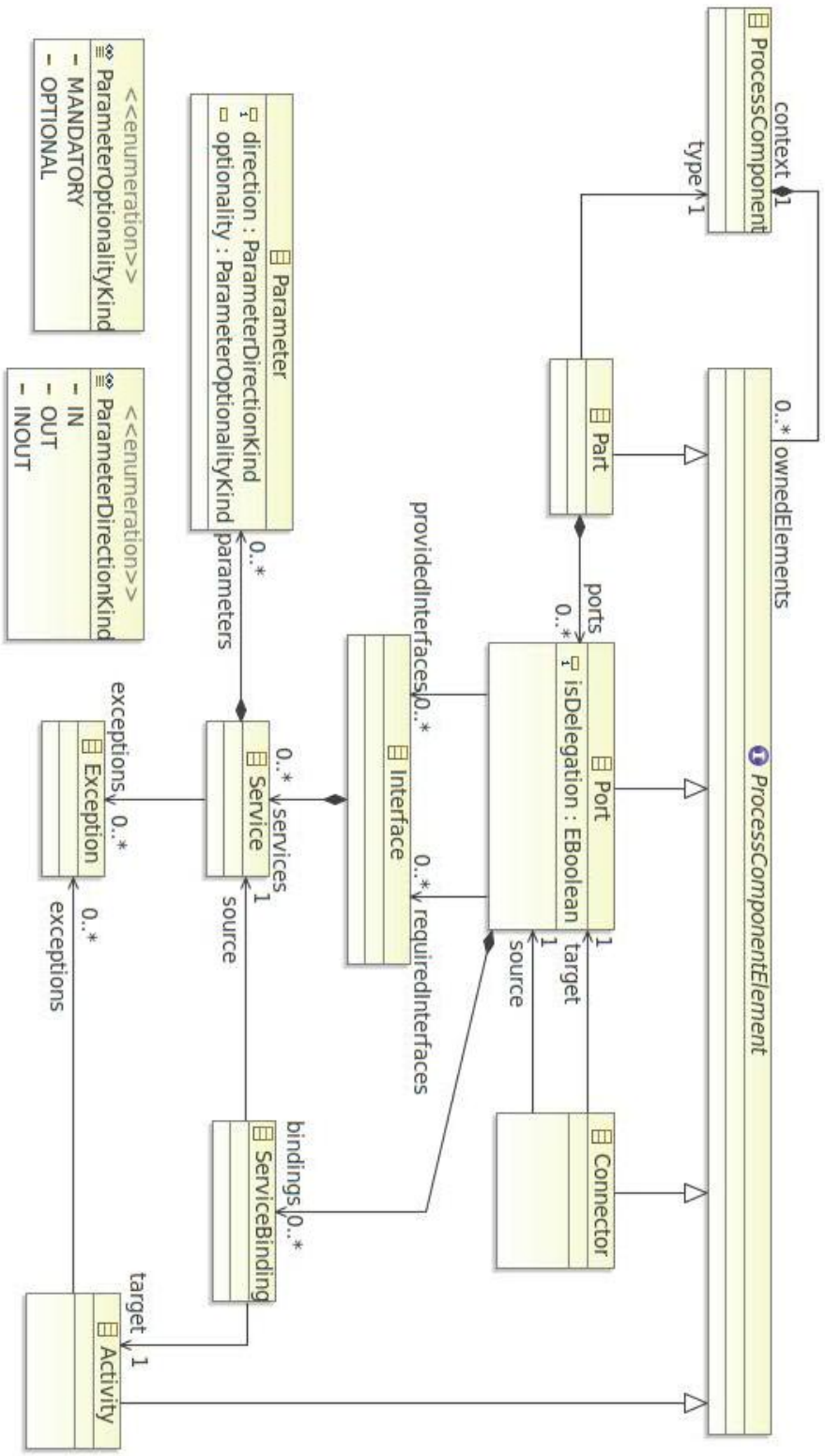


FIGURE 4.10 – Les composants de processus MODAL

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

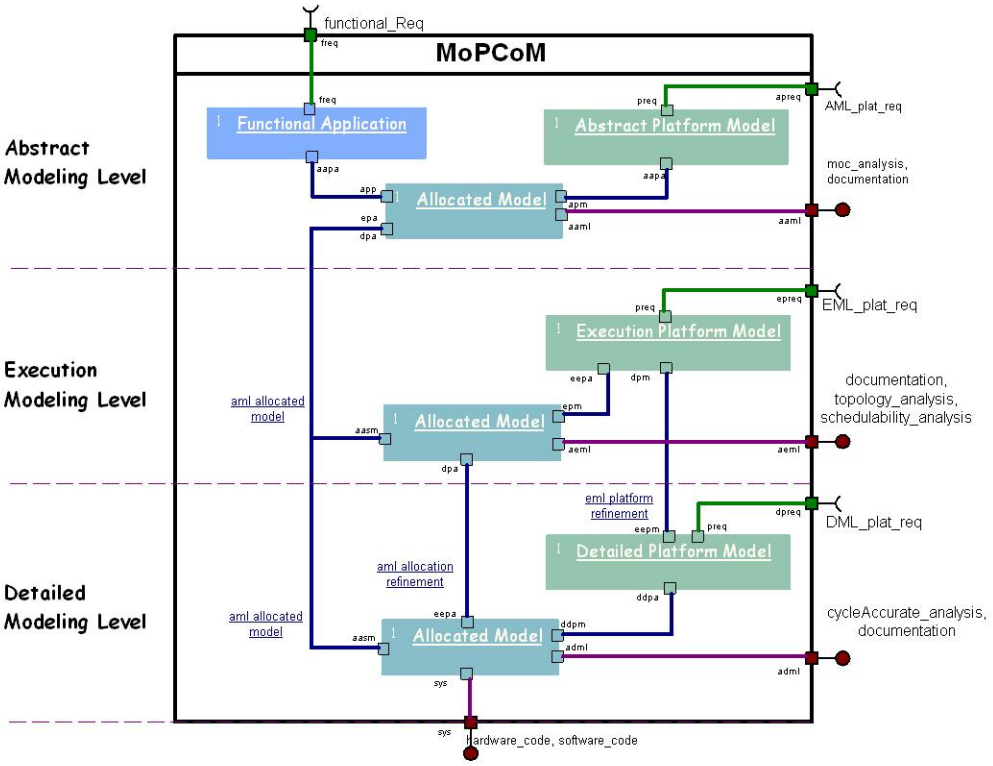


FIGURE 4.11 – Composant de processus MoPCoM SoC/SoPC

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

en termes de concurrence et de communication. De fait, la modélisation de ces aspects revêt une importance cruciale dans l'application d'une approche IDM dans la conception de systèmes sur puce.

Le modèle de calcul définit les règles qui régissent la concurrence et la communication. Il est caractérisé par au moins un modèle de temps (causal, continu, discret), une représentation de données (type de données abstraits, vecteurs de bit, signal) et un type de communication (IPC – Inter-Process Communication, variables partagées, FIFO, signaux). Le choix d'un MoC dépend d'une part des caractéristiques du système à implanter et d'autre part du type d'analyses à réaliser. Dans la mesure où les systèmes mêlent généralement plusieurs modèles de calcul, il est important de pouvoir disposer d'outils permettant la conception et l'analyse de systèmes multi-MoCs. L'outil Ptolemy [BHLM02] développé par l'université de Berkeley dans le cadre de l'ESL en est un exemple.

Il existe aujourd'hui dans l'IDM plusieurs manières de modéliser la concurrence et la communication. Nous nous intéresserons ici à celles proposées par UML et le profil MARTE. Pour exprimer la concurrence dans UML, on peut au niveau comportemental utiliser les "And-State" dans les statecharts ou les "fork node" dans les diagrammes d'activités. Au niveau structurel, le méta-attribut "isActive" de la classe supporte la notion d'entité concurrente. Enfin, il est également possible d'utiliser le pattern "Concurrence" [Dou02] dans la modélisation d'architectures système. Cependant, toutes ces techniques pour exprimer la concurrence ne permettent pas de spécifier plus précisément les mécanismes sous-jacents de communication et de synchronisation. Or ces techniques sont des pré-requis pour pouvoir mener des analyses pertinentes.

Il existe cependant des éléments de réponses apportés dans la version 1.0 du profil MARTE sur ces questions de sémantique dans le chapitre "Generic Component Model".

La figure 4.12 illustre la définition des composants génériques de MARTE. Un composant est une entité structurée caractérisée par ses ports permettant des interactions avec d'autres composants. Cette figure est complétée par la figure 4.13 qui spécifie un type de port d'interaction qui est le "ClientServerPort". Ce type de port fournie / requiert des opérations ou des réceptions d'événement. En particulier, la page 142 de la spécification précise :

> "ClientServerPorts support a request/reply communication paradigm (also called client-server model of communication), where messages that flow across ports represent operation calls or si-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

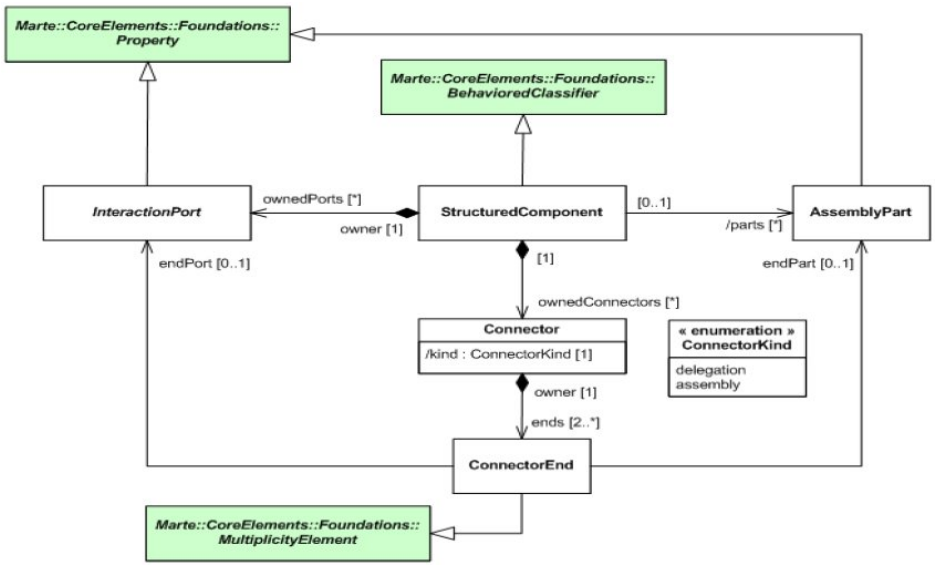


FIGURE 4.12 – Composant générique MARTE

gnals."

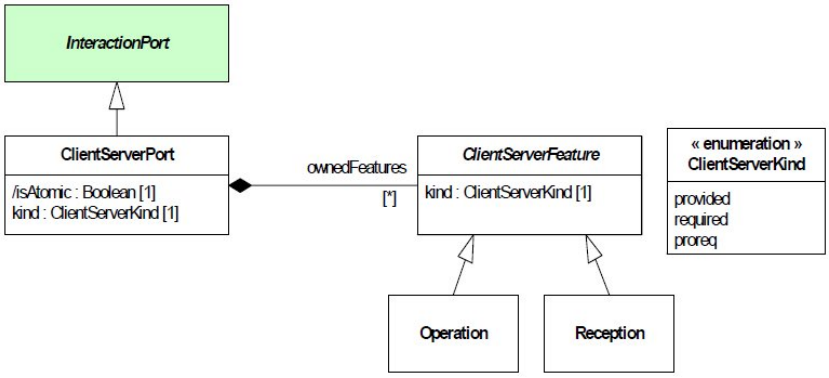


FIGURE 4.13 – Port de service MARTE

De plus, le profil MARTE propose dans le paquetage "HLAM" la notion de "RTUnit" (Real-Time Unit – Unité temps réel) (figure 4.14) qui supporte la notion d'entité concurrente temps réel. Un "RTUnit" est défini comme un bloc qui fournit/requiert un ensemble de services temps réel ("RTService") et possède au moins un comportement temps réel ("RTBehavior") avec une queue bornée / non bornée. À grain fin, les comportements temps réel sont

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

décomposés en actions temps réel (“RTAction”) auxquelles sont associés un ensemble de méta-attributs (tags) précisant leurs caractéristiques temps réel.

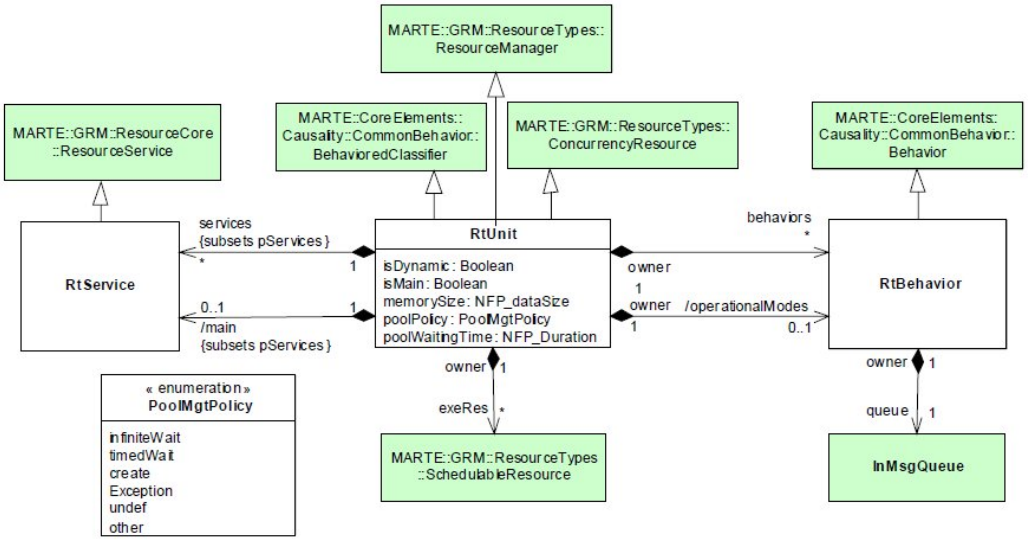


FIGURE 4.14 – Unité concurrente temps réel de MARTE

À ce niveau, toutes les communications sont en point-à-point et sont supportées par des connecteurs temps réel (“RTeConnector”) implantant des mécanismes de haut niveau (put/get ou read/write). Malheureusement, la sémantique d'exécution de ces modèles est peu ou prou traitée, que ce soit dans UML ou dans MARTE. Elle est en fait exprimée de manière informelle, laissant le soin aux outilleurs de définir comment les modèles doivent être exécutés [VP02] ou même de résoudre les nombreux points de variation sémantique que les spécifications peuvent contenir [CJ05].

Il y a dans cette partie de la spécification un début de réponse sur le support de modèles de calculs qui mériterait d'être approfondit pour mieux caractériser les modèles de calcul et en particulier pour mieux représenter l'hétérogénéité des modèles de calculs. En effet, on constate que même si MARTE améliore effectivement la modélisation des MoCs, en particulier sur les aspects temps réel, le travail des concepteurs pourrait être grandement facilité si les patrons de communications les plus fréquemment utilisé pouvaient être capitalisés et réutilisés plus facilement. À cet effet, je propose un méta-modèle dont le but est de fournir des moyens de décrire et d'analyser de nouveaux modèles de calcul. En particulier, ce méta-modèle devrait

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

permettre de :

- Générer par des transformations de modèles des librairies de MoC génériques réutilisables,
- Fournir un support d'exécution de MoCs dans l'outillage choisi,
- Améliorer la génération de code HDL pour les langages SystemC, CatapultC et VHDL.

Aujourd'hui ce méta-modèle est principalement orienté sur les aspects communication des modèles de calcul. La section suivante présente les principes de ce méta-modèle. La seconde section présente les principaux concepts introduits par le méta-modèle COMETA et la troisième section discute de la capitalisation et de la réutilisation que permettent le méta-modèle et son outillage. Enfin, je termine ce chapitre par un exemple d'utilisation.

4.3.1 Principes

La méthodologie de conception conjointe basée sur l'ingénierie des modèles proposée dans cette thèse et détaillée dans le chapitre 5, prend en entrée une spécification système formalisée et exécutable. Dans une approche IDM telle que préconisée par le MDA (modèle en Y), c'est l'allocation de cette spécification sur la plateforme et les résultats des analyses qui en découleront qui déterminera l'implantation finale du système. Or, dans cette approche, les choix portant sur l'exécution et la communication des blocs du système sont généralement fait de manière implicite. Aussi, dans le but de faciliter les choix d'allocation, je propose un niveau de modélisation intermédiaire qui explicite ces choix. Ce niveau de modélisation est basé sur l'utilisation du langage Cometa.

Le niveau de modélisation AML (Abstract Modeling Level) présenté au chapitre 5 a pour principe d'allouer des blocs systèmes sur une plateforme virtuelle spécifiant les choix en termes de concurrence et de communication. Cette plateforme est constituée d'unités concurrentes interconnectées via des canaux de communication spécifiques fournissant des services de transport de données en point-à-point. À ce niveau, la plateforme peut être considérée comme idéale dans la mesure où elle ne fait aucune hypothèse sur la limitation de ressources qui caractérisent les plateformes physiques.

L'allocation de blocs système sur une telle plateforme doit permettre alors des analyses pertinentes sur les problèmes d'interblocage ou de famine. Mais au delà des aspects analytiques, ce méta-modèle poursuit plusieurs buts :

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

– Favoriser la séparation des préoccupations entre le calcul et les communications pour faciliter les choix d'allocation,
– Préserver les comportements applicatifs à travers les allocations pour faciliter les activités de vérification,
– Fixer une sémantique opérationnelle afin de garantir des simulations uniformes, et ce quelque soit l'outil de simulation utilisé.

4.3.2 Concepts introduits

Les systèmes sur puce sont par nature hétérogènes. En conséquence, les modèles représentant ces systèmes doivent gérer cette hétérogénéité à travers différents niveaux d'abstraction. En se situant à un niveau d'abstraction donné, l'hétérogénéité est exprimée par les interactions entre les différents sous-systèmes qui constituent les systèmes en étude. La figure 4.15 illustre cette idée : on peut y voir que le comportement global du système est réalisé par les interactions des comportements de ses sous-systèmes. Chacun de ces sous-systèmes est caractérisé par son modèle de calcul (MoC – Model of Computation) qui est une explicitation des comportements et des communications.

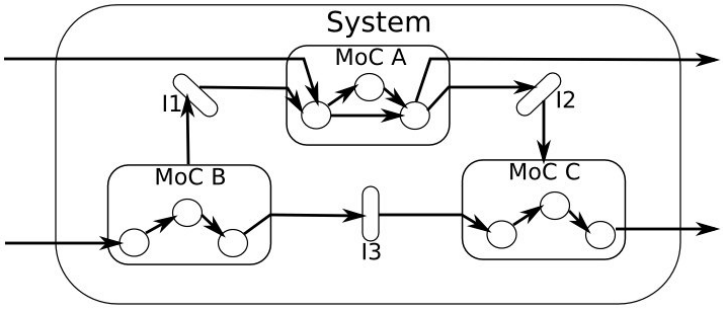


FIGURE 4.15 – Exemple de MoC hétérogène

La difficulté de l'analyse de système ne vient pas tant de l'hétérogénéité de ses sous-systèmes mais plutôt de leurs interfaçages.

La figure 4.16 illustre la définition des composants MoC dans le méta-modèle COMETA. Dans ce méta-modèle, la structure est constituée d'entités concurrentes reliées par des canaux de communications spécifiques. La notion d'entité concurrente est réifiée sous le terme "MoCComponent". La partie publique d'un composant MoC est représenté par l'ensemble de ses ports qui

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

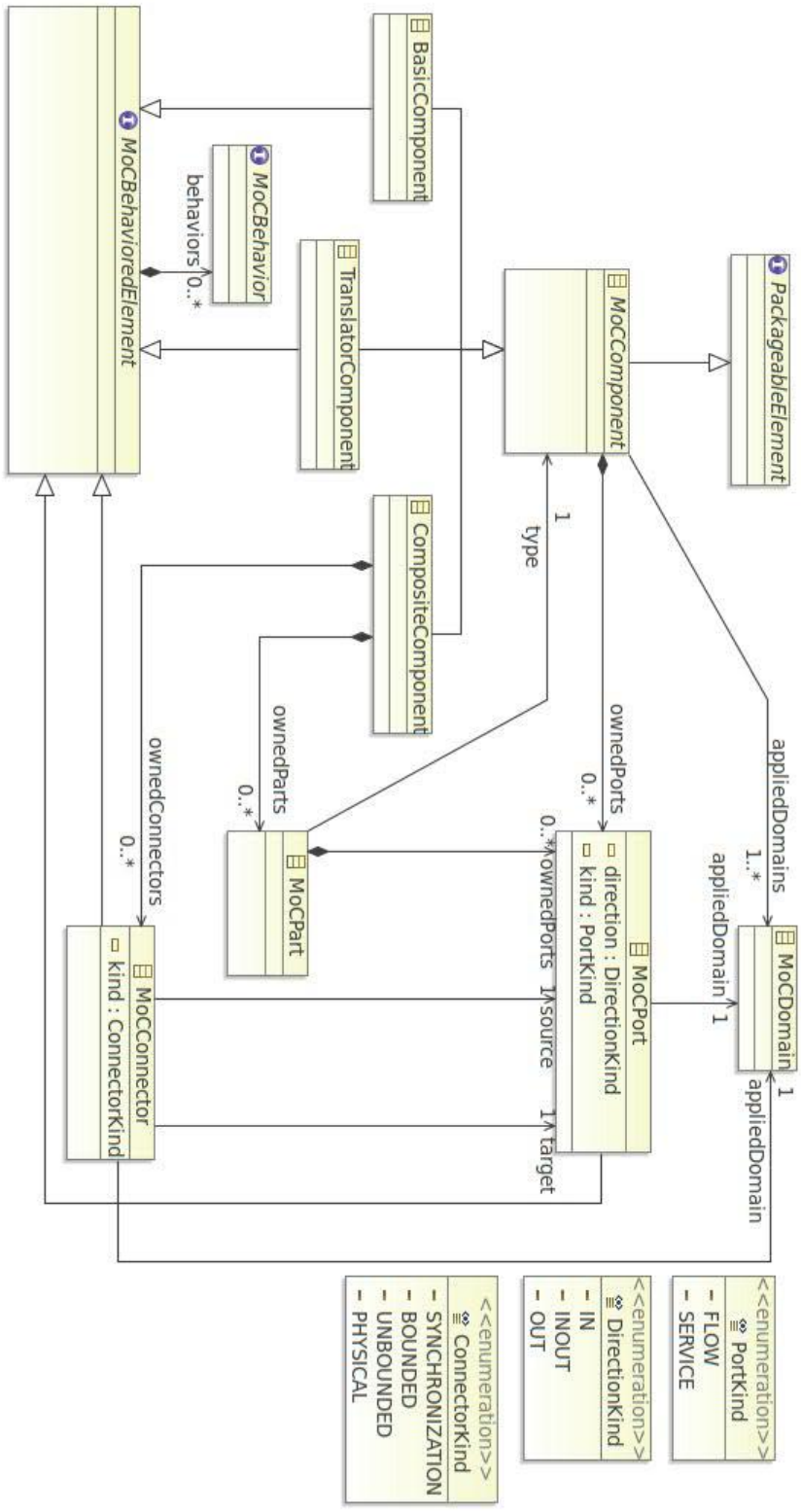


FIGURE 4.16 – Syntaxe abstraite des Composants MoC

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

offrent ou requièrent des interfaces exhibant des services de communication de haut niveau comme des services de lectures / écritures bloquantes / non bloquantes ou de synchronisation. Plus précisément, le composant MoC est une unité de structuration déclinée en trois types :

- le composant composite, constitué de constituants hétérogènes,
- le composant atomique qui est le réceptacle des blocs système,
- le composant de traduction qui permet de traduire un schéma de communication en un autre.

Le composant composite est constitué d'entités ("MoCPart") typé par des composants MoC et relié par des connecteurs ("MoCConnector") réalisant les services de communication ou de synchronisation requis. Le composant atomique est constitué d'entités permettant de gérer l'adaptation des blocs système conformément aux modèles de calcul choisis. Enfin, le composant de traduction encapsule des comportements pour la traduction de protocoles permettant l'interconnexion MoCs hétérogènes.

Ainsi, dans cette approche, l'hétérogénéité des systèmes modélisés est supportée d'une part par le découpage structurel hiérarchique et d'autre part par l'application de domaines MoC sur les ports et les connecteurs. La notion de "MoCDomain" permet de décrire les mécanismes supportant la sémantique d'exécution de ces divers éléments. Un domaine MoC est principalement caractérisé par :

- Un schéma comportemental qui définit quel type de comportement caractérise le MoC (machines à états, Ordinary Differential Equation – ODE, etc.),
- Un schéma temporel qui définit le modèle de temps sous jacent du MoC (temps causal, discret, continu, etc.),
- Un schéma de donnée qui définit le type de donnée manipulé par le MoC (type abstrait, vecteur de bits, signal, etc.),
- Un schéma de communication qui définit les mécanismes supportant la communication et la synchronisation.

La figure 4.17 illustre la syntaxe abstraite des domaines de MoC. Elle se focalise en particulier sur les domaines orthogonaux de communication et de comportement. Pour un modèle de calcul particulier, il est nécessaire d'associer une spécification formelle pour tous ses éléments caractéristiques. Cela inclut pour le schéma de communication la spécification des ports (flot ? service ? broadcast ?), des connecteurs (synchronisation ? fifo ? borné ?) ou des adaptateurs. Pour chacun de ces éléments, il est nécessaire d'associer un

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

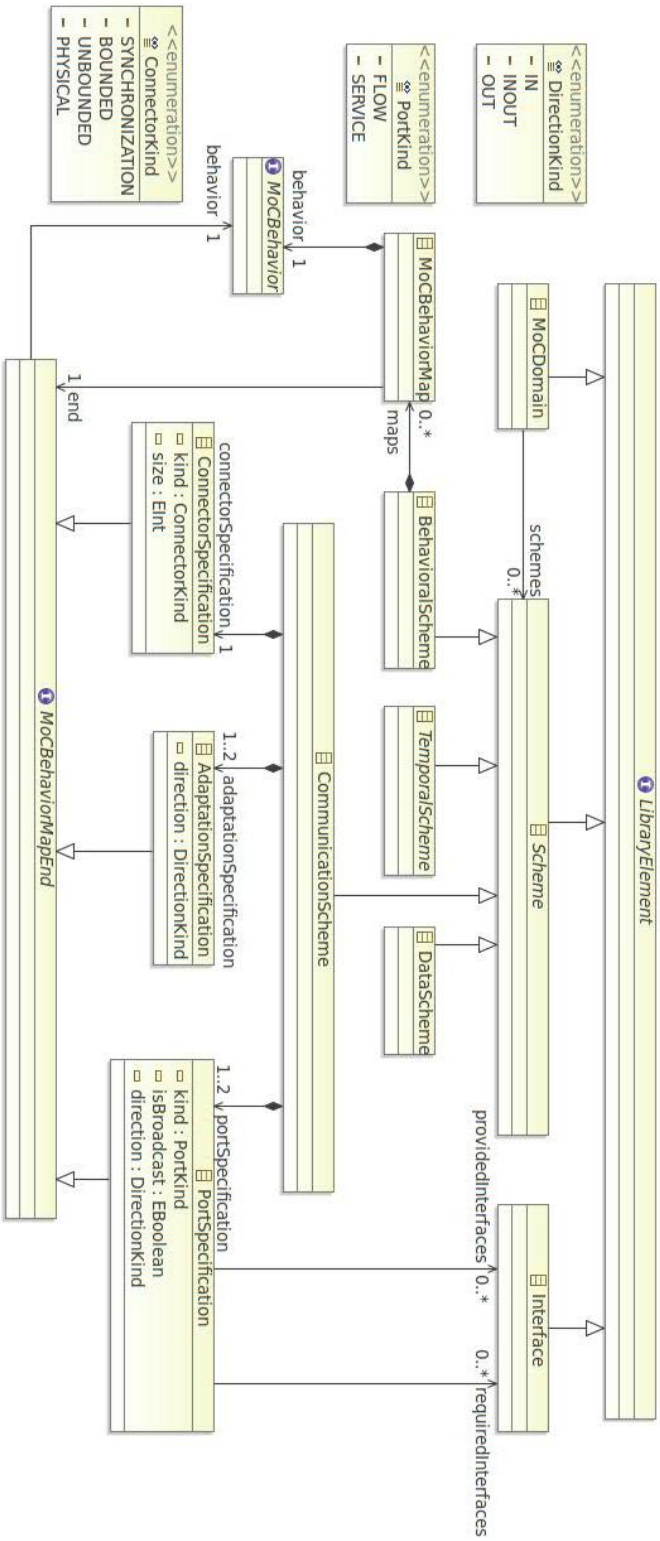


FIGURE 4.17 – Syntaxe abstraite des domaines MoC

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

comportement générique qu'il faudra ensuite spécialiser lors de l'allocation de l'application. La définition de ce comportement fait parti du schéma comportemental et doit respecter les autres schémas orthogonaux. Par exemple, des contraintes OCL permettront de vérifier qu'on ne peut spécifier un comportement continu à partir de machine à états finis.

Ainsi, un domaine MoC appliqué aux ports et aux connecteurs permet non seulement d'attacher une sémantique à ces éléments mais aussi à adapter le comportement applicatif sans le modifier. Bien entendu, l'application d'un domaine doit respecter les différents schémas qui le caractérisent, ce qui peut aussi être vérifié par des contraintes exprimées en OCL. Alors, un composant MoC constitué de ports appliquant différents domaines peut être considéré comme un composant hétérogène appliquant indirectement, à travers ses ports et ses adaptateurs, plusieurs domaines MoC.

L'adaptation du comportement applicatif est rendu possible par l'introduction de deux notions importantes : l'adaptateur ("MoCAdapter") et le gestionnaire d'accès ("MoCManager").

L'adaptateur a la charge de :

- adapter les requêtes MoC ("MoCRequest") en appels systèmes,
- adapter les appels du bloc en système en requêtes MoC,
- encapsuler les résultats d'appels du système en réponses MoC ("MoCResponse"),
- contrôler l'exécution du bloc système,

Pour ce faire, l'adaptateur a recours aux services du gestionnaire d'accès afin de garantir l'intégrité du bloc système en cas d'appels concurrents provenant de plusieurs adaptateurs.

La figure 4.18 fournit une vue complémentaire pour la définition des composants basiques et les composants de traduction. Les composants basiques, réceptacles des blocs systèmes encapsulent des comportements d'adaptation sur chacun de ces ports. Ces comportements ont pour responsabilité d'adapter les communications MoC en communication système, et inversement. L'intégrité du bloc système est assurée par un gestionnaire qui attribut des droit d'accès aux adaptateurs. Les composants de traduction permettent de traduire des protocoles afin d'interfacer différents composants MoC.

Contrairement à l'approche Ptolemy vue dans le chapitre 3, l'hétérogénéité peut être gérée autrement qu'à travers la hiérarchie car il y a une séparation claire entre le comportement applicatif et celui lié à la gestion des communications ou de la synchronisation. Aussi, l'approche proposée possède

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

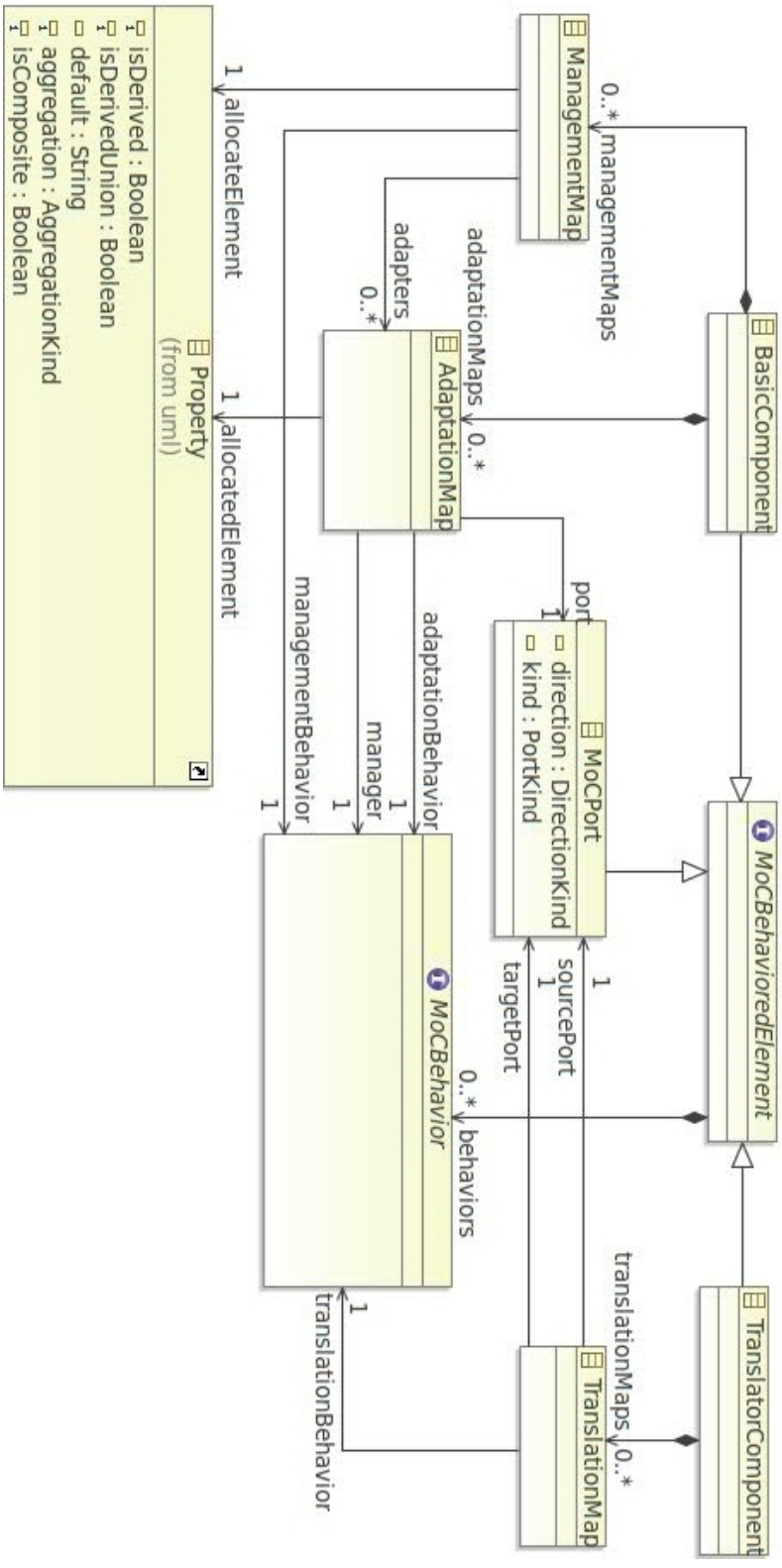


FIGURE 4.18 – Composants d'adaptation et de traduction

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

deux grandes forces :

- elle permet de gérer l'hétérogénéité au sein d'une seule et même entité,
- elle préserve le comportement applicatif et facilite ainsi les activités de vérification.

4.3.3 Capitalisation des MoCs

Nous voyons dans cette section un cas concret d'utilisation du méta-modèle COMETA pour gérer les activités du niveau AML de la méthodologie co-design proposée au chapitre 5. Avant cela, je présente la mise en œuvre de COMETA dans le flot MoPCoM.

Le méta-modèle COMETA a été outillé dans l'environnement Eclipse et expérimenté dans un modèle UML industriel à travers l'utilisation de l'outil de modélisation Rhapsody. Dans cette expérimentation industrielle, l'idée principale était d'utiliser COMETA pour décrire des modèles de calculs existants ou nouveaux, et de générer des librairies de MoCs réutilisables pour l'environnement de modélisation Rhapsody. Le choix de cette cible a été guidé par un besoin industriel de fournir des modèles simulables. Or, l'outil de modélisation Rhapsody offre un tel environnement grâce à son moteur de simulation à événements discrets OXF (Object eXecution Framework).

Aussi, de la même manière que les auteurs de [HSV04] ont utilisé le moteur de simulation à événements discret de SystemC pour implanter plusieurs MoCs, j'ai utilisé COMETA pour générer des librairies de MoCs simulables au dessus d'OXF. Ainsi, à partir du méta-modèle COMETA, j'ai défini un flot d'utilisation schématisé dans la figure 4.19.

En premier lieu, les concepteurs capturent le modèle de calcul en utilisant COMETA. Dans le cadre de la méthodologie proposée, il s'agit de décrire une plateforme d'exécution AML constituée de composants MoC interconnectés (Structure) puis d'attacher un domaine MoC à chacun de ces constituants (Sémantique). À partir de ce modèle, une transformation de type "model-to-model" permet de générer :

- L'ensemble des stéréotypes UML nécessaires,
- L'ensemble des interfaces et des événements utilisés par la simulation,
- L'ensemble des composants destinés à recevoir l'application.

Ça n'est que lorsque que les allocations entre les blocs du système et ceux de la plateforme ont été définis qu'une seconde transformation "model-to-model" va générer la plateforme alloué et tisser une sémantique d'exécution à chacun

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

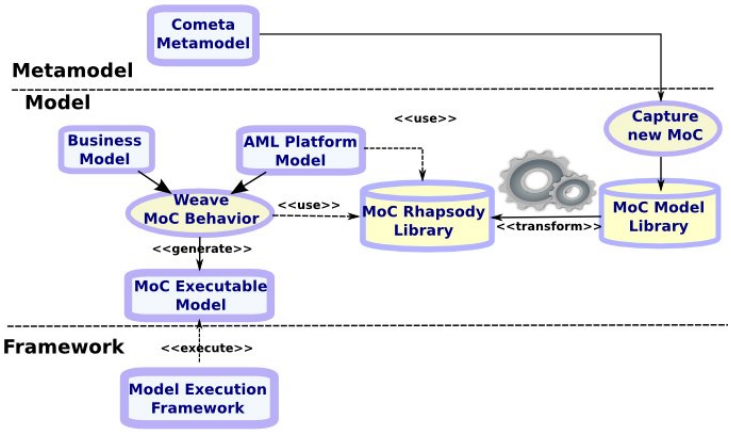


FIGURE 4.19 – Flot pour l'outillage d'un nouveau MoC

des éléments structurels relatif aux MoCs choisis.

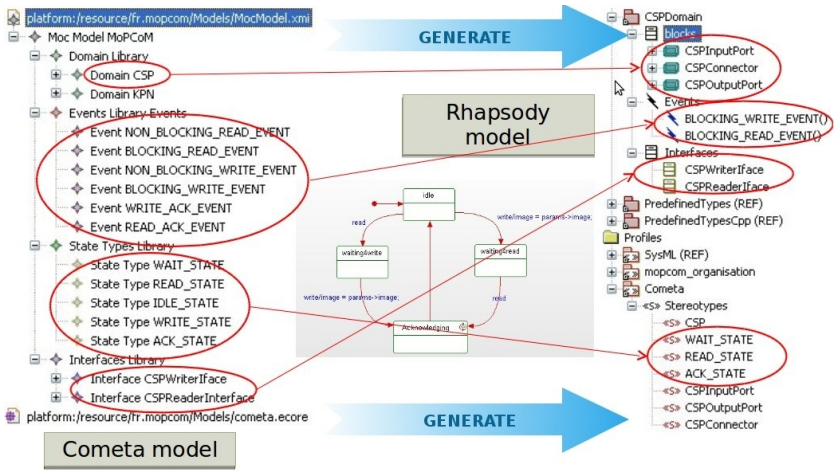


FIGURE 4.20 – Transformation AML

La figure 4.20 illustre la génération des artefacts pour supporter les MoCs KPN et CSP dans l'outil Rhapsody. Une première transformation de modèles COMETA vers Rhapsody permet de générer l'ensemble des stéréotypes et de leurs tags. Enfin, une seconde transformation, après tissage, permet de générer les comportements de la plateforme, et d'adapter des comportements génériques issus des librairies MoC. On peut à ce niveau là citer les travaux de [RCGT09] où les auteurs présentent une approche similaire pour le support

© 2012 Tous droits réservés.

Thèse d'Ali Koudri, Lille 1, 2010

des composants GCM de MARTE dans l'outil Papyrus.

La figure 4.21, extraite du modèle d'application industriel, illustre la spécification d'une allocation entre le modèle métier ① et la plateforme AML spécifiée avec COMETA ②. Cet exemple mélange deux MoCs que sont CSP (Concurrent Sequential Process) et KPN (Kahn Process Network), bien connus de la littérature.

Dans le modèle CSP, chaque processus qui émet une donnée doit interrompre son comportement jusqu'à ce que la donnée produite soit consommée. Aussi, chaque processus en attente d'une donnée bloque son comportement jusqu'à ce que la donnée attendue soit produite. Dans ce modèle, chaque écriture / lecture sur un connecteur est bloquant.

Dans le modèle KPN, chaque processus qui émet une donnée ne se préoccupe jamais de savoir si la donnée à été consommé ou non. À chaque fois qu'une donnée est produite, le processus émetteur continu son comportement nominal. Lorsque qu'un processus requiert une donnée, il bloque son comportement jusqu'à ce que la donnée attendue soit disponible. Dans ce modèle, chaque écriture dans un connecteur est non-bloquante et chaque lecture est bloquante. Contrairement au modèle de calcul SDF (Synchronous Data Flow), les rythmes de production/consommation sont inconnus et les connecteurs possèdent des tailles non-bornées.

Dans la plateforme allouée ③, on retrouve les artefacts définis par la plateforme plus quelques autres qui auront été générées lors de la transformation de modèles pour gérer les communications ou la synchronisation :

- Des connecteurs constitués de files infinies pour les modèles KPN,
- Des connecteurs avec des files de taille 1 pour les modèles CSP,
- Des adaptateurs à chaque port pour transformer les requêtes MoC en appel système et inversement,
- Des gestionnaire d'accès pour chaque composant MoC pour gérer l'accès au bloc système afin d'en garantir l'intégrité.

Nous avions fait ce choix de modèle de calcul avec les ingénieurs systèmes parce qu'il semblait correspondre le mieux au besoin. Par ailleurs, pour la petite anecdote, c'est grâce à l'utilisation de COMETA et aux simulations autorisées par les transformations que nous nous sommes aperçus que ce choix ne convenait pas. En effet, grâce aux simulations, nous avons pu constater qu'il ne pouvait y avoir de détection d'écho dans le modèle de Radio Intelligente présenté dans le chapitre suivant parce que les descripteurs des signaux ne pouvaient que difficilement se trouver dans une même fenêtre temporelle.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

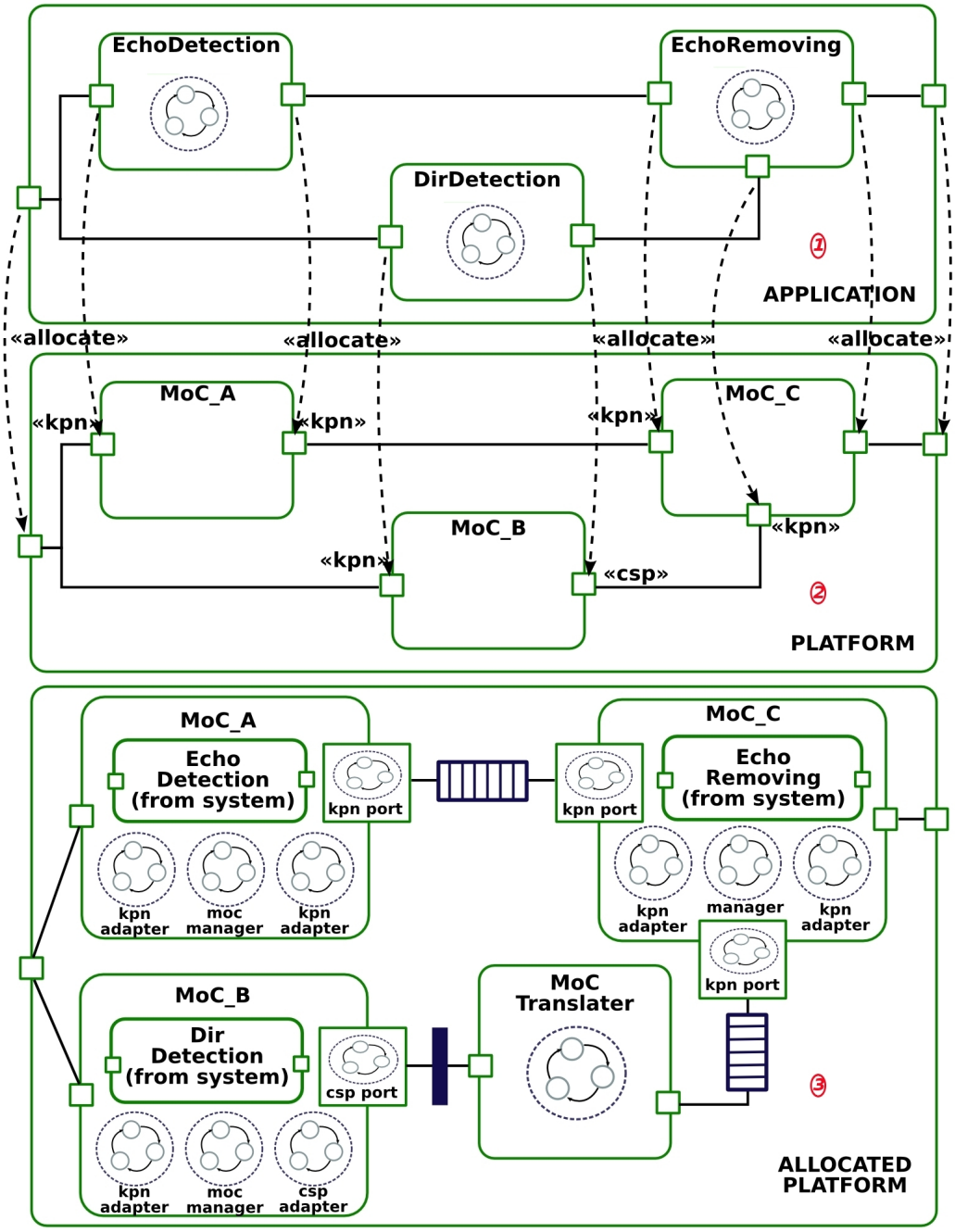


FIGURE 4.21 – Exemple d'allocation sur une plateforme AML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

En effet, conformément au modèle KPN, pendant que le composant est bloqué sur la lecture d'un port, il peut rater les autres données qui arrivent par les autres ports, violant ainsi les contraintes temporelles imposées par l'application. Nous avons ainsi pu corriger ce défaut en proposant un modèle KPN avec timeout qui correspond à un modèle KPN dans lequel la lecture n'est bloquante que pour un temps donné.

4.4 Conclusion

Après avoir constaté les manques des méta-procédés actuels dans la clarification des modèles de processus, j'ai introduit les notions d'intention, de stratégie pour décorréler les aspects intentionnels des processus de leurs réalisations. J'ai ensuite proposé d'aligner la définition des produits sur la notion de modèle. Ces propositions m'ont permis de préciser la définition des activités, des contraintes et des composants de processus.

Dans l'objectif de mieux gérer les activités de l'ESL, j'ai constaté des manques dans le profil MARTE pour la prise en compte de l'hétérogénéité des plateformes. Dans ce cadre, j'ai proposé une extension permettant d'introduire de la flexibilité dans le choix des modèles de calculs afin de faciliter les activités d'allocation et d'analyse.

Dans le chapitre suivant, je propose un modèle de processus de conception conjointe, basé sur l'utilisation des méta-modèles introduits. Ce chapitre présente les niveaux d'abstraction qui le caractérise et leurs motivations. Ce travail est aussi pour moi l'occasion d'évaluer la pertinence des langages MARTE et COMETA pour représenter les artefacts de la conception conjointe. La modélisation du processus me permettra également d'évaluer la pertinence de l'extension MODAL.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Chapitre 5

Processus MoPCoM

Sommaire

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

5.1 Introduction

Conformément aux principes de l'IDM, l'utilisation de modèles permet de mieux séparer les préoccupations et de rendre explicite ce qui était auparavant implicite. Mais comme nous l'avions mentionné dans [ofbr09], l'IDM s'est surtout illustrée dans des expérimentations de développement logiciel et n'a pas pris en compte les préoccupations de l'ESL. Par exemple, l'implantation de référence de l'IDM qui est le MDA [OMG03] ne considère qu'un niveau de plateforme et ne permet pas une bonne exploration d'architecture [Ghe05].

Dans le chapitre précédent, j'ai fait des propositions visant à améliorer la modélisation des processus IDM de codesign. Afin de valider les contributions COMETA et MODAL proposées, je propose une méthodologie IDM de codesign qui applique ces deux propositions. Cette proposition originale représente un volet important de cette thèse et permet de valider la pertinence de mes propositions par l'expérimentation.

La méthodologie MoPCoM peut être vue comme un raffinement du modèle en Y proposé dans l'approche MDA pour prendre en compte les préoccupations de l'ESL comme l'exploration d'architecture ou l'approche "Platform Based Design" présenté dans le chapitre 2. La figure 5.1 donne une vue générale du processus. Le point d'entrée de cette méthodologie est un ensemble d'exigences fonctionnelles / non-fonctionnelles formalisées.

La méthodologie MoPCoM est basée sur la représentation de trois niveaux de modélisation / abstraction représentant différentes préoccupations, respectivement nommées AML, EML et DML :

- Le niveau AML – Abstract Modeling Level – a pour but de décrire une plateforme virtuelle exprimant le niveau de concurrence et de pipeline souhaité ainsi que l'allocation de l'application sur cette plateforme. Le résultat de cette allocation représente un premier niveau d'implémentation permettant d'analyser la correction fonctionnelle au regard des contraintes non fonctionnelles de haut niveau,
- Le niveau EML – Execution Modeling Level – a pour but de décrire la topologie de la plateforme d'exécution à gros grain, c'est à dire en termes de nœuds d'exécution, de communication ou de mémorisation. L'allocation du résultat de l'allocation AML sur cette plateforme permet de déterminer l'adéquation entre les algorithmes et l'architecture physique,
- Le niveau DML – Detailed Modeling Level – a pour but de décrire

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

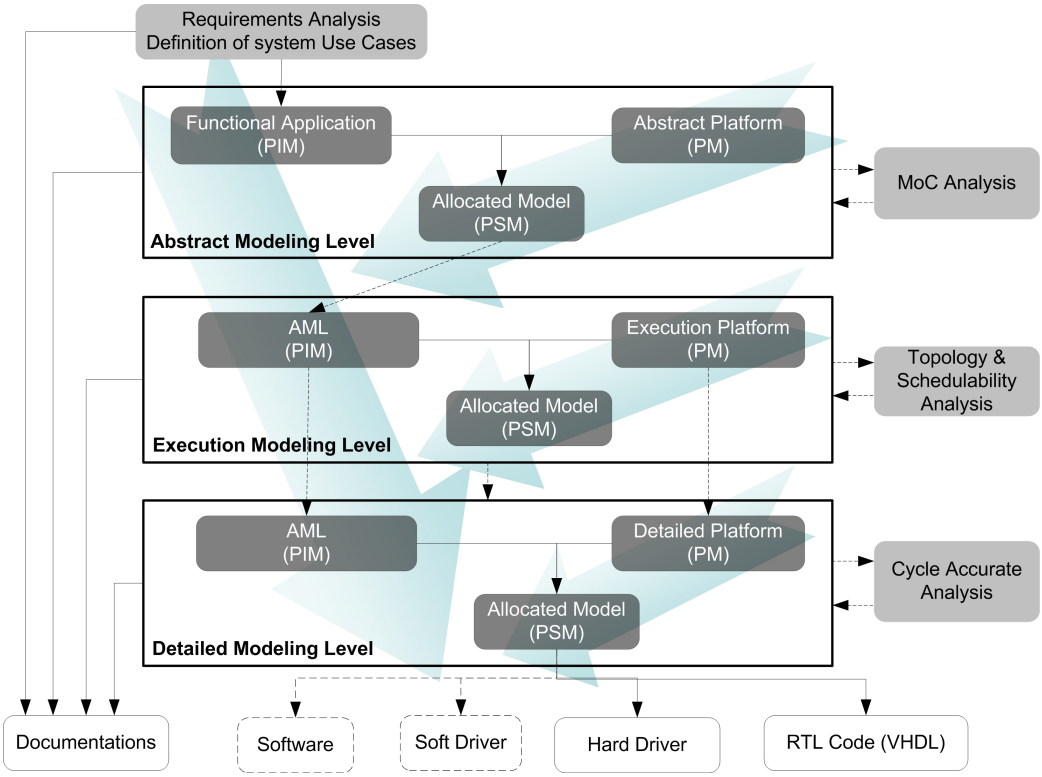


FIGURE 5.1 – Vue générale de la méthodologie proposée

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

une plateforme d'exécution à grain fin qui représente un raffinement du niveau précédent. L'allocation du modèle AML alloué sur cette plateforme permet de mener les analyses du niveau précédent au cycle et au bit près.

En considérant le très grand nombre de concepts présents dans UML ajouté à ceux introduit par les différents profils, les développeurs peuvent être vite perdus dans l'utilisation de ces concepts. Aussi, la définition d'une telle méthodologie de codesign basée sur l'utilisation de modèles nécessite l'identification de l'ensemble des concepts nécessaires et suffisants pour arriver à un bon compromis coût / qualité. Aussi, à travers la proposition d'une méthodologie IDM de codesign, je désirais :

- Capturer les savoirs faires métiers à travers des transformations de modèles,
- Formaliser les règles méthodologiques sous forme de contraintes OCL appliquées au modèle applicatif.

La partie haute de la méthodologie proposée reprend des techniques de conception et d'analyse classiques telles que HARMONY [Dou09] menant à la production d'une spécification fonctionnelle formalisée et exécutable. La méthodologie MoPCoM ajoute en sus la formalisation des plateformes et des contraintes non-fonctionnelles. Dans cette thèse, J'ai essentiellement adressé le niveau AML. Les niveaux EML et DML sont décrites de manière globale et font l'objet de deux thèses à venir. Je n'ai évidemment pas la prétention de résoudre tous les problèmes du codesign à travers la méthodologie proposée mais tout au moins de dégager des gros axes devant être validés par différentes mise en application.

Dans la seconde section de ce chapitre, je présente succinctement l'application de référence qui a été développée au sein de l'entreprise Thalès en guise de démonstrateur. Dans la troisième section, je fournis un résumé de la méthodologie afin de fournir un fil conducteur au lecteur. Les sections suivantes décrivent les différents niveaux de modélisation de la méthodologie proposée. Pour chaque niveau, je décris les intentions des parties prenantes et les stratégies associées. Au delà de l'apparente complexité des figures présentées, il faut bien garder à l'esprit que ces stratégies sont destinées à être automatisées.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

5.1.1 Présentation de l'application de référence

Afin de mieux fixer les idées, j'utilise dans ce chapitre un exemple d'application industrielle de radio intelligente (Cognitive radio). Cet exemple, développé dans le cadre du projet MoPCoM SoC/SoPC [MoP07], traite un sous ensemble des problèmes de l'ESL et me permet d'illustrer les différents niveaux de modélisation.

L'application "Radio Intelligente" est une application caractéristique du type d'application de traitement du signal développé par la société Thalès. Aussi, la modélisation de cette application répond à trois besoins :

- Vérifier la pertinence des extensions MODAL et COMETA proposées,
- Vérifier la pertinence du profil MARTE et cadrer son utilisation,
- Servir d'exemple d'approche IDM pour le déploiement de la méthodologie dans la société Thalès.

Aujourd'hui, les fournisseurs de technologie sans fil offrent un très large éventail d'applications à leurs clients. Ces applications nécessitent des qualités de service de plus en plus en fortes et notamment en termes de transferts de données. Les statistiques démontrent que le spectre électromagnétique dans le transfert des données reste sous-exploité. Par exemple, la figure 5.2 illustre la moyenne des taux d'occupation par bande de six grandes villes dont New-York ou Tokyo en 2005.

Aujourd'hui, la solution la plus sérieusement envisagée pour exploiter au mieux l'utilisation du spectre électromagnétique est la radio intelligente, plus connue sous le terme "Cognitive Radio". La radio intelligente succède à la radio logicielle (SDR – Software Defined Radio). En plus des services de gestion des différents supports de communication par air (protocoles, bandes, etc.), la radio intelligente offre des services de reconfiguration dynamique basés sur la connaissance de l'environnement. De fait, les plateformes FPGA constituent de bons candidats à l'implémentation de radios intelligentes.

Les principes de base de la radio intelligente sont représentés dans la figure 5.3. Le cycle présenté dans cette figure est connue sous le terme "Cognition Cycle" : dans un premier temps, la radio intelligente surveille l'espace électromagnétique, elle collecte des données afin d'établir des statistiques sur l'occupation des bandes de fréquences puis décide de transmettre des données sur les bandes sous-utilisées. Le lecteur pourra trouver une description détaillé de la radio intelligente et de ces principes dans [MJ00].

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

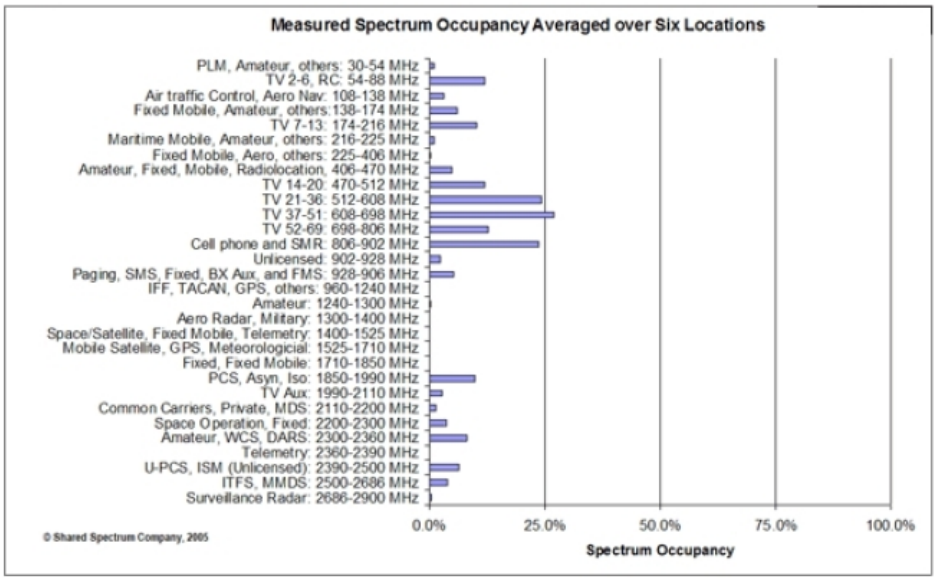


FIGURE 5.2 – Occupation du spectre électromagnétique

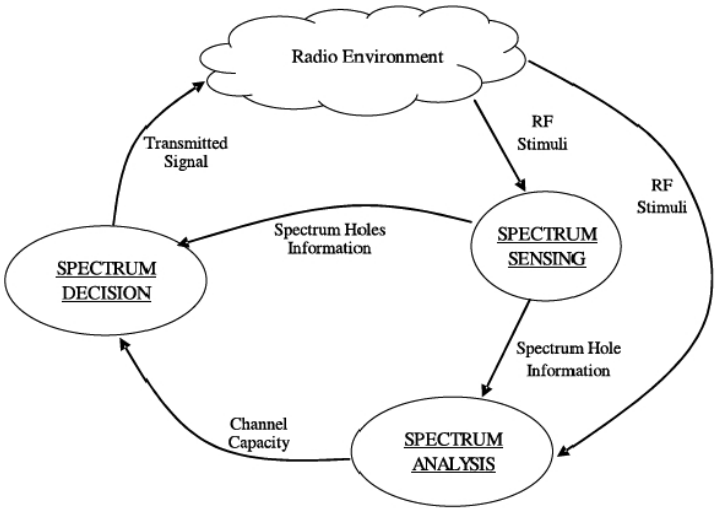


FIGURE 5.3 – Cognition Cycle

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

5.1.2 Résumé de la méthodologie MoPCoM

Cette section a pour but de donner un fil conducteur au lecteur afin de faciliter la lecture des sections suivantes. Pour commencer, l'ingénierie système utilise les formalismes offerts par SysML pour saisir les exigences et établir des relations entre elles. Les cas d'utilisation offrent un bon cadre d'analyse pour établir les frontières du système et identifier ses contrats opérationnels.

Un découpage logique du système permet de regrouper les fonctions et les données dans des classes applicatives pour favoriser l'implantation du système ou la réutilisation dans de futurs projets. À cet effet, les patrons de conception permettent de répondre de manière élégante à ces questions [GHJV95].

La figure 5.4 montre un exemple de raffinement des cas d'utilisation du système extrait du modèle "Cognitive Radio". Il utilise les cas d'utilisation UML formalisant les exigences et les scénarios associés, annotés par des contraintes exprimées dans le langage de spécification VSL fournit par MARTE.

Le concepteur choisit ensuite un type d'implantation qui explicite la concurrence et la communication : c'est le choix du modèle de calcul. Pour ce faire, il utilise les formalismes offerts par l'extension COMETA. L'analyse des MoCs permet de valider par la simulation les choix d'implantation.

Parallèlement, l'ingénierie matérielle conçoit une architecture gros grain de la plateforme d'exécution en utilisant les formalismes offerts par le profil *MARTE* dans le sous-paquetage GRM (Generic Resource Modeling). Cette activité consiste à caractériser les différentes ressources de la plateforme par leurs services associées à leurs qualités de service, leurs interfaces et leurs protocoles de communication de haut niveau (message passing, rendez-vous, fifo, ...).

Il est nécessaire ensuite de fournir un choix d'implantation du système à travers un modèle d'allocation qui définit la distribution spatio-temporelle des fonctions, des données et des communications sur la plateforme. Ce choix d'implantation doit être validé par des scénarios d'analyse mettant en exergue le caractère statistique et probabiliste de la plateforme. Ces scénarios permettent de générer des modèles exécutables à travers des transformations de modèle permettant de générer du code. On peut à ce niveau cibler les outils du langage SystemC car il permet de faire de la co-simulation aux différents niveaux définit par le TLM (Transaction Level Modeling).

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

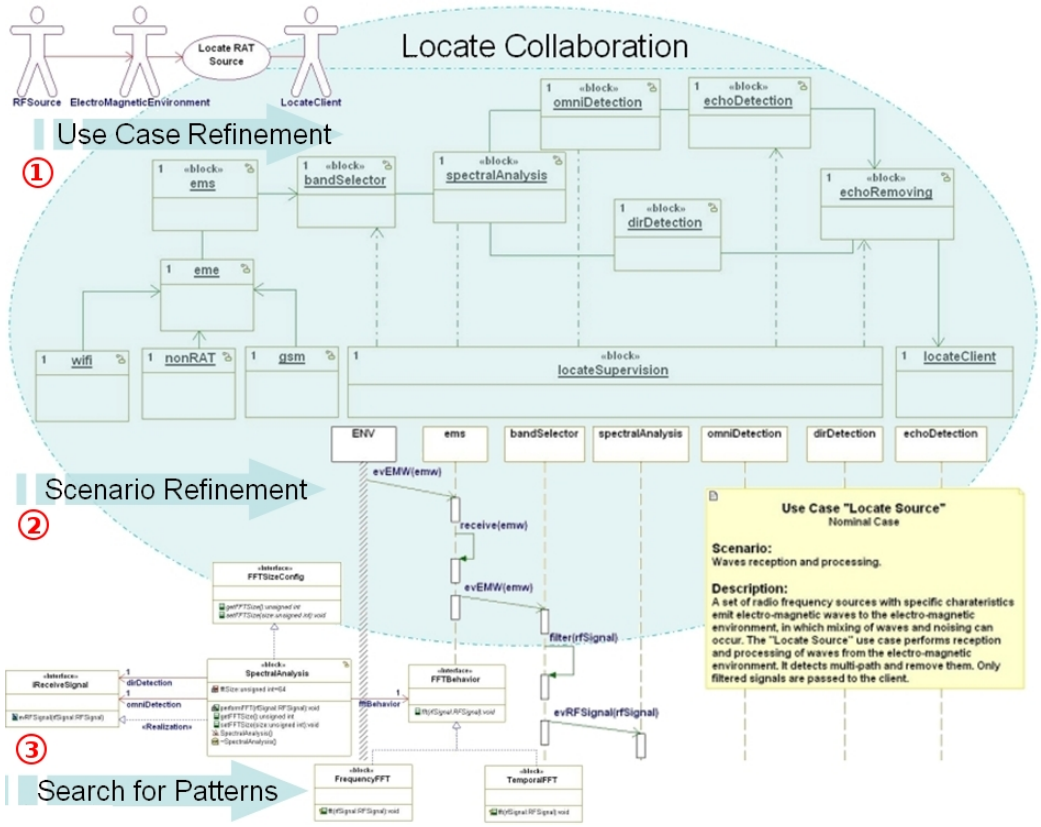


FIGURE 5.4 – Exemple de décomposition fonctionnelle

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

La validation du système au niveau gros grain entraine le raffinement de la plateforme et des analyses. La phase suivante sera donc de raffiner le modèle en identifiant les sous-composants jusqu'à l'obtention d'une hiérarchie de composants détaillée. Le raffinement implique d'une part le raffinement des interfaces et des protocoles de communications, et d'autre part celui des allocations et des scénarios d'analyse. Le but est de permettre l'étude du système au bit près et au cycle près (CABA – Cycle Accurate Bit Accurate). Là encore la validation du système passe par le ciblage d'outils permettant l'exécution de modèles à ce niveau de détails (RTL – Register Transfer Level).

5.2 Le niveau AML

5.2.1 Présentation

Alors que l'analyse fonctionnelle se focalise sur la capture du problème et son amendement par différentes parties prenantes à travers des modèles exécutables, cette phase permet de spécifier des choix d'implantation en termes de concurrence ou de modèle de communication. Sa réalisation passe par la capture d'un modèle de plateforme considérée parfaite dans la mesure où elle ne présume en rien de la limitation des ressources de la véritable plateforme d'exécution.

Ce niveau correspond au besoin d'orthogonaliser les préoccupations mentionné dans le chapitre 4 et aussi de fournir des facilités de conception afin de ne pas avoir à se soucier des problèmes de synchronisation ou de communication. Il s'agit là de spécifier une "topologie" d'entités concurrentes communicantes en point-à-point par des protocoles de haut niveau. Le modèle de plateforme spécifié à ce niveau correspond à la notion de modèle de calcul (MoC – Model of Computation) présenté dans le chapitre 4.

Les notions principales qui ont trait à ce niveau sont les notions de :

- ressource concurrente,
- ressource partagée,
- lien de communication,
- allocation.

Une ressource concurrente est une unité autonome d'exécution chargée de réaliser des calculs séquentiellement. Sa définition est d'ailleurs proche de celle de la notion de processus système. Une ressource concurrente peut être décrite de manière hiérarchique ce qui revient à spécifier un parallélisme

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

interne. Dans ce cas, il peut être utile de spécifier les mécanismes de gestion de cet intra-parallélisme. Une ressource concurrente est capable de traiter des messages provenant d'autres ressources concurrentes, parfois au même instant, par des canaux de communications de natures diverses et variées. Ces canaux servent à la fois au transport des données et comme mécanisme de synchronisation entre les ressources concurrentes.

La modélisation de la plateforme AML consiste à élaborer un système où des processus coopèrent de manière efficace dans la réalisation d'une fonction de plus haut niveau, en interagissant par échange d'informations à travers des canaux abstraits en point à point. Dans le but de simplifier toute hypothèse de réalisation concrète de ce type de modèle, il est nécessaire de considérer que :

- il existe un mécanisme préexistant de coordination des processus, indépendant des vitesses relatives d'exécution de ces processus et indépendant des calculs réalisés,
- l'accès aux données est considéré comme instantané et inconditionnel.

L'allocation des blocs fonctionnels sur la plateforme AML caractérise les choix d'implantation du système à un haut niveau. Ces choix d'implantation doivent alors être validés par des analyses dont le type est conditionné par le modèle de calcul choisi. Par exemple, les réseaux de Pétri sont basés sur la spécification de modèles complètement déterministes et leur cadre formel permet d'évaluer tout risque d'interblocage.

Une des principales difficultés de ce niveau consiste à s'accorder sur le modèle de calcul sous-jacent. Il est important de remarquer que le caractère fondamental de ces modèles continue d'alimenter une recherche fondamentale très active en la matière. Ici, il est nécessaire de dégager des éléments de réflexions qui puissent aboutir à un choix d'un modèle "satisfaisant". Afin de fixer les idées, on peut rappeler quelques MoCs significatifs traditionnels à ce niveau d'abstraction :

- Réseaux de Kahn (KPN) : il s'agit ici de processus qui échangent des données via FIFO infinies. Les écritures sont non-bloquantes alors que les lectures sont bloquantes. Ce MoC est bien adapté aux traitements de données multimédia,
- Synchronous Dataflow (SDF) : la connaissance a priori des différents rythmes de production des données permet la coordination des processus (lecture/écriture) selon un mécanisme calculable de manière statique,

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

- Communicating Sequential Process (CSP) : ici les processus communiquent par passage de message synchrone, ce qui signifie que les processus ne communiquent que par la conjonction de lectures et d'écritures de part et d'autre,
- Réseaux de Petri (PN) : ici l'activation des processus se fait par propagation de jetons dont l'accumulation, en certains points du système, permet le franchissement de barrières de synchronisation,
- Automates d'états finis (FSM) : il s'agit d'un MoC qui se base sur la détermination d'un ensemble d'état possible du système et de fonctions de transition assurant son évolution,
- Modèle réactif synchrone (SR) : il s'agit d'un modèle fondé sur l'exécution instantanée des communications et l'existence d'horloges calculées statiquement.

Certains de ces MoCs sont sensibles à diverses limitations :

- connaissance a priori de rythmes d'exécution (SDF),
- finitude de l'ensemble des états (FSM),
- difficulté à maîtriser la bufferisation et la transformabilité (KPN),
- difficulté à assurer l'hypothèse synchrone (SR) et le raffinement temporel des horloges.

Dans les sections suivantes, nous détaillons, pour chaque niveau de modélisations, les intentions associées ainsi que les stratégies UML/MARTE permettant de les satisfaire. Ces stratégies sont illustrées par des exemples extraits de la modélisation du système "Cognitive Radio".

5.2.2 Intentions et stratégies du niveau AML

La description des intentions du niveau AML sont modélisées dans la figure 5.5. Dans les sections suivantes, nous voyons les stratégies choisies pour satisfaire chacune de ces intentions.

Description de la plateforme

La stratégie que nous adoptons est essentiellement basée sur de l'extension COMETA. La figure 5.6 représente le modèle de cette stratégie. À ce niveau, nous utilisons COMETA afin de modéliser une plateforme virtuelle qui reflète les choix en termes de modèle de calcul. La capture de cette plateforme consiste à interconnecter un ensemble de composants MoC. Aussi, il

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

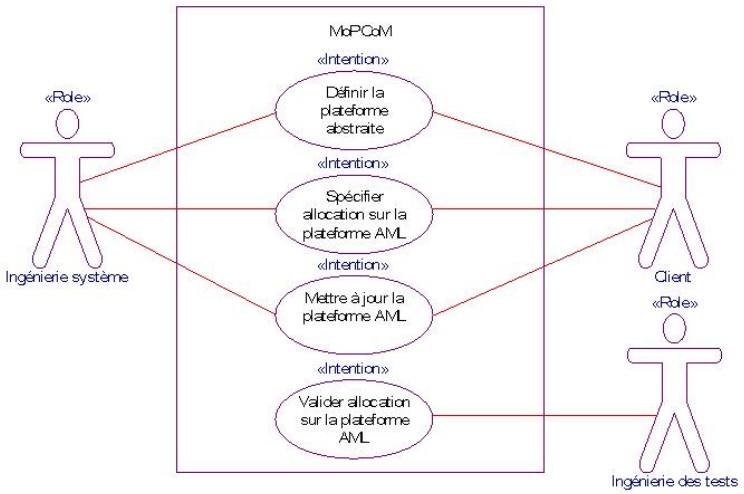


FIGURE 5.5 – Intentions du niveau AML

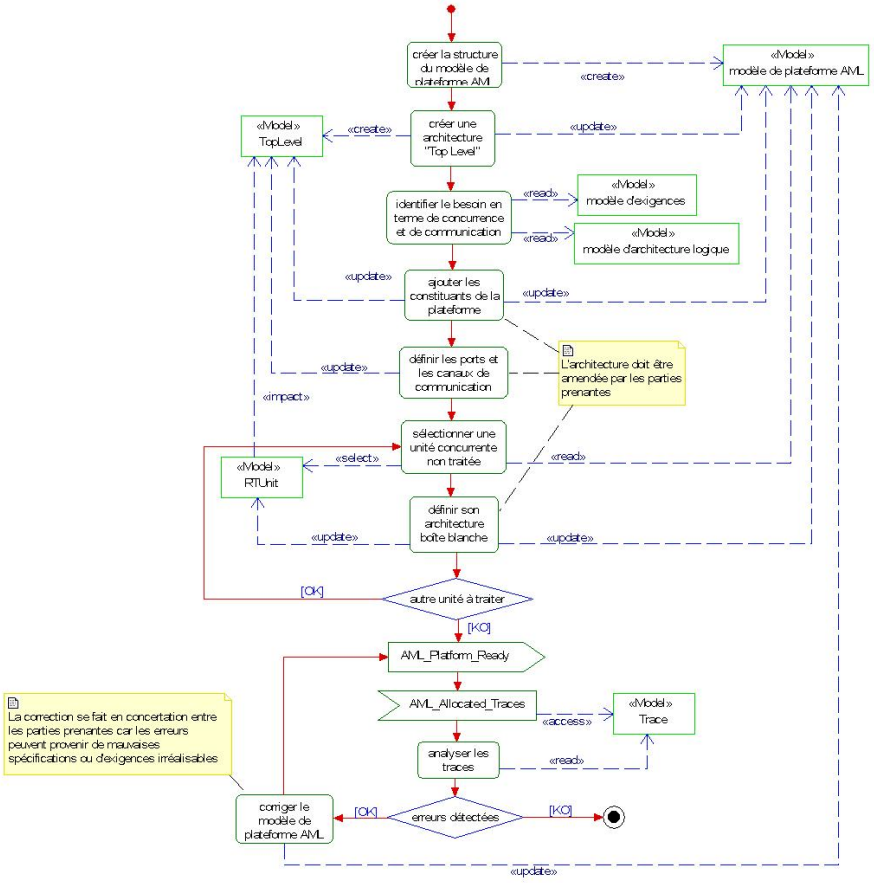


FIGURE 5.6 – Stratégie pour la description de la plateforme AML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

est important à ce niveau de bien spécifier la nature des ports des composants MoCs afin que les transformations de modèles qui suivent les allocations puissent générer les artefacts nécessaires à l'implantation des MoCs choisis. Un exemple de description de plateforme est présenté au chapitre 4.

Description de l'allocation

La stratégie adoptée pour décrire l'allocation (figure 5.7) est basée sur l'utilisation du sous-paquetage "Allocation Modeling" du profil MARTE. L'allocation décrit les choix d'implantation du modèle de calcul sous-jacent. Ce choix est modélisé par un lien de dépendance entre un bloc applicatif (source) et la ressource chargée de l'exécuter (cible).

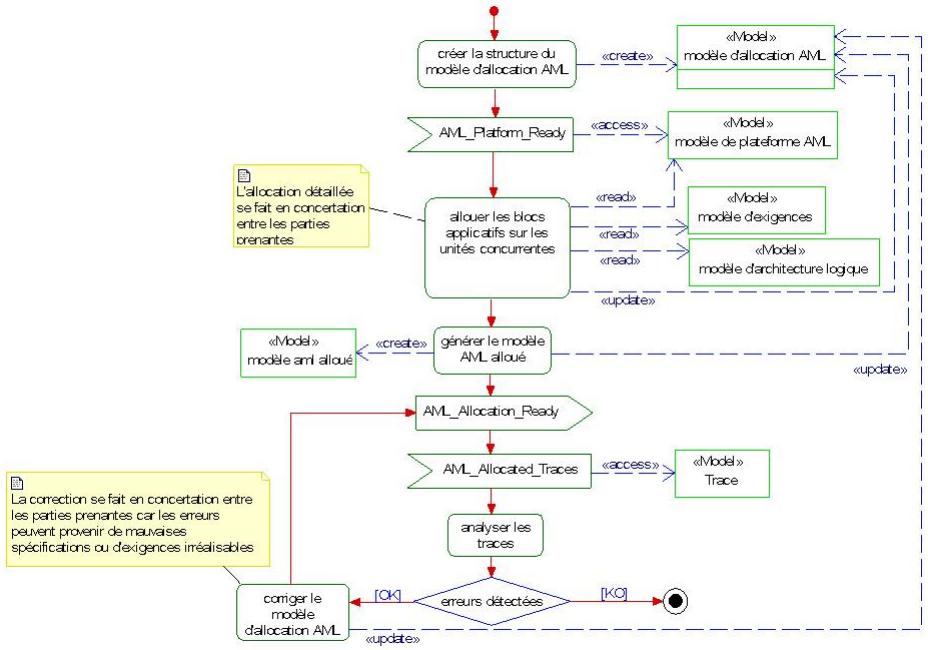


FIGURE 5.7 – Stratégie pour l'allocation AML

Les choix d'allocations doivent permettre d'établir si les frontières qui délimitent les processus sont réellement significatives et naturelles du point de vue de l'application. De manière opérationnelle, peu d'éléments quantitatifs sont à disposition dans cette estimation. On pourra néanmoins se reposer sur l'observation des événements d'activation apparaissant aux interfaces de ces processus, et sur leurs occurrences relatives. À partir de là, il paraît

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

naturel d'aboutir à un système correctement équilibré faisant apparaître des occurrences d'événements équitablement répartis. Cette analyse doit servir à la fois à raffiner le système, mais également à donner des indicateurs lors des transformations vers les plateformes sous-jacentes. Un exemple d'allocation est présenté au chapitre 4.

Validation de l'allocation

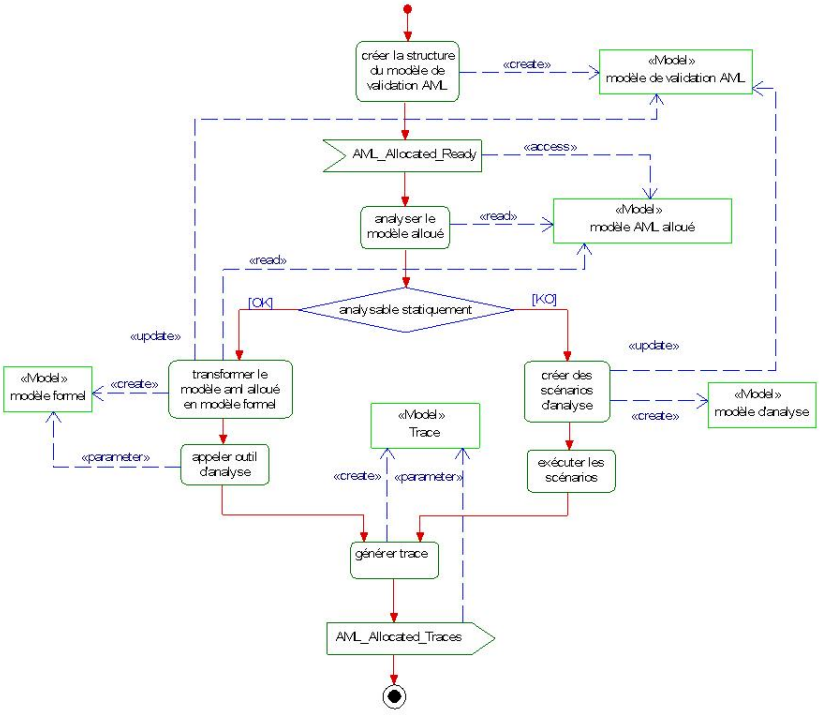


FIGURE 5.8 – Stratégie de la validation de l'allocation

La stratégie adoptée pour valider les allocations est basée sur l'utilisation du sous-paquetage "GQAM" (Generic Quantitative Analysis Modeling) qui permet de spécifier des scénarios d'analyse (figure 5.8). Les analyses se focalisent sur des propriétés particulières du système comme par exemple l'ordonnancement ou les performances.

La validation au niveau AML consiste à annoter le modèle d'allocation du modèle d'architecture fonctionnelle sur le modèle de plateforme MoC. Ces annotations concernent principalement les aspects temps réel du système (deadlines, jitter, etc). Les annotations du modèle sont exploitées afin

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

de générer des vecteurs de test pertinents. L'élaboration des scénarios passe par la représentation de différents contextes du système à travers la génération d'acteurs possédant les comportements en adéquation avec les analyses devant être menées (deadlock, wcet, etc.).

Les évaluations de ce niveau portent sur l'analyse de la communication entre composants logiques encapsulant les composants applicatifs. Nous chercherons ici à évaluer l'impact du parallélisme et du pipeline choisi. Il est important de rappeler, qu'à ce niveau, les qualités du modèle attendu ont trait à la lisibilité, le caractère rapidement modifiable de la capture, ainsi que son caractère "transformable".

Les figures 5.9 et 5.10 montrent les traces de simulation de l'allocation d'une même application sur deux plateformes différentes. La première figure montre les traces d'exécution de deux blocs applicatifs sur des composants MoC de type KPN. On peut voir sur cette figure que les blocs générés par transformation respectent le protocole de haut niveau imposé par ce modèle de calcul, et ce sans modifier le comportement applicatif : le bloc "Application 2" se bloque jusqu'à la production d'une donnée (lecture bloquante) ①. Lorsque le bloc "Application 1" produit une donnée ②, elle l'envoie par le connecteur et reprend son activité (écriture non-bloquante) ③. La disponibilité d'une nouvelle donnée dans le connecteur débloque ainsi le bloc "Application 1" ④.

De même, sur la figure 5.10, on peut constater que les composants MoCs générés réagissent conformément au protocole de haut niveau imposé par ce modèle CSP, également sans modifier le comportement applicatif : le bloc "Application 2" se bloque jusqu'à ce qu'une donnée soit disponible (lecture bloquante) ①. Lorsque le bloc "Application 1" produit une donnée ②, il l'envoie par le connecteur et se bloque jusqu'à ce que celle ait été consommée (écriture bloquante). La production d'une nouvelle donnée débloque le bloc "Application2" qui la consomme ③, débloquant ainsi le bloc "Application1".

Lorsque les choix d'implantation du MoC ont été validés et amendés par les parties prenantes, on peut commencer à intégrer les préoccupations de l'implantation liées aux choix de la plateforme physique ou logique.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

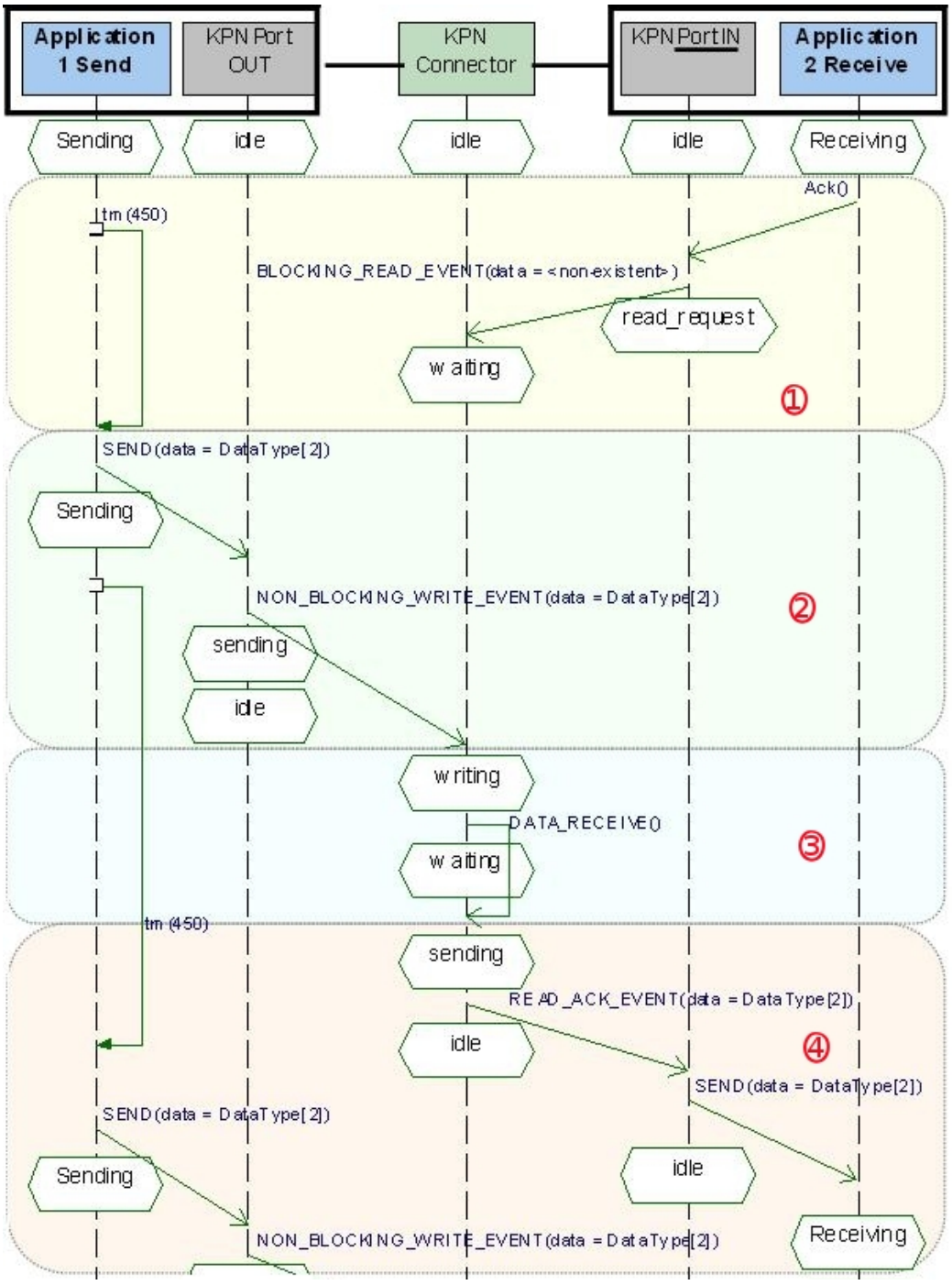


FIGURE 5.9 – Simulation KPN avec Rhapsody

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

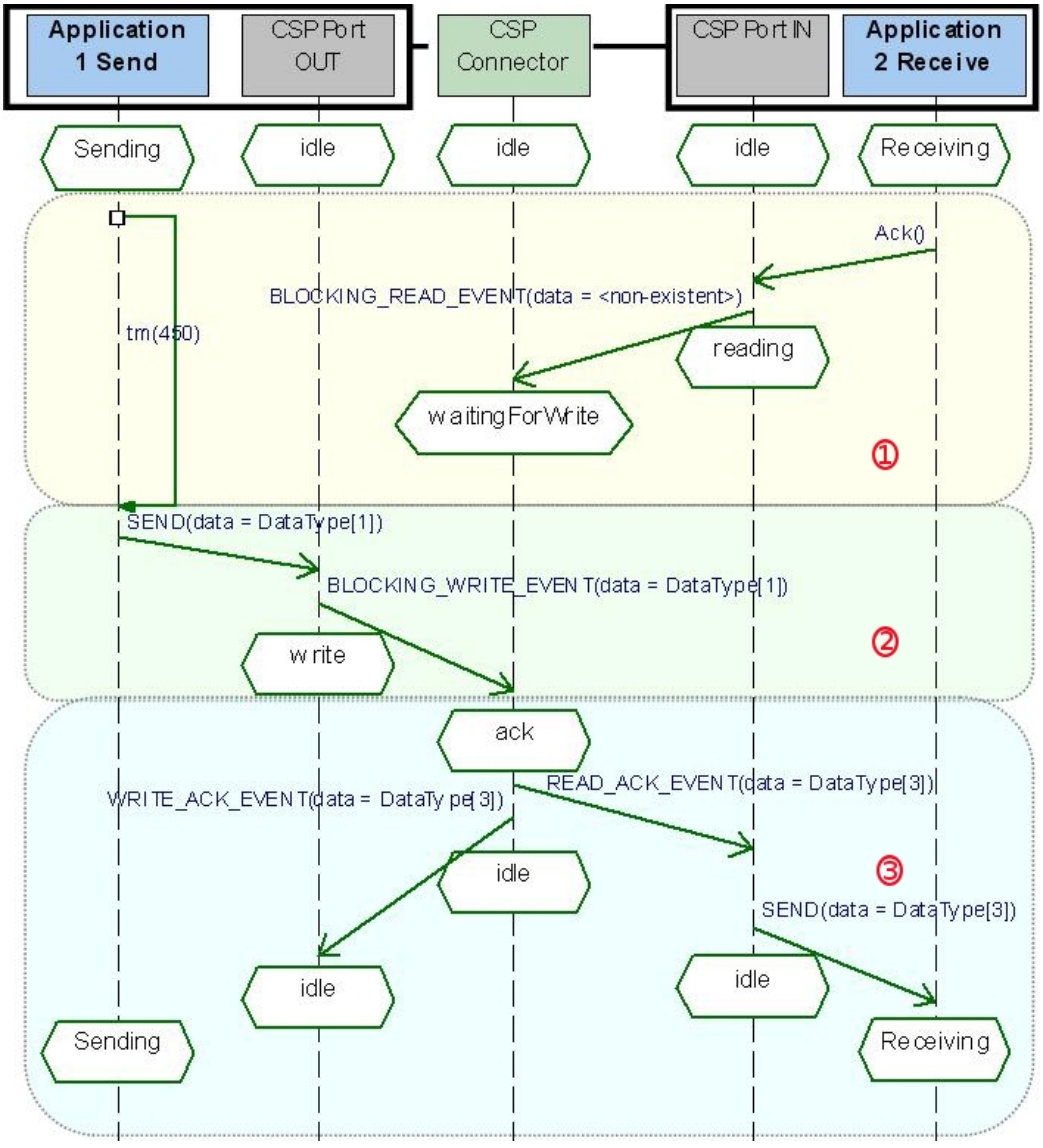


FIGURE 5.10 – Simulation CSP avec Rhapsody

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

5.3 Le niveau EML

5.3.1 Présentation

Le principe de ce niveau est de définir une plateforme servant de support d'exécution au modèle alloué AML. Les choix d'implantation sont matérialisés par des allocations des blocs applicatifs sur les ressources de la plateforme. Les analyses menées à ce niveau abstraient les coûts liés au matériel (CPU, FPGA) ou aux couches d'abstraction (OS, Middleware) à travers des modèles génériques. Les définitions de la plateforme et de l'allocation sont réalisées au regard des compromis entre les performances et les coûts, et à travers un processus itératif mettant en avant la topologie de la plateforme.

La topologie de la plateforme regroupe les nœuds de calcul, de communication ou de mémorisation, interconnectés à travers des bus qui implantent des protocoles de haut niveau (TLM). Les données transitées sont exprimées au bit près et le modèle de temps est approximatif. Les ressources modélisées au niveau EML sont avant tout caractérisées par les services qu'elles offrent aux applications et les qualités de services associées. À ce niveau, les ressources modélisées possèdent des vitesses d'exécution relatives. Les contraintes temps réel sont vérifiées grâce à l'utilisation d'une horloge idéale unique (pattern singleton) mesurant la progression du temps réel et qui est fournit par la librairie de modèles MARTE.

Les résultats des analyses conséquents aux allocations peuvent entrainer des refactorisations au niveau des modèles de plateforme comme au niveau du PIM. Actuellement, ce processus de refactorisations est réalisé manuellement et dépend essentiellement des compétences des ingénieurs qui en ont la charge. Cependant, il est possible de prévoir par la suite des mécanismes de rétro-annotation permettant d'automatiser l'exploration d'architecture.

5.3.2 Intentions et stratégies du niveau EML

Les intentions du niveau EML sont décrites dans le diagramme d'intentions de la figure 5.11. Nous voyons dans les sections suivantes les stratégies adoptées pour satisfaire chacune de ces intentions.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

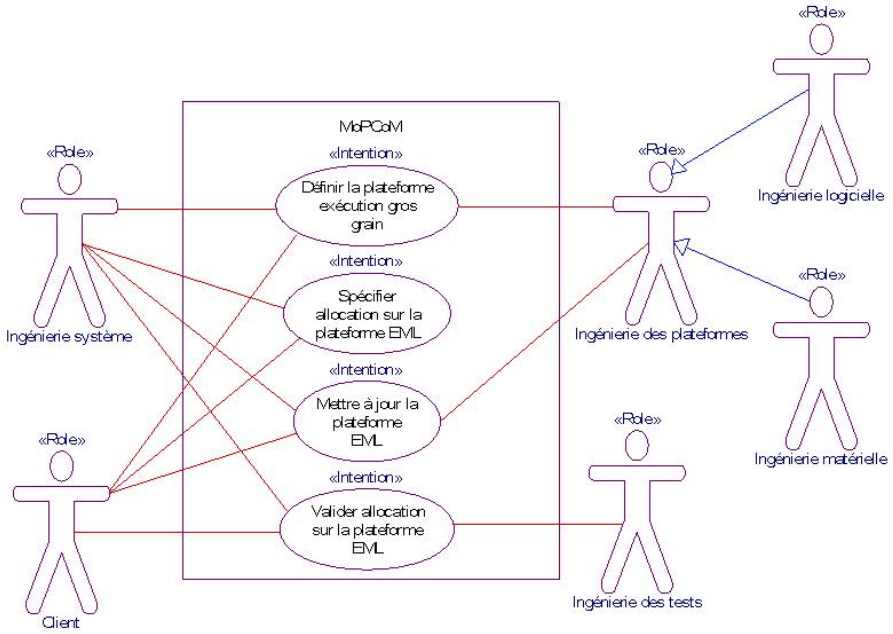


FIGURE 5.11 – Intentions du niveau EML

Description de la plateforme

La stratégie adoptée pour la définition de la plateforme EML est principalement basée sur l'utilisation du sous-paquetage "GRM" (Generic Resource Modeling) du profil MARTE. Ce paquetage offre les concepts nécessaires et suffisant pour modéliser des plateformes à gros grain, sans présumer de leur future nature matérielle ou logicielle. Parmi les stéréotypes proposés dans ce paquetage pour modéliser des ressources, on trouvera entre autres les stéréotypes :

- "ComputingResource" pour la modélisation des ressources de calcul,
- "StorageResource" pour la modélisation des mémoires avec leur taille, leur fréquence de rafraichissement ou leur fréquence d'exécution,
- "TimingResource" pour la modélisation d'horloges logiques ou chronométriques,
- "DeviceResource" pour la modélisation des entités externes au système, identifiées par leurs interfaces.

En premier lieu, les ingénieries logicielles et matérielles extraient des exigences non fonctionnelles les besoins en termes de plateforme d'exécution. Les

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

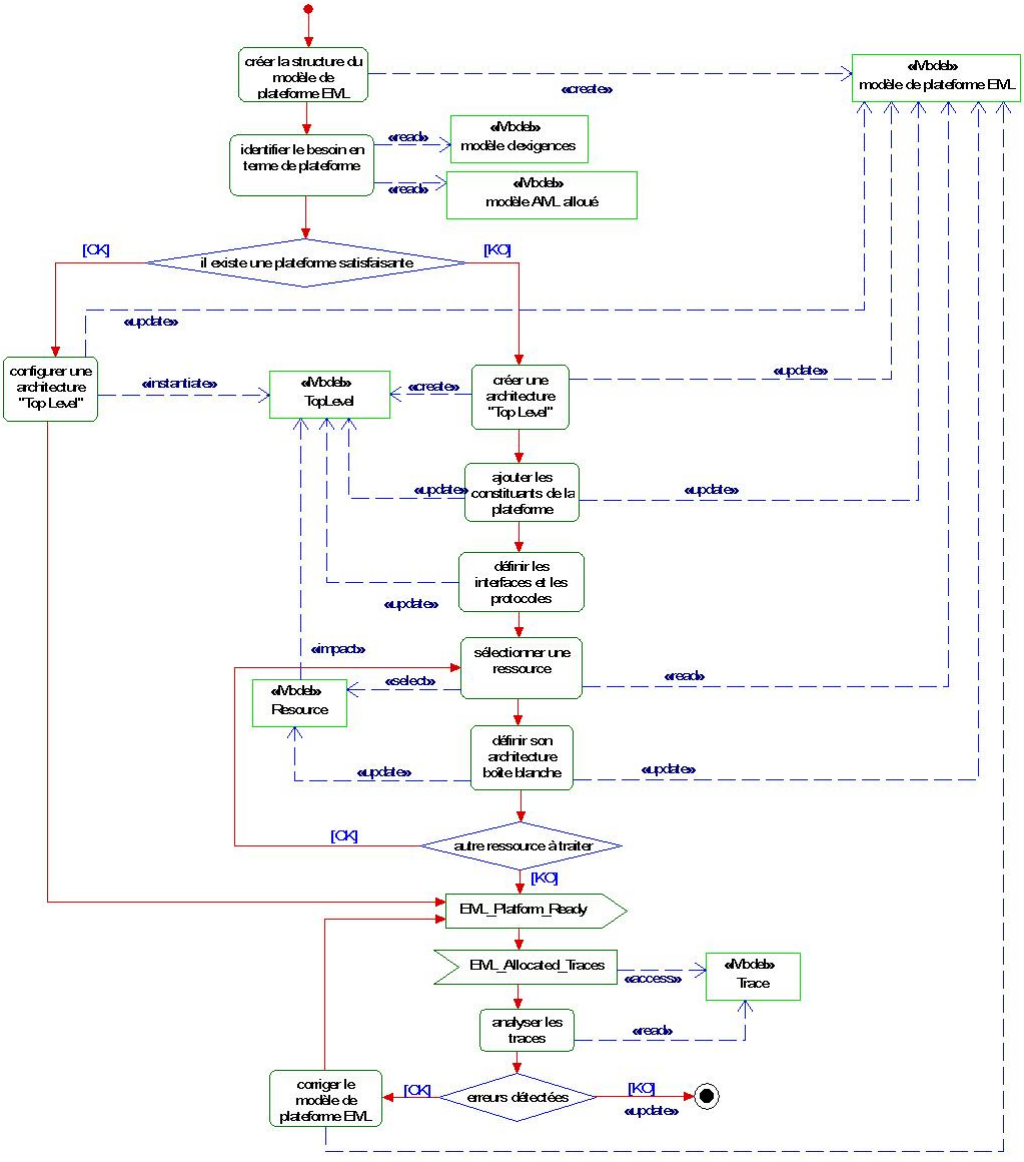


FIGURE 5.12 – Stratégie pour la description de la plateforme EML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

caractéristiques déterminantes de la plateforme d'exécution sont les performances, la taille des mémoires, le poids, les dimensions ou le coût. Comme je l'ai mentionné précédemment, les choix d'allocation sont nombreux et la validation de ces choix au bit près et au cycle près peuvent prendre énormément de temps. Il est alors nécessaire de représenter les plateformes à différents niveaux d'abstractions afin de pouvoir éliminer les solutions non appropriées en termes de compromis coûts / performance / délai.

À ce niveau, ces ressources sont davantage considérées par les services qu'ils rendent et les qualités de service associées. On peut donner comme exemples le service "compute" d'une ressource de calcul ou le service "store" d'une ressource de mémorisation. Les communications sont supportées par des ressources de type bus associées à des protocoles haut niveau. La notion de communication peut revêtir plusieurs formes. À ce niveau, elle est matérialisée sous la forme de transactions.

Actuellement, la modélisation de la plateforme d'exécution s'effectue principalement sur la base de l'aptitude des ingénieurs à sélectionner la solution la plus adaptée à la résolution du problème. Même si l'exécution du modèle EML donne des éléments de réponse significatifs, elle ne suffit pas à démontrer la faisabilité du projet. En effet, la plateforme d'exécution possède certaines caractéristiques (largeur de bus, horloges, latence, débit, nombre de CLB, etc.) dont il faudra tenir compte dans les scénarios d'analyse. La prise en compte de ces caractéristiques passe par un raffinement que nous verrons dans la section suivante.

La figure 5.13, extrait de l'application, illustre un exemple d'allocation de ce niveau. Cette figure montre un ensemble de blocs du modèle issu des analyses AML sur une plateforme EML, constituée de ressources de calcul et de communication.

Description de l'allocation

La stratégie adoptée pour définir les choix d'implémentation est basée, comme pour l'allocation du niveau AML, sur l'utilisation du sous-paquetage "Allocation Modeling" du profil MARTE. Elle est réalisée de manière conjointe entre les ingénieries système, matérielle et logicielle.

L'allocation permet de générer une plateforme allouée par une transformation de modèle qui sera validée par les différents types d'analyses qui seront menées par l'ingénierie des tests. Aussi, l'allocation poursuit un pro-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

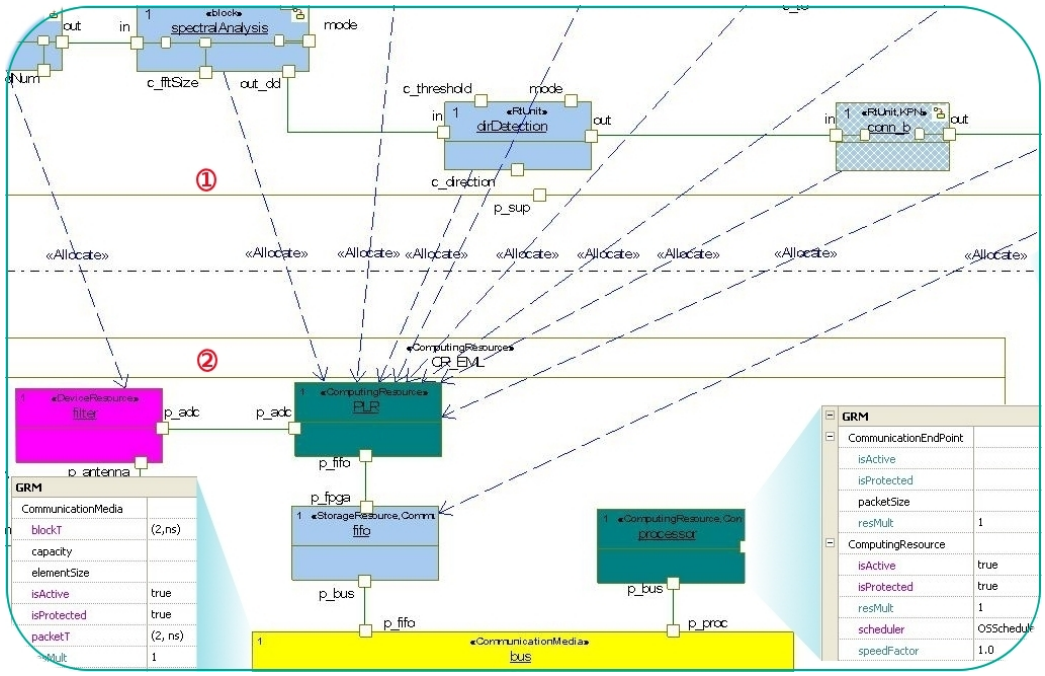


FIGURE 5.13 – Exemple description de la plateforme EML

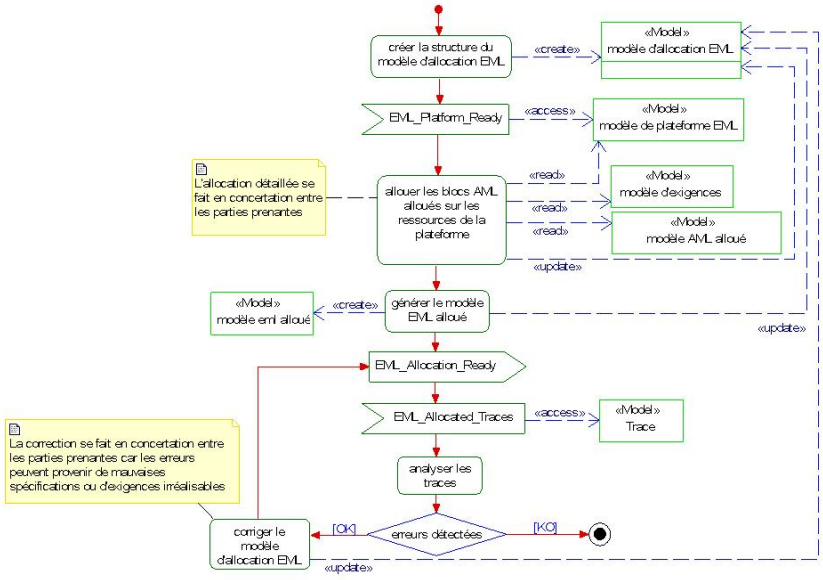


FIGURE 5.14 – Stratégie pour la description de l'allocation EML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

cessus itératif de raffinement menant une implantation validée et amendée par toutes les parties prenantes.

Validation de l'allocation

Comme pour la validation du niveau AML, la stratégie choisie est basée sur l'utilisation du sous-paquetage "GQAM" (Generic Quantitative Analysis Modeling) de MARTE. La différence ici est l'utilisation des sous-paquetages "SAM" (Schedulability Analysis Modeling) et "PAM" (Performance Analysis Modeling) qui a ici un sens dans la mesure où on dispose d'informations pertinentes sur la plateforme d'exécution.

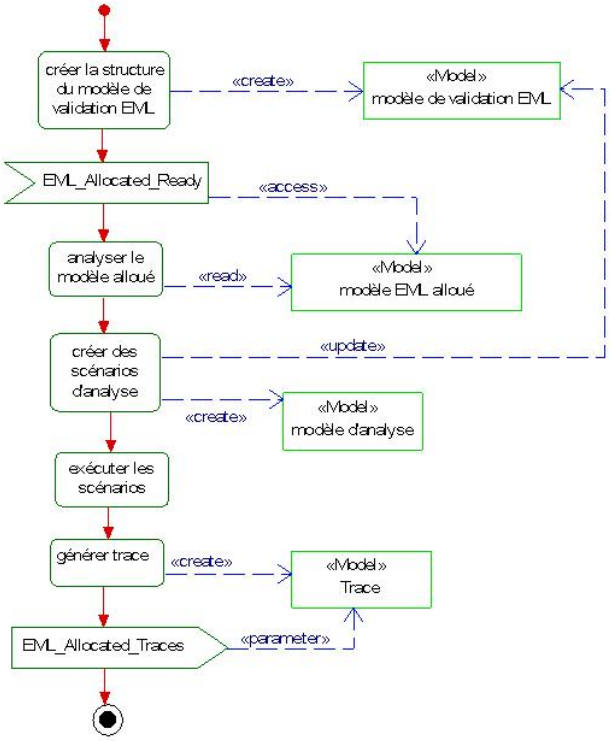


FIGURE 5.15 – Stratégie pour la validation EML

Les caractéristiques déterminantes permettant de valider que les performances requises peuvent être atteintes sont :

- la taille des bus et leur débit,
- les vitesses relatives des ressources de calcul,
- la taille des mémoires.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Les analyses des choix d'implantation doivent mettre en avant ces caractéristiques. Aussi, les scénarios de tests doivent prendre en compte la caractère imprédictible de l'environnement dans lequel le système sera immergé. Dans ce cas, l'utilisation de modèles statistique ou probabilistes est nécessaire pour rendre les simulations plus réalistes et plus pertinentes.

La validation du choix d'allocation EML repose sur des annotations du modèle d'allocation avec des informations portant sur les aspects temps réel du système (exemple : contraintes d'ordonnancement). Les modèles annotés servent de point d'entrée à l'élaboration de scénarios de test. La figure 5.16 illustre un exemple de scénario annoté basé sur l'utilisation du paquetage GQAM du profil MARTE. Dans cet exemple, le but des analyses est de déterminer si le bus supporte bien les charges.

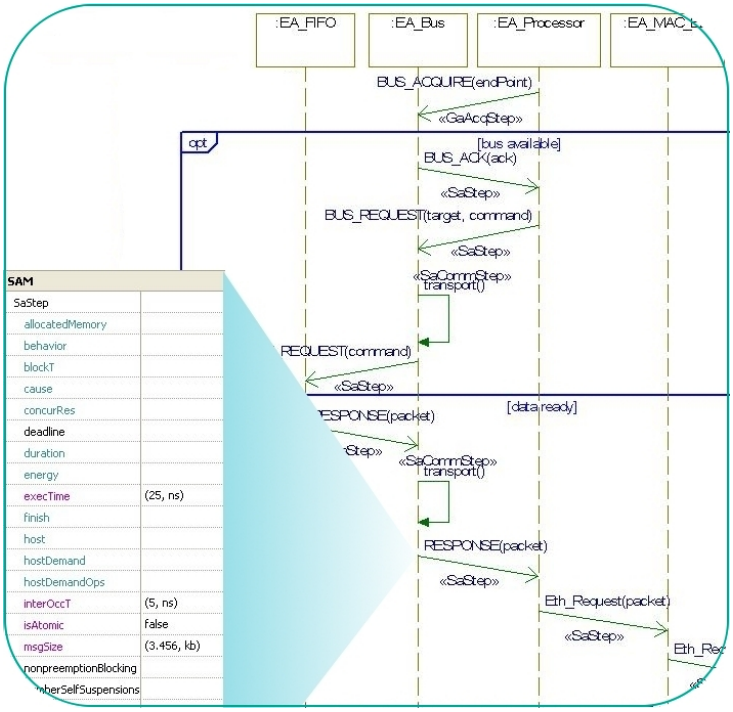


FIGURE 5.16 – Exemple de scénario d'analyse basé sur l'utilisation du paquetage GQAM

La simulation / exécution du modèle alloué produit des traces dont l'exploitation permettra de vérifier que les contraintes temps réel imposées par l'application sont bien respectées et de valider la topologie de la plateforme.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Dans ce cadre, la librairie TLM2.0 de SystemC offre les concepts nécessaires à la modélisation des canaux de communication de ce niveau de modélisation.

5.4 Le niveau DML

5.4.1 Présentation

Ce niveau vise à fournir un modèle détaillé de la plateforme par un raffinement des modèles du niveau précédent. Le modèle alloué de ce niveau doit contenir assez d'information pour permettre la génération du code d'implantation pour les parties matérielles (VHDL) ou logicielles (C), ainsi que la glue logique (pilotes) permettant l'interfaçage des deux parties. Par exemple, les nœuds de calcul définis au niveau EML sont raffinés en processeurs ou en FPGA, les nœuds de mémorisation sont raffinés en SRAM ou en Flash, etc. On s'intéresse à ce niveau autant aux services fournis par les ressources qu'aux caractéristiques physiques précises (fréquence de fonctionnement, poids, prix, etc.). De même, les couches d'abstraction matérielle deviennent des OS ou des middlewares, et sont caractérisées par leurs API. La norme POSIX (http ://-standards.ieee.org/regauth/posix/) constitue un exemple d'API.

Le raffinement de la plateforme implique celui des interfaces, des protocoles de communication ainsi que celui du modèle de temps sous-jacent. La topologie de communication décrite au niveau EML est reprise et raffinée en un modèle de communication par cycle. Les modèles de calculs sont détaillés et définis en termes de protocole tout en gardant la sémantique des MoCs. Les ressources sont cadencées par au moins une horloge et les analyses sont menées au bit près et au cycle près (Cycle Accurate Bit Accurate). Un tel raffinement entraîne le raffinement des analyses menées au niveau EML.

5.4.2 Intentions et stratégies du niveau DML

Les intentions du niveau DML sont modélisées dans la figure 5.17. Dans les sections suivantes, nous voyons les stratégies adoptées pour satisfaire chacune de ces intentions.

Raffinement de la plateforme

La stratégie adoptée pour satisfaire cette intention est principalement basée sur l'utilisation du sous-paquetage “DRM” (Detailed Resource Modeling)

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

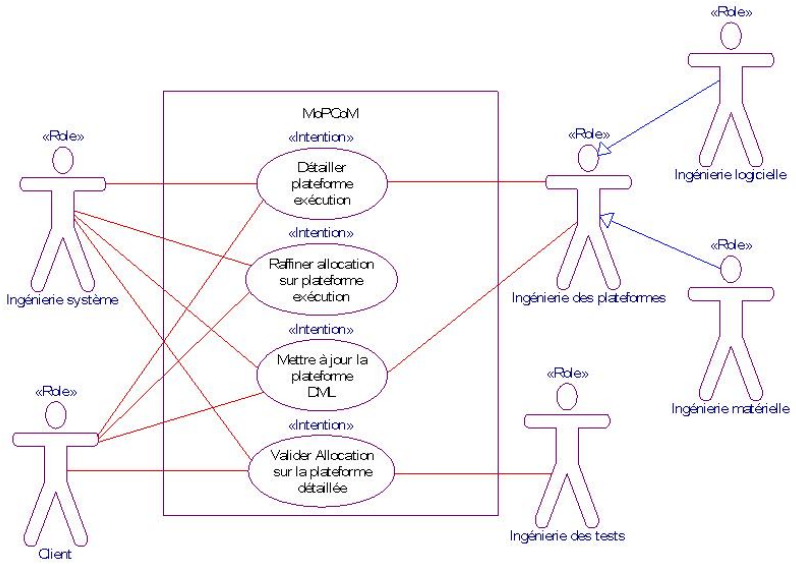


FIGURE 5.17 – Intentions du niveau DML

du profil MARTE. Ce paquetage offre les concepts nécessaires et suffisants pour modéliser des plateformes logiques ou physiques de manière détaillée. Ce paquetage est constitué de deux sous-paquetages qui sont "HRM" (Hardware Resource Modeling) et "SRM" (Software Resource Modeling). Parmi les stéréotypes proposés dans ce paquetage, on retrouve entre autres les stéréotypes de :

- "HW_Processor" représentant des familles de processeurs ayant des caractéristiques communes en termes de fréquence d'exécution ou de nombre d'unité arithmétique et logique,
- "HW_PLD" représentant les composants reprogrammables matériellement ; les PLD sont caractérisés par leur technologie (Antifuse, SRAM, etc.) ou leur nombre de flip-flop,
- "HW_RAM" représentant les mémoires de calculs,
- "HW_BUS" représentant les bus de communications caractérisés par leur largeur d'adresses et de données, ou leur politique d'arbitrage.

Après avoir validé le modèle EML à travers les différents types d'analyse (ordonnançabilité, performance, etc.), les composants matériels sont réévalués pour être différentiés en constituants matériels ou logiciels. En particulier, la description de l'application doit préserver les propriétés des niveaux supérieurs (AML, EML) et un plus haut degré de précision est cherché dans

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

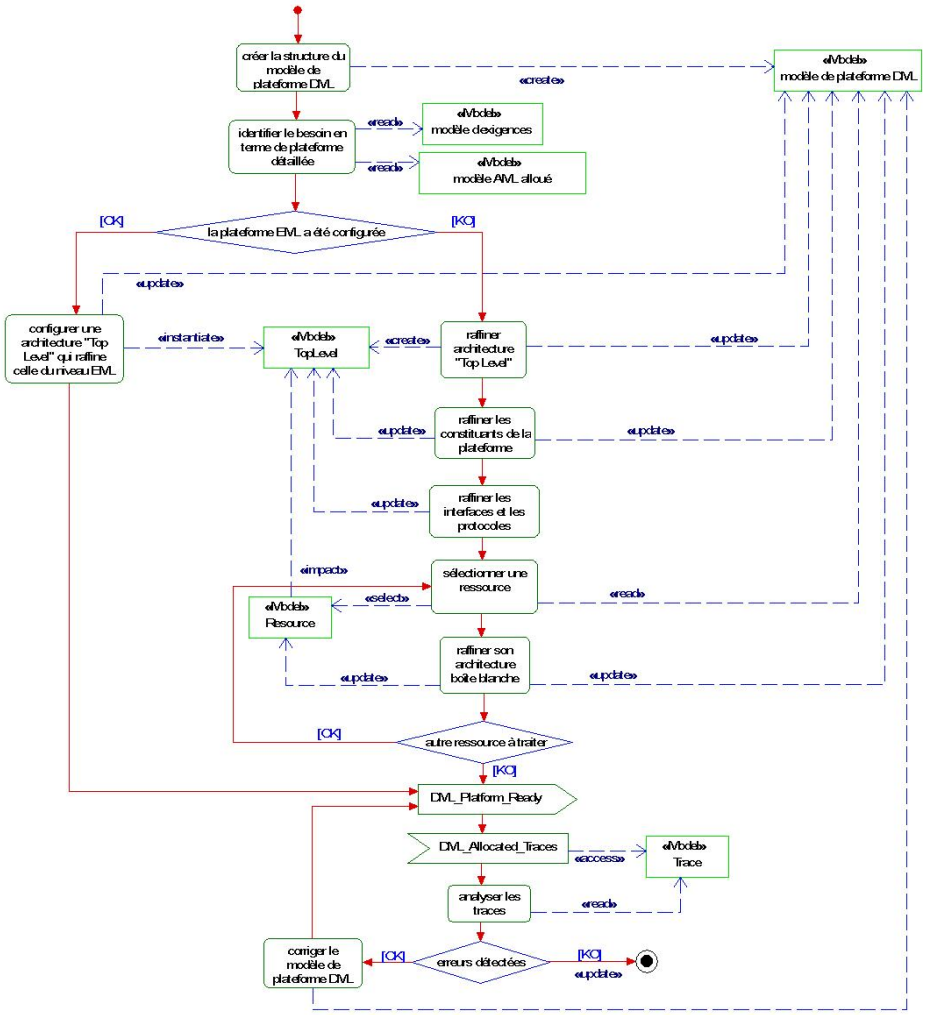


FIGURE 5.18 – Stratégie pour le raffinement de la plateforme

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

le but de raffiner les données et les comportements du système.

Par exemple, les données d'entrée qui étaient immédiatement disponibles et complètes dans l'interface de niveau AML vont arriver de façon répartie dans le temps, de nouveaux échanges se créent pour véhiculer du contrôle et des données, les évènements uniques deviennent multiples. Le cœur fonctionnel des composants va donc en "subir" les conséquences puisqu'il sera bloqué par l'arrivée retardée des données d'entrée : on voit donc apparaître à ce niveau une spécialisation des données dans le temps et l'espace dans la mesure où elles seront localisées (non disponibles immédiatement). Une fois la plateforme raffinée, l'allocation doit permettre un certain nombre de vérifications pour valider le modèle (par exemple la cohérence des types de ports).

Raffinement de l'allocation

La stratégie adoptée pour raffiner l'allocation est, comme pour la description de l'allocation du niveau EML, basée sur l'utilisation du sous-paquetage "Allocation Modeling". Il s'agit de prendre en compte l'ajout des détails de la plateforme mentionnée dans la section précédente et de raffiner les analyses.

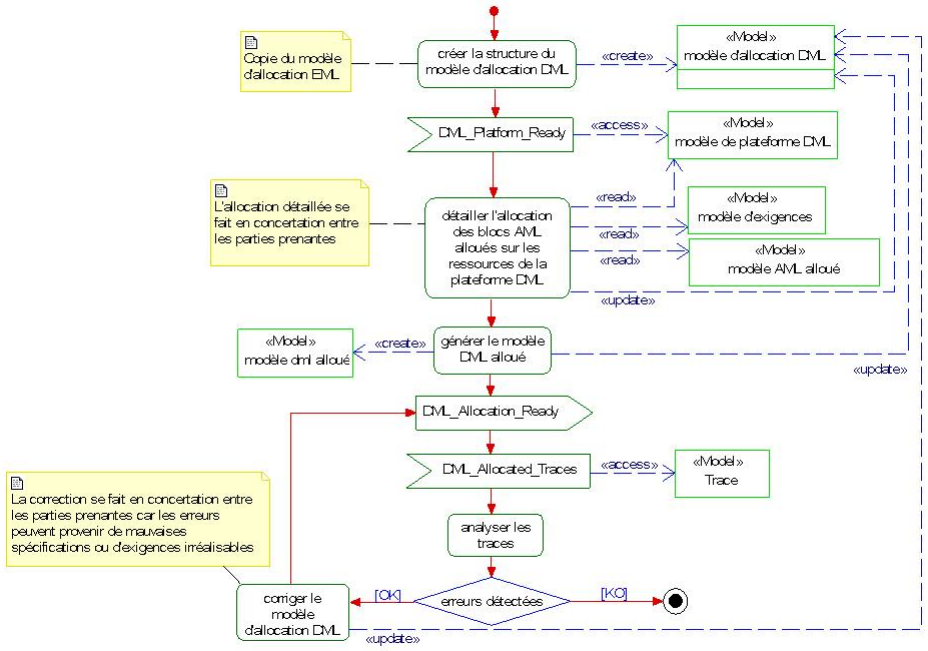


FIGURE 5.19 – Stratégie pour la spécification de l'allocation DML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Dans cette thèse, la granularité pour définir l'allocation est le composant (structure de classe). Ce déploiement permet la distribution du traitement de manière non ambigüe de l'application. Au cours du déploiement plusieurs points peuvent être raffinés automatiquement : la définition des adresses du composant dans le but de définir l'allocation mémoire sur la plateforme, les protocoles de communication, et les "wrappers" permettant de connecter différents composants.

L'étape de déploiement conduit à définir des vérifications afin de valider les choix du concepteur. Ces vérifications garantissent la qualité de l'allocation avant de générer le VHDL ou le SystemC. La génération du VHDL relève uniquement de la partie matérielle du système. Cette génération nécessite de transformer la représentation UML du processus et du contrôle en langage VHDL et d'introduire automatiquement des wrappers associés aux communications. La spécification de ce niveau doit être suffisamment détaillée pour que le code généré soit correct et permette un prototypage correct de la solution.

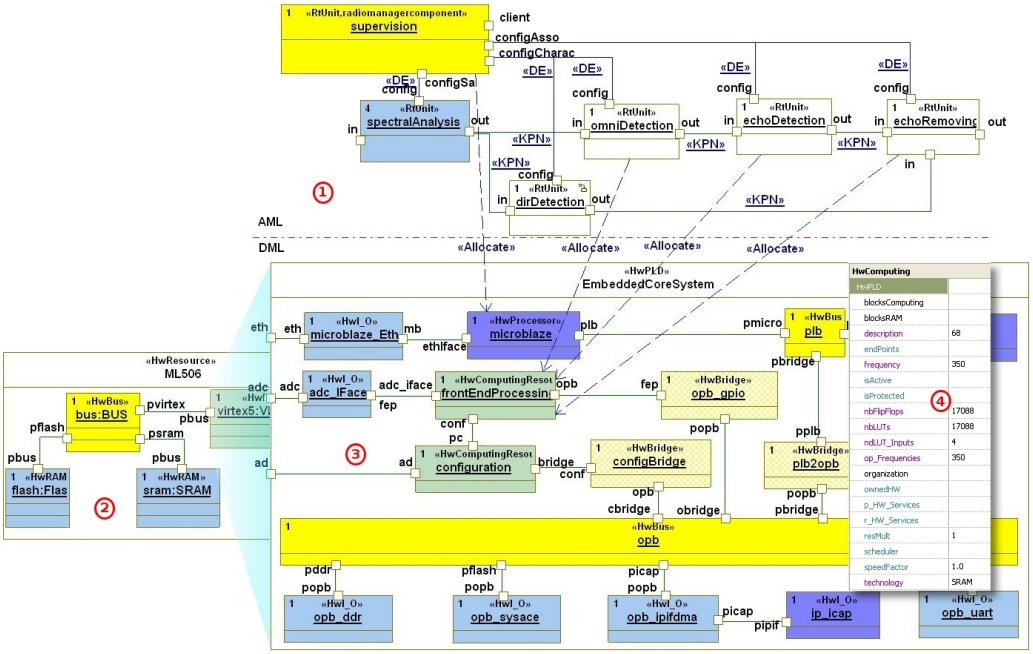


FIGURE 5.20 – Exemple d'allocation DML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Validation de l'allocation

Suite au raffinement de la plateforme d'exécution et de l'allocation, il est nécessaire de raffiner les scénarios de validation. La stratégie adoptée consiste à reprendre les scénarios définis précédemment et de les raffiner de manière à prendre en compte le raffinement de la plateforme. Aussi, il est nécessaire de s'assurer qu'aucune incohérence n'apparait dans le raffinement. Par exemple, dans le but de garantir la correction du modèle alloué, il faut vérifier que la taille des éléments alloués et celui des chemins de communication de la plateforme entre les éléments applicatifs sont cohérents de ce qui a été défini plus haut. Enfin, la génération de code SystemC permet de valider l'allocation au niveau RTL.

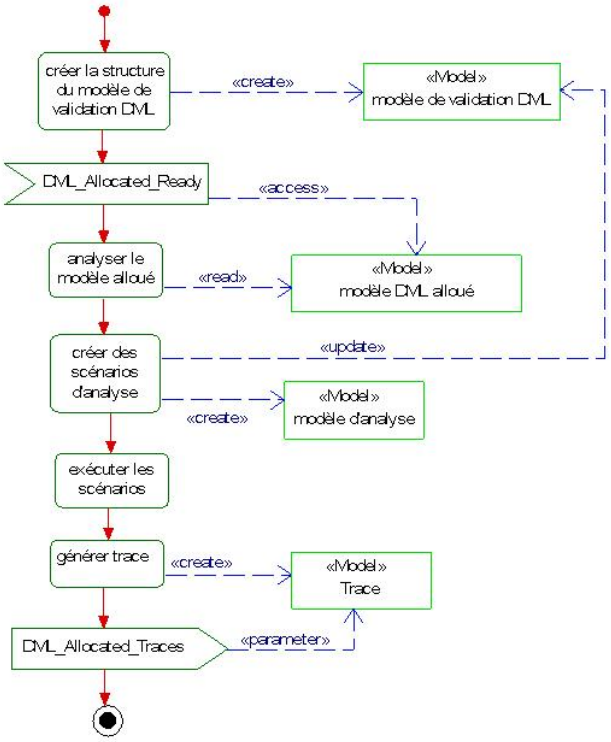


FIGURE 5.21 – Stratégie pour la validation de l'allocation DML

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

5.5 Bilan des travaux

5.5.1 Bilan de l'utilisation de MARTE

Les bénéfices de MARTE se situent essentiellement dans :
- la modélisation des plateformes à différents niveaux d'abstraction,
- la modélisation du temps à différents niveaux de granularité,
- la modélisation des propriétés non-fonctionnelles.

Les expérimentations menées autour du développement de l'application de radio intelligente nous permettent d'affirmer que le profil MARTE convient généralement bien au codesign. En outre, ce profil a l'avantage d'uniformiser et de simplifier les approches qui existaient avant lui (UML for SPT, UML for QoS&FT, UML for RT, etc.) et d'être aligné sur la version 2.1 de UML qui a intégré déjà beaucoup de concepts chers aux systémiers et aux concepteurs de plateformes (composants). Toutefois, nous avons vu que le profil MARTE avait quelques limitations dans l'expression des modèles de calcul. Pour pallier à ce problème, j'ai proposé l'extension COMETA.

Aussi, de par le nombre importants de concepts qu'il ajoute au langage UML, déjà volumineux, il reste difficile à appréhender. C'est la raison pour laquelle, il est nécessaire de définir des méthodologies de conception dédiées à des types d'application particuliers, qui explicite de manière détaillée comment l'utiliser. En outre, il serait appréciable de pouvoir identifier formellement quels concepts utiliser à quel moment du processus. C'est un des objectifs de notre contribution MODAL.

5.5.2 Bilan de l'utilisation de COMETA

L'objectif de cette section est d'évaluer le gain de COMETA d'un point de vue quantitatif et qualitatif. Sur un plan quantitatif, on retiendra les métriques du tableau 5.1 tirées de l'application de radio intelligente pour le niveau AML dans l'outil Rhapsody.

On constate sur ce tableau de métriques qu'un nombre non négligeable d'artefacts sont générés par transformation pour gérer les aspects liés aux choix de modèle de calcul. En effet, l'ensemble des éléments générés sont d'habitude codés de manière ah-doc dans les développements. Ce travail contribue ainsi à décharger les ingénieurs système d'une tâche de conception significative en effort de développement.

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Capturé		Généré	
Élément	**Nombre**	**Élément**	**Nombre**
Bloc Système	7	Composant MoC	7
Comportement Système	7	Translateur MoC	1
Port	10	Port MoC	19
Connecteur Système	9	Connecteur MoC	10
Composant MoC	7	Adapteur MoC	19
Port MoC	10	Manager MoC	7
Connecteur MoC	9		
Allocation	10		

TABLE 5.1 – Nombre d'éléments capturés et générés

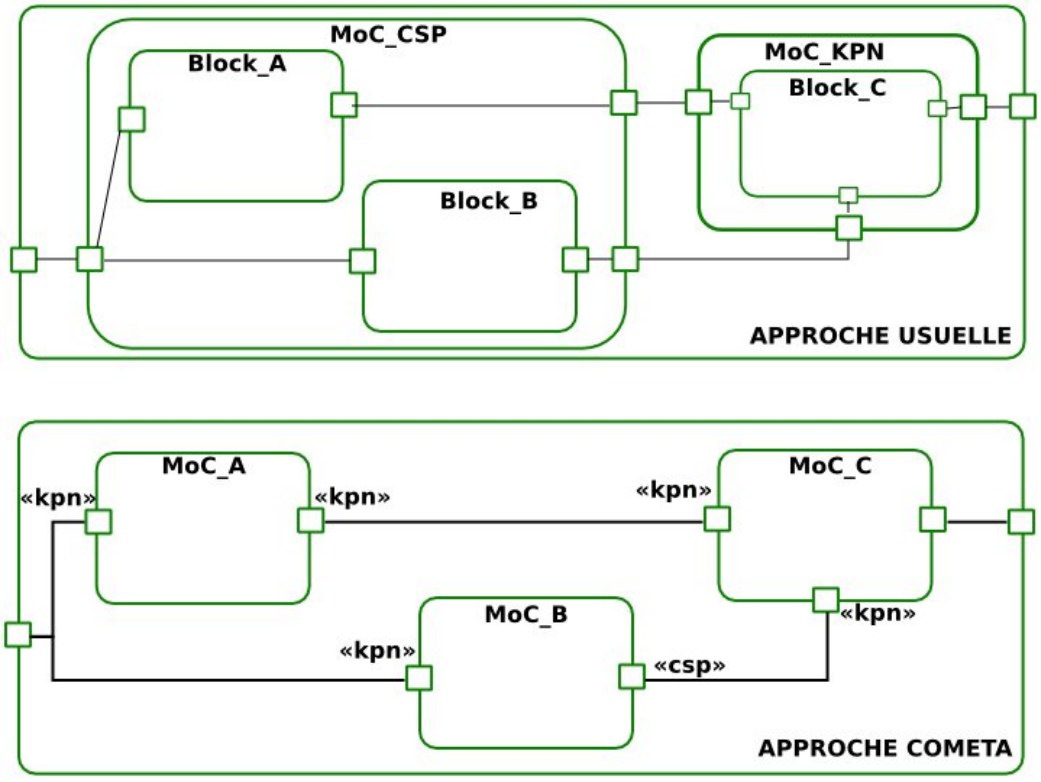


FIGURE 5.22 – Comparaison COMETA / Approche usuelle

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Sur un plan qualitatif, l'apport de COMETA apporte une réelle flexibilité dans l'implantation de MoC hétérogènes. En effet, nous avons vu dans l'état de l'art les approches proposées pour gérer l'hétérogénéité des MoCs. À notre sens, ces approches imposent trop de contraintes en termes de modélisation car il faut penser l'hétérogénéité à travers des hiérarchies de composants, ce qui n'est pas le cas de notre approche (figure 5.22). En outre, par rapport aux approches étudiées, notre approche veille à préserver le comportement applicatif, ce qui va dans le sens d'une meilleure séparation des préoccupations. Le tableau 5.2 fournit un récapitulatif des apports de Cometa par rapport à l'état de l'art.

Propriété	**MARTE**	**Mod'Helix**	**Cometa**
Représentation des données	abstrait, vecteur de bits, signal	Idem	Idem
Représentation du temps	Causal, Discret, Réel	Idem	Idem
Représentation de la concurrence	Structure et Comportement	Idem	Idem
Explicitation des communications	Partiel	Oui	Oui
Séparation des préoccupations	Non	Non	Oui

TABLE 5.2 – Comparaison MARTE vs Mod'Helix vs COMETA

5.5.3 Bilan de l'utilisation de MODAL

L'utilisation de SPEM convient bien pour la représentation des processus système et logiciel tant que les modèles capturés restent contemplatifs. Par ailleurs, un exemple de référence de l'utilisation de SPEM est le celui de la description du processus de développement logiciel "OpenUP". Le grand degré de précision de ce modèle provient essentiellement du nombre important de détails qu'il contient. Quoi qu'il en soit, ce modèle reste informel et son application est encore sujet à interprétation. Si l'on veut voir plus loin et parvenir à l'élicitation des processus à travers l'exécution de modèles, on s'aperçoit vite des limites de SPEM qui, en fin de compte, ne permet qu'une documentation mieux organisée. En particulier, il est aujourd'hui nécessaire

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

de tirer parti des bénéfices de l'ingénierie des modèles dans l'automatisation des processus. Bien sûr, nous avons vu dans le chapitre 3 qu'il existe plusieurs réponses aux problèmes posés par SPEM en particulier et par l'IDM appliquée à l'ingénierie des processus en général. Malheureusement, les solutions proposées restent somme toute locales. Aussi, à travers cette thèse, j'espère contribuer à élaborer une solution globale qui intègre ces apports afin de gérer des processus de grande taille.

Même s'il est nécessaire de mener plus d'expérimentations pour valider notre approche, nous pouvons d'ors et déjà, sur la base des 3 applications qui ont été développées dans le cadre du projet MoPCoM, tirer un bilan de l'utilisation de MODAL. En premier lieu, la construction ou l'amélioration d'un processus nécessite des points de vue permettant l'analyse des processus. Il est en effet important de décorréler le besoin en termes de processus de ses solutions. La figure 5.23 illustre ce propos. Dans SPEM, une telle séparation ne peut être exprimée. De fait, les besoins des processus ne peuvent être appréhendés qu'à travers ses solutions. Dans MODAL, l'introduction des notions d'intention et de stratégie améliorent les analyses.

FIGURE 5.23 – MODAL pour une meilleure compréhension des processus

Le choix de solutions particulières conformément à un ensemble de résolutions technologiques permet de contribuer à l'élaboration de processus exécutable à la condition que tous les aspects de ces solutions soient manipulables par la machine. Par exemple, dans SPEM, les activités peuvent être décomposées en actions élémentaires en utilisant un langage d'action approprié. Malheureusement, les artefacts produits, consommés ou transformés par ces activités restent, dans la spécification actuelle de SPEM, des boîtes noires non manipulables. Dans la mesure où le langage SPEM a clairement été défini dans le contexte IDM, il semblerait cohérent de considérer que les produits

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

des processus puissent être décrits par des modèles. En faisant cette hypothèse, nous pouvons trouver un bon compromis entre les besoins en termes de description et d'exécution. Aussi, sans le raffinement de la définition des processus proposé dans MODAL, l'expression des actions ou des contraintes restent informels. Ainsi, l'extension proposée contribue à rendre exécutables les actions et permet l'expression de contraintes automatiquement vérifiables. De plus, concernant l'application des contraintes, la notion de sévérité introduite dans MODAL apporte plus de flexibilité que SPEM. Par ailleurs, ces contraintes exprimées en KerMeta, et dont une partie se trouve en annexe, permettent de vérifier la correction des modèles pour chaque niveau de notre processus. Ces contraintes sont aujourd'hui déployées dans le MDK (Model Development Kit) de KerMeta.

Pour finir, la définition des composants de processus dans la spécification de SPEM ne permet en fin de compte que l'encapsulation de savoir-faire informels. Les clarifications proposés par MODAL permettent d'opérationnaliser les composants de processus. Au delà de l'exécutabilité, mes propositions permettent de mieux capturer la connaissance, de manière non ambigüe. Le tableau 5.3 résume les contributions de MODAL comparé à SPEM.

	SPEM	**MODAL**
Formalisation du besoin	?	Intention
Formalisation de la solution	Activité	Stratégie
Exploitation des produits	Non	Oui
Exploitation des contraintes	Non	Oui
Exécutabilité des activités	Non	Oui
Exécutabilité des composants de processus	Non	Oui
Support de la maturation	Non	Partielle

TABLE 5.3 – Comparaison SPEM vs MODAL

L'approche proposée contribue à créer des chaines d'outils à la demande qui constitue un challenge dans la gestion de l'obsolescence des technologies. Mon laboratoire d'accueil à l'ENSIETA continue de travailler sur la génération de composants de processus exécutables déployés dans un environnement distribué. Les technologies utilisées sont celles relatives à Eclipse (EMF, CNF, GMF, KerMeta, xText) et sont présentées en annexe.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

5.5.4 Bilan du processus MoPCoM

À travers le processus MoPCoM SoC / SoPC appliqué au développement du système “Cognitive Radio”, j'ai pu valider de manière expérimentale mes propositions COMETA et MODAL. Ce processus a fait l'objet de plusieurs autres expérimentations dans les domaines du traitement vidéo et du traitement d'images. Par ailleurs, les retours de ces expérimentations sont globalement positifs car il existe pour les différents niveaux de modélisation présentés, un ensemble de transformations outillés qui déchargent les ingénieurs d'activités fastidieuses et qui facilitent les analyses. Par exemple, la vérification des modèles ou la documentation sont des activités consommatrices en temps et leur automatisation par les transformations de modèle permet aux concepteurs de s'affranchir d'un travail fastidieux et de disposer de plus de temps pour trouver de meilleures solutions aux problèmes dont ils ont la charge.

5.6 Conclusion

Dans ce chapitre, j'ai présenté ma proposition de processus IDM pour le codesign, nommé MoPCoM SoC/SoPC. En particulier, je me suis davantage focalisé sur le niveau AML qui était plus dans la préoccupation de cette thèse. Les expérimentations menées et les publications m'ont ainsi permis d'établir la pertinence de l'approche proposée. Bien sûr, il faudra du temps et d'autres expérimentations pour définitivement valider et consolider ces propositions.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Chapitre 6

Conclusion

6.1 Récapitulatif de la thèse

Le contexte de cette thèse était fortement lié à des problématiques industrielles nécessitant des réponses à court et moyen termes. Après l'analyse des pratiques industrielles et des solutions existantes dans l'état de l'art, mes recherches m'ont amené à utiliser des standards de modélisation afin de mieux organiser les activités et les artefacts de l'ESL, et ce à travers un processus formalisé. L'utilisation même de ces standards a fait ressortir des manques qui étaient peu couverts par les travaux existants. Ainsi, l'identification de ces manques m'a permis de proposer des contributions originales permettant d'améliorer l'utilisation de modèles pour l'organisation des activités et des artefacts de l'ESL.

L'état de l'art de l'ingénierie des processus m'a permis d'identifier les besoins en termes de maturation et de maîtrise des processus. J'ai ainsi pu proposer une extension du langage SPEM nommé MODAL pour répondre aux problèmes de la séparation des aspects intentionnels de leur réalisation. Cette contribution a permis entre autres de clarifier la définition des artefacts de processus basés sur les modèles, et par la même de pouvoir contraindre l'utilisation des métamodèles et de clarifier la définition des composants de processus. Au delà de l'organisation des activités de l'ESL, les contraintes de processus exprimées sur les activités et les artefacts permettent également de formaliser les règles liées à l'application des standards et des normes, et d'en assurer ainsi de manière automatique la bonne application. Cette contribution, dans le cadre de processus certifiables, constitue un atout certain.

L'état de l'art de l'ESL m'a permis d'extraire l'essence des développe-

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

ments de systèmes sur puce afin de mieux comprendre les besoins en termes de conception et d'analyse. Ce travail m'a permis d'identifier les manques dans l'adéquation entre le langage MARTE et les artefacts de l'ESL. L'extension du langage COMETA qui découle de cette analyse contribue à combler ces manques en permettant de meilleures représentations de modèles de calculs, essentiels aux activités de l'ESL. De plus, la génération automatique de librairies de modèles de composants MoC offre une meilleure capitalisation de patrons de communication les plus couramment utilisés. Par exemple, dans le domaine du traitement du signal, le modèle de calcul "KPN" est très utilisé et dans la pratique actuelle, les ingénieurs matériels sont souvent amenés à réimplanter ces mécanismes de communication de manière ad-hoc et bien souvent sans savoir qu'ils implantent du "KPN".

Grâce à ces contributions, j'ai pu, dans le cadre du projet MoPCoM, proposer la définition d'un processus formalisé dédié à la conception de système sur puce. Ce processus basé sur l'ingénierie des modèles répond de manière claire aux attentes de la communauté ESL car il sépare bien les préoccupations et permet de fait une meilleure exploration d'architecture. En effet, le processus MoPCoM, résumé dans la figure 6.1, considère la modélisation de la plateforme à trois niveaux d'abstraction donnant lieu à 3 types d'allocation et diverses analyses menant à la construction du système de manière itérative :

Le niveau AML décrit une plateforme virtuelle exprimant le niveau de concurrence et de pipeline souhaité ainsi que l'allocation de l'application sur cette plateforme. Le résultat de cette allocation représente un premier niveau d'implantation qui permet d'analyser la correction fonctionnelle au regard des contraintes non fonctionnelles.

Le niveau EML décrit une plateforme d'exécution à gros grain, c'est à dire en termes de nœuds d'exécution, de communication ou de mémorisation. Le résultat de l'allocation AML sur cette plateforme permet de déterminer l'adéquation entre les algorithmes et l'architecture physique.

Le niveau DML décrit une plateforme d'exécution à grain fin qui est un raffinement du niveau EML. L'allocation du modèle AML alloué sur cette plateforme permet de mener des analyses au cycle près et au bit près. L'obtention d'une allocation au niveau DML jugée satisfaisante au regard des exigences fonctionnelles et non-fonctionnelles permet de générer le code d'implantation pour les différentes parties matérielles

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

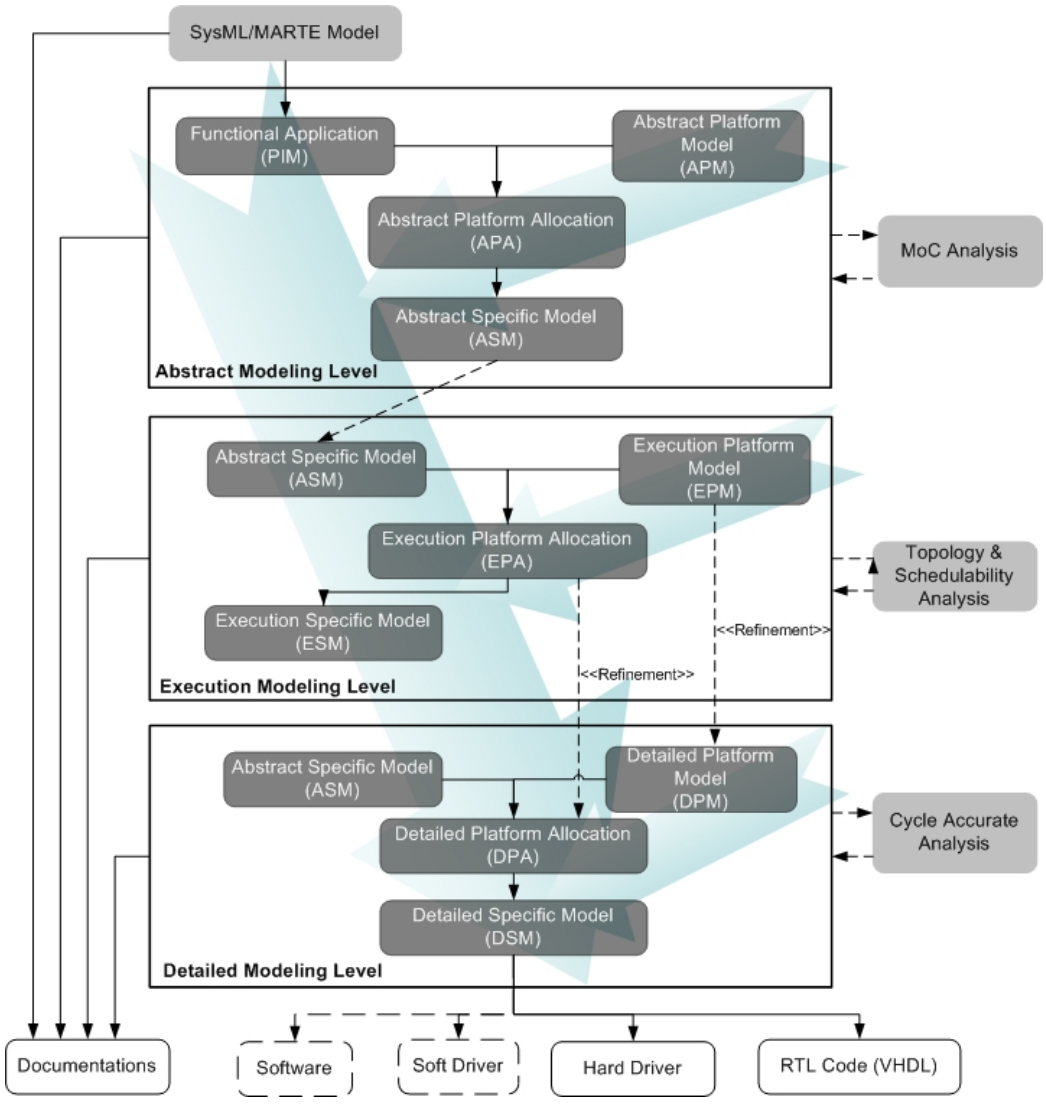


FIGURE 6.1 – Processus MoPCoM

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

et logicielles, ainsi de la glue logique permettant les communications entre les blocs de natures différentes.

Pour chaque niveau, j'ai identifié et formalisé l'ensemble des concepts nécessaires et suffisants permettant de gérer les activités associées en collaboration avec les partenaires du projet MoPCoM. Il résulte de ce travail une meilleure gestion des activités et des artefacts de l'ESL par les modèles. Ainsi, les ingénieurs système ou matériel s'en retrouvent mieux guidés dans leurs activités d'analyse car l'outillage d'un tel processus, rendu possible par l'utilisation de modèles et de transformations de modèles, permet de les décharger d'activités fastidieuses telles que le codage ou la documentation. Cette formalisation permet notamment une meilleure maturation du processus par une séparation claire des aspects intentionnels de leur réalisation.

Les contributions que je propose ont été validées de manière expérimentale sur un projet industriel servant de modèle de référence dans la société Thales. Le bilan de la mise en œuvre de ces contributions a fait ressortir un gain évident par rapport aux pratiques actuelles mais également par rapport à l'état de l'art. Ce travail a donné lieu à un outillage développé et déployé par la société Sodius et déployé dans l'outil de modélisation Rhapsody de IBM dans le contexte du projet MoPCoM. Il a également donné lieu au développement d'un outil de modélisation supportant le langage MODAL, développé et maintenu par l'ENSIETA.

6.2 Analyse critique

L'expérimentation qui m'a permis de valider mes contributions, bien qu'étant complexe, est une application somme toute "locale". J'ai été amené à travailler avec d'autres ingénieurs à l'élaboration de ce modèle et je pense qu'ils ont été convaincus par l'approche proposée, ce qui était loin d'être évident au début. Mais il reste des problèmes de taille des projets industriels qui freinent encore l'adoption d'une telle approche. En effet, nos expérimentations nous ont amené à considérer que les modèles se prêtent encore mal au travail collaboratif sur des projets industriels nécessitant de la gestion de configuration.

Ensuite se pose le problème de la standardisation. En effet, un des arguments important avancé lors de la préconisation de l'utilisation de modèles était l'indépendance aux outils et un meilleur échange de modèles. Or, l'adoption des extensions proposées est conditionnée par leur acceptation par les

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

communautés respectives, ce qui peut s'avérer difficile quand il n'existe pas de consensus sur les sujets traités. Par exemple, le choix des 3 niveaux de représentation de plateformes AML, EML et DML a été inspiré par l'état de l'art de l'ESL, qui définit également 3 niveaux de programmation qui sont PV (Programmer's View), PVT (Programmer's View + Time) et CC (Cycle Callable). Et bien que la nécessité de ces niveaux soit reconnue, de même que leur label, personne de la communauté ESL n'est capable aujourd'hui de s'accorder sur la définition et le contenu de ces niveaux, et les choses évoluent très vite. Dans ce cadre, il est difficile de se positionner correctement et la proposition peut s'avérer pertinente aujourd'hui et caduque demain. Derrière ce problème se pose également celui de la définition de la notion de niveau d'abstraction qui est essentielle.

La proposition du métamodèle MODAL a certaines limites dans la capitalisation des savoir-faire. En effet, beaucoup des décisions humaines ne peuvent se résumer à de simples GO-NOGO. Par exemple, l'évaluation des risques se base sur différents critères dont il faudrait tenir compte dans la construction du processus : fiabilité des outils utilisés, compétences des acteurs du développement, etc. Là encore se pose le problème du passage à l'échelle car le modèle de processus que j'ai réalisé dans le cadre du projet MoPCoM est un modèle conséquent.

Concernant le retour d'expérience du développement de l'application "Cognitive Radio", il subsiste quelques points sur lesquels les modèles n'ont pu nous aider. Nous avons ainsi constaté une inadéquation entre l'utilisation des modèles de type UML et les métiers du traitement du signal. Typiquement, pour la partie "acquisition des données" de l'application, nous avons dû recourir à des outils plus adaptés au métier comme MATLAB. En effet, l'interfaçage entre le domaine du continu et celui du discret est aujourd'hui encore mal traité par l'ingénierie des modèles.

Enfin, le code généré à l'issu de l'analyse du niveau DML, bien que simulable et synthétisable, ne permet pas de faire une utilisation optimale des ressources qu'offrent la plateforme. C'est là un des arguments forts des développeurs de code VHDL encore réticents aux modèles. Mais nous ne doutons pas que les futurs évolutions techniques et technologiques résoudront ce problème.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

6.3 Perspectives

Bien évidemment, au vu de l'ampleur du travail à réaliser, tous les aspects du processus ou de sa formalisation n'ont pu être couverts. Aussi avons nous identifié un certain nombre de travaux en perspectives.

Concernant la proposition du processus MoPCoM, des travaux futurs pourront faire le lien entre les modèles d'analyse du processus MoPCoM et les outils d'analyse du marché. Par exemple, au niveau AML, on peut envisager l'établissement d'une passerelle entre les modèles COMETA et les outils d'analyse de MoC comme Ptolemy. Aussi, au niveau EML, on peut envisager l'utilisation de passerelles existantes entre les modèles MARTE et les outils d'analyse d'ordonnançabilité telles que CHEDDAR. Ces travaux devront permettre de réinjecter le résultat des analyses dans les modèles afin de permettre, par la suite, une exploration d'architecture automatisée.

Les futurs travaux autours de COMETA devront s'attacher à mieux prendre en compte les domaines orthogonaux autre que les communications. Aussi, les expérimentations que nous avons mené étaient essentiellement cadrées par des modèles UML et il sera nécessaire dans les futurs travaux de prendre plus de hauteur et de généraliser COMETA pour tout type de DSML (Domain Specific Modeling Language).

Dans les motivations, nous avons parlé de l'importance du traitement de l'obsolescence. Malheureusement, le temps nous a manqué pour montrer que l'approche préconisée permet effectivement de traiter ce problème. Des travaux futurs pourront reprendre le développement de la "Cognitive Radio" et évaluer l'impact d'un changement de plateforme sur les temps de conception et d'analyse. Ces travaux permettraient notamment de réduire de manière significative le coût des risques liés aux choix technologiques.

Concernant la formalisation des processus, et dans la mesure où les interfaces des composants sont mieux identifiées, un travail important reste à faire sur la provision d'un cadre d'exécution permettant l'opérationnalisation des composants de processus répartis. Ce travail permettra de tendre vers la construction de chaînes d'outils prenant en charge une grande partie du processus et de formaliser les rapports entre maîtres d'œuvre et maître d'ouvrage.

Enfin, concernant les composants de processus, l'ajout de mécanismes de paramétrisation, permettraient à l'avenir de pouvoir configurer les composants de processus pour les adapter aux organisations qui les emploient ou à

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

la gestion des lignes de produit. Là encore, ces travaux permettront de réduire les coûts de l'entreprise liés à l'évolution d'un produit ou de son adaptation pour différents marchés.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Chapitre 7

Annexes

Contraintes d'utilisation de MARTE par niveau

Dans cette annexe, nous présentons quelques unes des contraintes KerMeta par niveau de modélisation. Ces contraintes permettent de définir entre autres des métamodèles par niveau d'abstraction comme sous ensembles du profil MARTE. L'ensemble des contraintes ont été implantées dans le projet MoPCoM et nous renvoyons le lecteur sur le site du projet http://www.mopcom.fr ou sur celui de KerMeta http://www.kermeta.org pour de plus amples informations.

Le niveau AML

```
1
aspect class RtBehavior
3 {
  /**
5  * If the owner of the RtBehavior is a protected passive unit
       both queueSchedPolicy and queueSize are not applicable.
   * Reference MARTE 07-08-04, p160
7  */
   inv owner passiveUnit is do
9
    var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
11  ph.initialize(self.containingResource,self.containingResource)
```


© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
13      if
        ph.getStereotypesFor(self.base_Behavior.asType(uml::
            NamedElement).owner).exists{st
15        |st.isInstanceOf(MARTE::MARTE_DesignModel::RTEMoCC::PpUnit)
              } then
        self.queueSchedPolicy.isVoid and self.queueSize.isVoid
17      end
    end
19  }

21  aspect class RteConnector
    {
23    /**
      * Attribute packetT should not be use because irrelevant at
          this level
25    */
      inv noPacketT is do
27      self.packetT.isVoid
      end
29
      /**
31
      * Attribute blockT should not be use because irrelevant at
          this level
33    */
      inv noBlockT is do
35      self.blockT.isVoid
      end
37  }

39  aspect class RteUnit
    {
41    /**
       * RTUnits communicate through ports
43     */
      inv useOnlyPorts is do
45      true
      end
47
    }
49
    aspect class RtUnit
51  {
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
  /**
53 * If isDynamic is true, the real-time unit do not owns a pool
       of schedulable resources. Hence, poolSize, poolPolicy and
   * poolWatingPolicy are not applicable.
55 * typo see doc
   * reference MARTE 07-08-04, p163
57 */
   inv isDynamic is do
59 if self.isDynamic then
     self.poolSize.isVoid and
61   self.poolPolicy.isVoid
     self.poolWaitingTime.isVoid
63 end
end
65
 /**
67 * A main real-time unit has to own a main operation.
   * reference MARTE 07-08-04, p163
69 */
 inv mainRT is do
71 if self.isMain then
     not self.main.isVoid
73 end
 end
75 }
```

Niveau EML

```

2 aspect class Release
  {
4   /**
    * The resource that owns the service must be a protected
        resource (i.e., its attribute isProtected must be true).
6   * reference MARTE 07-08-04, p104
    */
8   inv isResProtected is do
      self.owner.isProtected
10  end
  }
12
  aspect class MemoryBroker
14 {
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
  /**
16 * Types of memories values must be stereotyped either as "
       StorageResource" or as "StorageResource" sub-Stereotype.
   *
18 * reference MARTE 07-08-04, p190
   */
20 inv mem types is do
   var ph : profiledResourceHelper::ProfilesHelper init
       profiledResourceHelper::ProfilesHelper.new
22 ph.initialize(self.containingResource,self.containingResource)

24 self.memories.forAll{t | ph.getStereotypesFor(t.type).exists{
       st|st.isKindOf(MARTE::MARTE Foundations::GRM::
       StorageResource)}}

26 end
}
28
aspect class SecondaryScheduler {
30
  /**
32 * A SecondaryScheduler takes its capacity from the
       virtualProcessingUnits list of schedulable resources, so it
        is not
   * possible to have processing resources capacity through the
       processingUnits list inherited from Scheduler.
34 * reference MARTE 07-08-04, p108
   */
36 inv no proc capacity is do
     self.processingUnit.size == 0
38 end
}
40
aspect class TimerResource {
42
  /**
44 * TimerResource isn't allowed in AML models
   */
46 inv forbiddenTimerResourceInAML is do
     var ph : profiledResourceHelper::ProfilesHelper init
         profiledResourceHelper::ProfilesHelper.new
48   ph.initialize(self.containingResource,self.
         containingResource)
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
50      var elem : uml::Element init ph.getBase(self)
        not amlModelHelper::AMLModelHelper.new.isInAML(elem)
52
    end
54

56 }

58
  aspect class Model
60
  {
62   /**
      * APA model must not contain mutual exclusion resources
64    */
      inv noMutualExclusionInModel is do
66      var ph : profiledResourceHelper::ProfilesHelper init
            profiledResourceHelper::ProfilesHelper.new
        ph.initialize(self.containingResource,self.
            containingResource)
68      var res : Boolean init false
        res :=  not self.allOwnedElements.exists{cl | cl.
            isInstanceOf(Class)
70          and ph.getStereotypesFor(cl).exists{st| st.isInstanceOf(
              MARTE::MARTE_Foundations::GRM::MutualExclusionResource
              )}
        }
72      res
    end
74 }

76 aspect class Scheduler
  {
78   /**
     The scheduling policy of the scheduler must be compatible with
         the scheduling parameters of all the schedulable
80   * resources that it has associated.
     * reference MARTE 07-08-04, p108 (unclear invariant see doc)
82     */
       inv shedPolicy is do
84       true
       end
86
       /**
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
88      * The scheduling policy of the scheduler must be compatible
            with the ProtectProtocolParameters of all the associated
    * MutualExclusionResources.
90  * reference MARTE 07-08-04, p108 (unclear invariant see doc)
      */
92    inv shedPolicy is do
        true
94    end
  }
96
  aspect class ResourceUsage
98{
    /**
100 * To consider the ResourceUsage fully specified, if the list
        usedResources is empty the list subUsages should not be
    * empty and viceversa. Further refinements of ResoureUsage may
        define additional attributes that may bring implicit
102 * elements into the usedResources list.
    * reference MARTE 07-08-04, p106
104 */
    inv ru fully specified is do
106   //true
      if self.usedResources.size == 0 then
108     self.subUsage.size > 0
      else
110     if self.subUsage.size ==0 then
          self.usedResources.size > 0
112     end
      end
114 end

116 /**
    * If the list usedResources has only one element, all the
        optional lists of attributes
118 *(execTime, msgSize,allocatedMemory, usedMemory, powerPeak and
        energy) refer to this unique Resource and at least one of
      them must
        * be present. reference MARTE 07-08-04, p106 (bad
            metamodel design see doc)
120 */
    inv usedRes one is do
122   true
    end
124
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
  /**
126 * If the list usedResources has more than one element, all of
      the optional lists of attributes
  * (execTime, msgSize, allocatedMemory, usedMemory, powerPeak,
      and energy) that are present, must have that number of
      elements, and
128 * they will be considered to match one to one.
  * reference MARTE 07-08-04, p106 (bad metamodel design see doc
      )
130 */
  inv usedRes more than one is do
132   true
  end
134
  /**
136 * If the list subUsages is not empty, and any of the optional
      lists of attributes
  * (execTime, packetSize, allocatedMemory, usedMemory,
      powerPeak, and energy) is present, then more than one
      annotation
138 * for the same resource and kind of usage may be expressed.
  * In this case, if the annotations have also the same source
      and statistical qualifiers
140 * they will be considered in conflict, and hence the
      ResourceUsage inconsistent.
  * reference MARTE 07-08-04, p106 (bad metamodel design see doc
      )
142 */
  inv subUsage not empty is do
144   true
  end
146


148 }

150 aspect class MemoryPartition
  {
152   /**
  * Types of concurrentResources values must be stereotyped
      either as "SwConcurrentResource" or as
154 * "SwConcurrentResource" sub-Stereotype.
  * reference MARTE 07-08-04, p191
156 */
  inv concRes types is do
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
158
    var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
160 ph.initialize(self.containingResource,self.containingResource)

162 self.concurrentResources.forAll{t | ph.getStereotypesFor(t.
        type).exists{st
            |st.isKindOf(MARTE::MARTE DesignModel::SRM::SW
                Concurrency::SwConcurrentResource)}}
164 end

166 /**
    * Types of memorySpaces values must be stereotyped either as "
        StorageResource" or as "StorageResource"
168 * sub-Stereotype.
    * reference MARTE 07-08-04, p191
170 */
    inv memSpace types is do
172 var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
    ph.initialize(self.containingResource,self.containingResource)
174
    self.memorySpaces.forAll{t | ph.getStereotypesFor(t.type).
        exists{st|st.isKindOf(MARTE::MARTE Foundations::GRM::
        StorageResource)}}
176 end

178 }
    aspect class SwSchedulableResource
180 {
    /**
182 * [1] The type of scheduler value must be stereotyped either
        as "Scheduler" or as "Scheduler" sub-Stereotype.
    * reference MARTE 07-08-04, p202
184 */
    inv shedType is do
186
    var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
188 ph.initialize(self.containingResource,self.containingResource)

190 ph.getStereotypesFor(self.scheduler).exists{st|st.isKindOf(
        MARTE::MARTE Foundations::GRM::Scheduler)}
    end
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
}

aspect class CommunicationMedia {

  /**
   * each communication media is stereotyped HwBus, except
       bridges between buses that are stereotyped HwBridge
   */
   inv communicationMediaIsHwBus is do
    true
   end

   /**
   * each communication media is stereotyped HwBus, except
       bridges between buses that are stereotyped HwBridge
   */
   inv busBridgeIsHwBridge is do
    true
   end
}
```

Le niveau DML

```

aspect class HwCache {

  /**
   * each Component stereotyped with HwCache must have its
       attributes
          * structure and type set
   */
  inv structureAndTypeAreSet is do
    var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
    ph.initialize(self.containingResource,self.
        containingResource)
    var modelHelper : dmlModelHelper::DMLModelHelper init
        dmlModelHelper::DMLModelHelper.new

    var component : uml::Component
    component ?= ph.getBase(self)
    // check if the stereotyped element is in the DML package
        and if this is a Component
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
    if not component.isVoid then
17    ( not structure.isVoid ) and (not type.isVoid )
    else
19    true
    end
21  end

23 }

25 aspect class Package
   {
27  /**
     * recursively check if this Package contains a class
         stereotyped "HwClock"
29   */
    operation containsAnHwClock(ph : profiledResourceHelper::
       ProfilesHelper) : Boolean is do
31    result := self.packagedElement.select{elem | elem.
         isInstanceOf(uml::BehavioredClassifier) }.exists{
         classifierElem |
              ph.getStereotypesFor(classifierElem).exists{st|
33            st.isInstanceOf(MARTE::MARTE DesignModel::HRM::
                 HwLogical::HwTiming::HwClock)}}
      if not result then
35      // try with one of the subpackages
        result := self.packagedElement.select{elem | elem.
           isInstanceOf(uml::Package) }.exists{subPackage |
37         subPackage.asType(uml::Package).containsAnHwClock(ph
               )}
      end
39  end
   }
41

   aspect class HwCache
43 {
   /**
45 * [4] memorySize is derived from structure attribute.
   * reference MARTE 07-08-04, p234
47 * Not clear to me how this can be derived (see doc)
   */
49 inv memSize derived is do
     true
51 end
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
53    /**
      * [5] addressSize is greater than the total cache entries
          number derived from the structure attribute.
55    * reference MARTE 07-08-04, p234
      */
57    inv addressSize is do
      var dCacheSize: Integer init Integer.new
59        self.caches.each{c|dCacheSize := dCacheSize + c.structure.
              nsSets.~value * c.structure.blockSize.~value
          * c.structure.associativity.~value}
61
      self.addressSize.~value > dCacheSize
63
      end
65
  }
67
  aspect class HwROM
69{
      /**
71    *[21] memorySize is derived from organization attribute.
      * reference MARTE 07-08-04, p245
73    * misses a more precise description (see doc)
      */
75    inv memSize is do
        true
77    end

79    /**
      * [22] addressSize is greater than the number of memory words
          derived from organization attribute.
81    * reference MARTE 07-08-04, p245
      * (assumed word number = rows* colums / wordSize)
83    */
      inv addressSize is do
85      var wordNumber : Real init
        (self.organization.nbColumns.~value * self.organization.
            nbRows.~value).toReal / self.organization.wordSize.~value
87       self.addressSize.~value > wordNumber
      end
89}

91aspect class HwComputingResource {
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
93    /**
       * HwComputingResource must set their frequency
95     */
    inv frequencyIsSet is do
97      var ph : profiledResourceHelper::ProfilesHelper init
            profiledResourceHelper::ProfilesHelper.new
        ph.initialize(self.containingResource,self.
            containingResource)
99      var modelHelper : dmlModelHelper::DMLModelHelper init
            dmlModelHelper::DMLModelHelper.new
        // check if the stereotyped element is in the DML package
101     if modelHelper.isInDML(ph.getBase(self)) then
          not self.op Frequencies.isVoid
103     else
          true
105     end
    end
107 }

109 aspect class Package
    {
111   /**
       * recursively check if this Package contains a class
           stereotyped "HwResource"
113    */
    operation containsAnHwResource(ph : profiledResourceHelper::
        ProfilesHelper) : Boolean is do
115     result := self.packagedElement.select{elem | elem.
            isInstanceOf(uml::BehavioredClassifier) }.exists{
            classifierElem |
                ph.getStereotypesFor(classifierElem).exists{st|
117                 st.isInstanceOf(MARTE::MARTE_DesignModel::HRM::
                        HwLogical::HwGeneral::HwResource)}}
        if not result then
119       // try with one of the subpackages
          result := self.packagedElement.select{elem | elem.
              isInstanceOf(uml::Package) }.exists{subPackage |
121           subPackage.asType(uml::Package).containsAnHwResource
                  (ph)}
        end
123   end
    }
125
    aspect class Package
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
{
  /**
   * recursively check if this Package contains a class
       stereotyped from one of  stereotype "HwPower"
   */
  operation containsElementFromHwPower(ph :
      profiledResourceHelper::ProfilesHelper) : Boolean is do
    result := self.packagedElement.select{elem | elem.
        isInstanceOf(uml::BehavioredClassifier) }.exists{
        classifierElem |
            ph.getStereotypesFor(classifierElem).exists{st|
            st.isInstanceOf(MARTE::MARTE_DesignModel::HRM::
                HwPhysical::HwPower::HwCoolingSupply)
            or st.isInstanceOf(MARTE::MARTE_DesignModel::HRM::
                HwPhysical::HwPower::HwPowerSupply)}}
      if not result then
        // try with one of the subpackages
        result := self.packagedElement.select{elem | elem.
            isInstanceOf(uml::Package) }.exists{subPackage |
            subPackage.asType(uml::Package).
                containsElementFromHwPower(ph)}
      end
  end
}

aspect class HwDMA {
  /**
   * each Component stereotyped with HwDMA must have its
       attributes
         nbChannels and transferWidth set
   */
  inv nbChannelsAndTransferWidthAreSet is do
    var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
    ph.initialize(self.containingResource,self.
        containingResource)
    var modelHelper : dmlModelHelper::DMLModelHelper init
        dmlModelHelper::DMLModelHelper.new

    var component : uml::Component
    component ?= ph.getBase(self)
    // check if the stereotyped element is in the DML package
        and if this is a Component
    if not component.isVoid then
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
      ( not nbChannels.isVoid ) and (not transferWidth.isVoid )
159     else
          true
161     end
    end
163
}
165
aspect class HwComponent
167 {
    /**
169  * [6] area must derive from dimensions.
     * reference MARTE 07-08-04 p235
171  * cannot be implemented because the dimensions attribute is
         not ordered (see doc)
     */
173  inv area is do
       true
175  end

177  /**
     * [7] subComponents positions must not exceed the grid.
179  * MARTE 07-08-04, p235
     * similar problem (see doc)
181  */
     inv pos is do
183    true
     end
185
     /**
187  * [8] requiredConditions intervals must be included within the
         subcomponents corresponding intervals.
     * MARTE 07-08-04, p235
189  * unclear invariant (see doc)
     */
191  inv reqConditions is do
       true
193  end
}
195
aspect class HwProcessor {
197
     /**
199   * HwProcessor must specify the nbCore and nbAlu
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
    */
201 inv nbCoreAndnbAluAreSet is do
      var ph : profiledResourceHelper::ProfilesHelper init
          profiledResourceHelper::ProfilesHelper.new
203   ph.initialize(self.containingResource,self.
          containingResource)
      var modelHelper : dmlModelHelper::DMLModelHelper init
          dmlModelHelper::DMLModelHelper.new
205   // check if the stereotyped element is in the DML package
      if modelHelper.isInDML(ph.getBase(self)) then
207     not self.nbALUs.isVoid and not self.nbCores.isVoid
      else
209     true
      end
211 end

213 /**
    * each HwProcessor is linked to at least one HwStorage
        through an HwBus
215 */
    inv isLinkedToHwStorage is do
217   true
    end
219
    /**
221 * if an HW Processor owns several cache levels, it implies
        the uniqueness of levels
    * ie. one L1, one L2, etc
223 */
    inv noDuplicateLevelsInCache is do
225   var ph : profiledResourceHelper::ProfilesHelper init
          profiledResourceHelper::ProfilesHelper.new
      ph.initialize(self.containingResource,self.
          containingResource)
227   var modelHelper : dmlModelHelper::DMLModelHelper init
          dmlModelHelper::DMLModelHelper.new
      if modelHelper.isInDML(ph.getBase(self)) then
229     if caches.size > 1 then
          caches.forAllCpl{c1, c2|
231           (c1.level.~value==c2.level.~value ).implies(c1==c2)
          }
233     else
          true
235     end
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
    else
      true
    end
  end

  /**
   * If there are more than two cache (HWCache) levels in a
       HWProcessor
   * then the frequency of cache Ln must be higher or equal to
       the frequency of cache Ln+x
   */
  inv coherentFrequenciesInCacheLevels is do
    var ph : profiledResourceHelper::ProfilesHelper init
        profiledResourceHelper::ProfilesHelper.new
    ph.initialize(self.containingResource,self.
        containingResource)
    var modelHelper : dmlModelHelper::DMLModelHelper init
        dmlModelHelper::DMLModelHelper.new
    if modelHelper.isInDML(ph.getBase(self)) then
      if caches.size > 1 then
        caches.forAllCpl{c1, c2|
            (c1.level.~value < c2.level.~value ).implies(c1.
                frequency.~value >= c2.frequency.~value)
          }
      else
        true
      end
    else
      true
    end
  end

  /**
   * each HwProcessor is linked to at least one HwRAM through an
       HwBus
   */
  inv isLinkedToHwStorage is do
    true
  end

  /**
   * HwProcessor must specify the nbAlu and nbPipelines
   */
  inv nbCoreAndnbPipelinesAreSet is do
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
273     var ph : profiledResourceHelper::ProfilesHelper init
            profiledResourceHelper::ProfilesHelper.new
        ph.initialize(self.containingResource,self.
            containingResource)
275     var modelHelper : dmlModelHelper::DMLModelHelper init
            dmlModelHelper::DMLModelHelper.new
        // check if the stereotyped element is in the DML package
277     if modelHelper.isInDML(ph.getBase(self)) then
          not self.nbALUs.isVoid and not self.nbPipelines.isVoid
279     else
          true
281     end
    end
283
    /**
285  * component stereotyped with HwProcessor must be composed of
         with at least one component stereotyped with HwCache
     */
287 inv isComposedWithHwCache is do
      var ph : profiledResourceHelper::ProfilesHelper init
          profiledResourceHelper::ProfilesHelper.new
289   ph.initialize(self.containingResource,self.
          containingResource)
      var modelHelper : dmlModelHelper::DMLModelHelper init
          dmlModelHelper::DMLModelHelper.new
291   var component : uml::Component
      component ?= ph.getBase(self)
293   // check if the stereotyped element is in the DML package
          and if this is a Component
      if not component.isVoid then
295     component.nestedClassifier.exists{classifierElem |
          ph.getStereotypesFor(classifierElem).exists{st|
297       st.isInstanceOf(MARTE::MARTE_DesignModel::HRM::HwLogical
              ::HwMemory::HwCache)}
        }
299   else
        true
301   end
    end
303 }

305 aspect class HwMemory {

307   /**
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
    * each Component stereotyped with HwMemory must be connected
        to a component
309 * stereotyped with HwStorageManager (or one of its children)
    */
311 inv isConnectedToHwStorageManager is do
      var ph : profiledResourceHelper::ProfilesHelper init
          profiledResourceHelper::ProfilesHelper.new
313   ph.initialize(self.containingResource,self.
          containingResource)
      var modelHelper : dmlModelHelper::DMLModelHelper init
          dmlModelHelper::DMLModelHelper.new
315
      var component : uml::Component component ?= ph.getBase(self)
317               // check if the stereotyped element is in the
                      DML package and
                  if this is a Component if not component.isVoid
                      then true //and
319               self.isSynchronous == true then // self.ownedHW.
                      exists{ hw |
                    hw.isInstanceOf(MARTE::MARTE
321                 DesignModel::HRM::HwLogical::HwTiming::HwClock
                        )} else true
                  end end
323
    /**
325  * each Component stereotyped with HwMemory must have its
        attributes memorySize, addressSize and timings set
     */
327 inv memorySizeAddressSizeTimingsAreSet is do
      var ph : profiledResourceHelper::ProfilesHelper init
          profiledResourceHelper::ProfilesHelper.new
329   ph.initialize(self.containingResource,self.
          containingResource)
      var modelHelper : dmlModelHelper::DMLModelHelper init
          dmlModelHelper::DMLModelHelper.new
331
      var component : uml::Component
333   component ?= ph.getBase(self)
      // check if the stereotyped element is in the DML package
          and if this is a Component
335   if not component.isVoid then
        ( not memorySize.isVoid ) and (not adressSize.isVoid ) and
            (not (timings.size == 0))
337   else
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010


```
        true
339     end
    end
341
}
343
aspect class HwComponent {
345
    /**
347  * no element stereotyped with HwPhysical in a DML model
     */
349  @note "as, HwCoolingSupply and HwPowersupply inherit from
        HwComponent, they are also forbidden in DML model using
        this invariant"
    @implemented "yes"
351  @mopcomConstraintUID "DML017"
    inv noHwPhysicalInDML is do
353     var ph : profiledResourceHelper::ProfilesHelper init
            profiledResourceHelper::ProfilesHelper.new
        ph.initialize(self.containingResource,self.
            containingResource)
355     var modelHelper : dmlModelHelper::DMLModelHelper init
            dmlModelHelper::DMLModelHelper.new
        // return false if the stereotyped element is in the DML
            package
357     not modelHelper.isInDML(ph.getBase(self))
    end
359
}
```

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Outillage du métamodèle MODAL

Afin d'outiller le métamodèle MODAL, nous avons utilisé les technologies fournies par l'outil Eclipse (http ://www.eclipse.org) telles que EMF, GMF ou CNF.

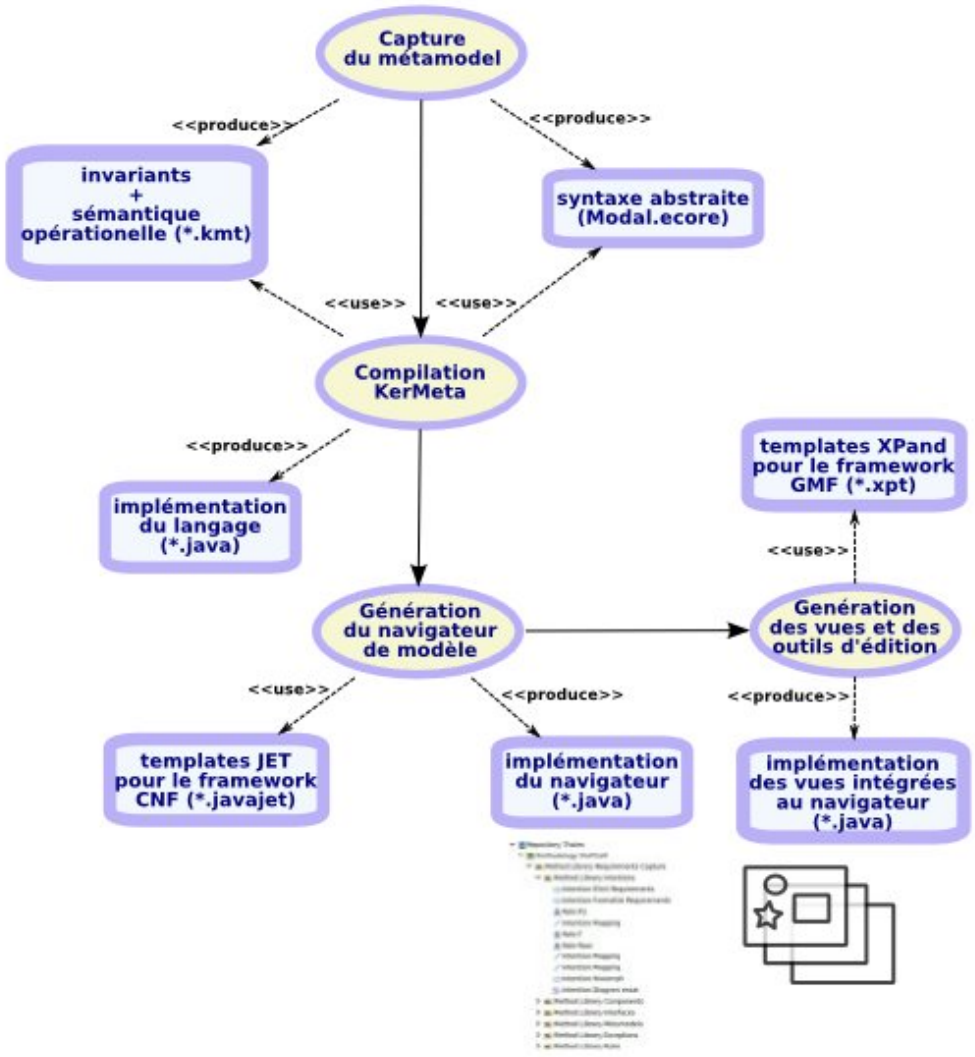


FIGURE 7.1 – Outillage du métamodèle MODAL

La figure 7.1 résume l'outillage de MODAL. Le métamodèle MODAL a tout d'abord été capturé au format Ecore qui est généralement considéré

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

comme un sous-ensemble du MOF. Dans la mesure où ce format ne permet que la capture de la syntaxe abstraite, nous avons utilisé les facilités du langage KerMeta afin de tisser les invariants et la sémantique opérationnelle du langage à la syntaxe abstraite.

L'idée de l'outillage était de pouvoir fournir la possibilité aux utilisateurs de créer des modèles avec différentes vues. Le framework GMF propose une chaîne d'outils permettant de générer la syntaxe concrète graphique d'un langage définit à partir d'un métamodèle Ecore et outillé par le framework EMF. Malheureusement, la notion de multi-vues dépasse les cas d'utilisation du framework GMF. Aussi, pour arriver à nos fins, nous avons dû modifier le flot de génération afin d'intégrer un navigateur de modèles (CNF) et les bons wizards pour créer les différentes vues nécessaires à la représentation et l'édition de modèles. Pour ce faire, nous intégrons dans la génération de l'outillage GMF la synchronisation entre les vues et le modèle.

Outillage de la méthodologie

L'outillage du processus proposé dans la cadre de cette thèse repose essentiellement sur les standards de l'OMG (MDA, UML, MOF, XMI) et Eclipse (EMF, ECore) :

- Le langage KerMeta [PAFJ05] développé par l'INRIA a été utilisé afin de formaliser et valider l'utilisation des métamodèles (concepts et contraintes),
- L'outil de modélisation Rhapsody a été utilisé pour modéliser l'application, les plateformes et les allocations conformément à la méthodologie présentée,
- L'outil de transformation MDWorkbench (model to model, model to text) de Sodius a été utilisé pour spécifier les transformations entre les différents niveaux d'abstraction et les générations de documents et de code.

Le générateur est délivré en tant que greffon en boîte blanche pour permettre des personnalisations de règles de transformations fournies. La figure 7.2 spécifie les interactions entre les différents outils mentionnés.

La modélisation de système temps réel embarqué nécessite un langage d'action dédié permettant d'exprimer les actions élémentaires pour la spécification du corps des opérations ou celle des gardes dans les transitions des machines à états. Le choix du langage d'action soulève la question de

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

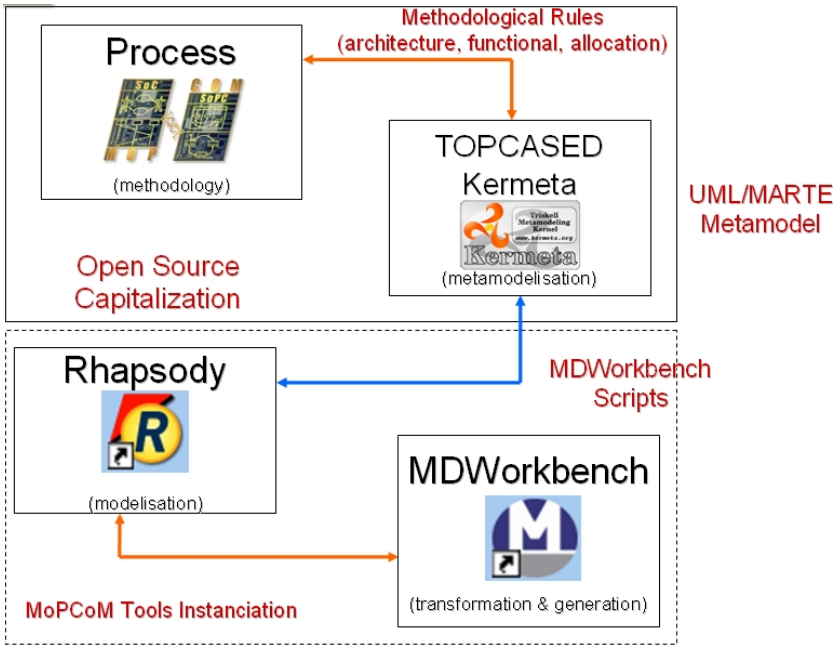


FIGURE 7.2 – Outillage MoPCoM

la nature textuelle ou graphique de sa syntaxe concrète. Dans le contexte d'UML, ce choix soulève également la question de la pertinence de disposer d'un langage généraliste ou dédié. Typiquement, le langage d'action présent dans UML n'a pas été pensé pour traiter les actions élémentaires spécifiques au matériel.

Dans la méthodologie MoPCoM, pour exprimer les actions élémentaires, notre choix s'est porté sur la syntaxe du C++ pour plusieurs raisons : d'une part, parce que c'est un langage de haut niveau avec une syntaxe facilement compréhensible, et d'autre part parce que c'est un langage flexible qui permet d'introduire de nouveaux éléments de syntaxe à travers la notion de macros. Accessoirement, ce choix nous a semblé pertinent car la plupart des outils de synthèse de haut niveau prennent du C ANSI en entrée. Par ailleurs, afin de traiter les actions dans les diverses transformation de modèles, la grammaire du langage a été remonté sous la forme d'un métamodèle.

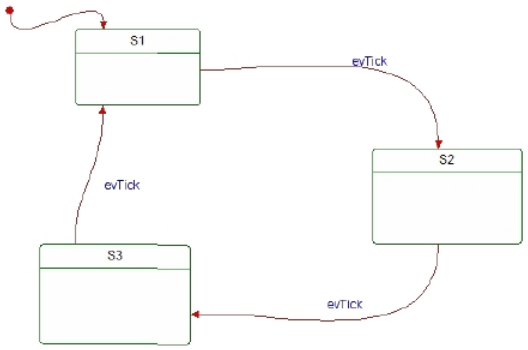
La méthodologie décrite dans cette thèse vise la génération de code VHDL. Aussi, le générateur livré avec l'outillage prend en entrée un modèle de niveau DML alloué. À partir de là, la génération de code exploite l'architecture du modèle pour générer les entités VHDL et les machines à états pour générer

© 2012 Tous droits réservés.
http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

le code du comportement des entités.

Typiquement, les ports UML sont transformés en ports VHDL au regard des protocoles de communications choisis. Les machines à états génèrent quant à elles des types énumérés (cf. figure 7.3). Selon les protocoles de communication, différentes machines à états peuvent être générées et dont le code est automatiquement inséré dans le code métier.




```
type STATECHARTOFFSM01_STS is (NOTA, S1, S2, S3);
signal statechartoffsm01_st : STATECHARTOFFSM01_STS ;
....
case statechartoffsm01_st is
  when NOTA =>
    statechartoffsm01_st <= S1;
  when S1 =>
    if (evtick) then
      statechartoffsm01_st <= S2;
    end if;
  when S2 =>
    if (evtick) then
      statechartoffsm01_st <= S3;
    end if;
  when S3 =>
    if (evtick) then
      statechartoffsm01_st <= S1;
    end if;
  when others =>
    statechartoffsm01_st <= NOTA;
end case;
```

FIGURE 7.3 – Traitement des machines à état

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Bibliographie

[Bai05] Brian Bailey. *Taxonomies for the Development and Verification of Digital Systems*. Springer, 2005.

[B.C88] N.Iscoe B.Curtis, H.Krasner. A field study of the software design process for large systems. *Communication ACM*, 31 :1268–1281, 1988.

[BCCG07] Reda Bendraou, Benoit Combemale, Xavier Cregut, and Marie-Pierre Gervais. Definition of an executable spem 2.0. In *APSEC '07 : Proceedings of the 14th Asia-Pacific Software Engineering Conference (APSEC'07)*, pages 390–397, Washington, DC, USA, 2007. IEEE Computer Society.

[BDG05] Reda Bendraou, Philippe Desfray, and Marie-Pierre Gervais. Mda components : A flexible way for implementing the mda approach. In *ECMDA-FA*, pages 59–73, 2005.

[BDGM08] Reda Bendraou, Philippe Desfray, Marie-Pierre Gervais, and Alexis Muller. Mda tool components : a proposal for packaging know-how in model driven development. *Software and System Modeling*, 7(3) :329–343, 2008.

[BDK+08] Thomas Birkhölzer, Christoph Dickmann, Harald Klein, Jürgen Vaupel, Stefan Ast, and Ludger Meyer. Customized predictive models for process improvement projects. In *PROFES '08 : Proceedings of the 9th international conference on Product-Focused Software Process Improvement*, pages 304–316, Berlin, Heidelberg, 2008. Springer-Verlag.

[Ben07] Réda Bendraou. *UML4SPM : Un Langage De Modélisation De Procédés De Développement Logiciel Exécutable Et Orienté Modèle*. PhD thesis, Université Pierre et Marie Curie, 2007.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

[ben09] Trustworthy software development processes, international conference on software process, icsp 2009 vancouver, canada, may 16-17, 2009 proceedings. In Qing Wang, Vahid Garousi, Raymond J. Madachy, and Dietmar Pfahl, editors, *ICSP*, volume 5543 of *Lecture Notes in Computer Science*. Springer, 2009.

[BG] J. Bezivin and S. Gérard. A preliminary identification of mda components.

[BGA07] B. Bailey, G.Martin, and A.Piziali. *ESL Design and Verification*. Systems on Silicon, 2007.

[BGML03] Jean Bézivin, S. Gérard, Pierre-Alain Muller, and L.Rioux. Mda components : Challenges and opportunities. In *Workshop on Metamodelling for MDA*, pages 23–41, York, England, 2003.

[BHLM02] Joseph Buck, Soonhoi Ha, Edward A. Lee, and David G. Messerschmitt. Ptolemy : a framework for simulating and prototyping heterogeneous systems. *IEEE*, 10 :527–543, 2002.

[BP] Hailpern B. and Tarr P. Model driven development : The good, the bad and the ugly. IBM Systems Journal.

[Bra87] Michael E. Bratman. *Intention, Plans, and Practical Reason*. Harvard University Press : Cambridge, 1987.

[Bre02] Erwan Breton. *Contribution à la représentation des processus par des techniques de méta-modélisation*. PhD thesis, Université de Rennes, 2002.

[Cal90] J.P. Calvez. *Spécification et Conception des Systèmes, une Méthodologie : MCSE*. Masson, 1990.

[CCH+99] Henry Chang, Larry Cooke, Merrill Hunt, Grant Martin, Andrew J. McNelly, and Lee Todd. *Surviving the SOC revolution : a guide to platform-based design*. Kluwer Academic Publishers, Norwell, MA, USA, 1999.

[CCTDDTB01] Bernard Coulette, Xavier Crégut, Thu Tran Dan, and Thuy Dong Thi Bich. Managing Processes through a Base of Reusable Components . In *ICEIS'2001 , Setubal, Portugal, 07/07/01-10/07/01*, pages 180–190. Kluwer Academic Publishers, juillet 2001.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

[CJ05] Franck Chauvel and Jean-Marc Jézéquel. Code generation from uml models with semantic variation points. In *Models UML*, 2005.

[CKS03] Mary Beth Chrissis, Mike Konrad, and Sandy Shrum. *CMMI : Guidelines for Process Integration and Product Improvement*. Addison Wesley Professional, 2003.

[CR99] A. Benjamen C. Rolland, N. Prakash. A multi-model of process modelling. *Requirement Engineering*, 4 :169–187, 1999.

[CR06] Jin Myung Choi and Sung Yul Rhew. Process component plug-in approach. In *Software Engineering Research and Practice*, pages 620–628, 2006.

[CSL+] Rong Chen, Marco Sgroi, Luciano Lavagno, Grant Martin, Alberto Sangiovanni-Vincentelli, and Jan Rabaey. Uml and platform-based design.

[DEAB08] Jean-luc Dekeyser, Anne Etien, Rabie Ben Atitallah, and Pierre Boulet. Using the uml pro le for marte to mpsoc co-design. First International Conference on Embedded Systems & Critical Applications (ICESCA'08), 2008.

[Dou02] Bruce Powel Douglass. *Real-Time Design Patterns : Robust Scalable Architecture for Real-Time Systems*. Addison-Wesley Professional, 2002.

[Dou09] Bruce Powel Douglass. *Real-Time Agility : The Harmony Method for Real-Time and Embedded Systems Development*. Addison-Wesley Professional, 2009.

[EG03] Martyn Edwards and Peter Green. Uml for hardware and software object modeling. *UML for real : design of embedded real-time systems*, pages 127–147, 2003.

[Ghe05] Frank Ghenassia. *Transaction-Level Modeling with SystemC*. Springer, 2005.

[GHJV95] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns : Elements of Reusable Object-Oriented Software*. Number ISBN 0-201-63361-2. Addison-Wesley, 1995.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

[GL09] Volker Gruhn and Ralf Laue. A heuristic method for business process model evaluation. In *CIAO ! / EOMAS*, pages 28–39, 2009.

[GMP+94] Pankaj K. Garg, Peiwei Mi, Thuan Pham, Walt Scacchi, and Gary Thunquest. The smart approach for software process engineering. In *ICSE '94 : Proceedings of the 16th international conference on Software engineering*, pages 341–350, Los Alamitos, CA, USA, 1994. IEEE Computer Society Press.

[GVNL94] Daniel Gajski, Frank Vahid, Sanjiv Narayan, and Jie Long. *Specification and Design of Embedded Systems*. Prentice Hall, 1994.

[Har08] Cécile Hardebolle. *Composition de modèles pour la modélisation multi-paradigme du comportement des systèmes*. PhD thesis, Université Paris-Sud XI, 2008.

[HSV04] Fernando Herrera, Pablo Sánchez, and Eugenio Villar. Modeling of csp, kpn and sr systems with systemc. pages 133–148, 2004.

[IGI+05] Katsuro Inoue, Pankaj K. Garg, Hajimu Iida, Ken ichi Matsumoto, and Koji Torii. Mega software engineering. In *PROFES*, pages 399–413, 2005.

[ITR07] ITRS. Design. Technical report, INTERNATIONAL TECHNOLOGY ROADMAP FOR SEMICONDUCTORS, 2007.

[Jan04] Axel Jantsch. *Modeling Embedded Systems and SoC's*. Systems on Silicon, 2004.

[KA09] Ali Koudri and Al. *MDD for RES*, chapter SoC/SoPC Development using MDD and MARTE profile, pages 254–260. Hermes, 2009.

[KAC07] Ali Koudri, Denis Aulagnier, and Joel Champeau. Fpga design based on uml/mda approach : Application to an rf emitter transceiver development. In *conf. DAC07*, June 2007.

[Kam00] Peter J. Kammer. Building the process : Component-based workflow architectures in a distributed world, 2000.

[KC10] Ali Koudri and Joel Champeau. Modal : A spem extension to improve co-design process models. In *International*

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

Conference on Software Process (ICSP10), 8-9 june, 2010, Paderborn, Germany, accepted, June 2010.

[KCA07] Ali Koudri, Joel Champeau, and Denis Aulagnier. Une sémantiques opérationelle pour une meilleure métamodélisation. In *Atelier SéMo'07*, March 2007.

[KCAS09] Ali Koudri, Joël Champeau, Denis Aulagnier, and Philippe Soulard. Mopcom/marte process applied to a cognitive radio system design and analysis. In *ECMDA-FA*, pages 277–288, 2009.

[KCLLA10] Ali Koudri, Joel Champeau, Jean-christophe Le Lann, and Denis Aulagnier. Uml/marte process for soc/sopc. In *Conférence Embedded Real Time Software and Systems (ERTS'10), 19-21 mai 2010, Toulouse*, May 2010.

[KCLLL10] Ali Koudri, Joel Champeau, Jean-christophe Le Lann, and Vincent Leilde. Mopcom methodology : Focus on models of computation. In *6th European Conference on Modelling Foundations and Applications (ECMFA10), 15-18 june, 2010, Paris*, June 2010.

[Kou96] A. A. Kountouris. Safe and efficient elimination of infeasible execution paths in wcet estimation. In *RTCSA '96 : Proceedings of the Third International Workshop on Real-Time Computing Systems Application (RTCSA '96)*, page 187, Washington, DC, USA, 1996. IEEE Computer Society.

[kou09] Génie logiciel, 2009.

[Kra03] Jerry Krasner. Embedded software development issues and challenges. *Embedded Market Forecasters*, Embedded Market Forecasters :1–5, 2003.

[KRF09] Harald Klein, Andreas Rausch, and Edward Fischer. Collaboration in global software engineering based on process description integration. In *CDVE*, pages 1–8, 2009.

[KVS+08] Ali Koudri, Didier Vojtsiek, Philippe Soulard, Christophe Moy, Joel Champeau, Jorgiano Vidal, and Jean-christophe Le Lann. Using marte in the mopcom soc/sopc methodology. In *workshop MARTE*, March 2008.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

[KWB03] Anneke Kleppe, Jos Warmer, and Wim Bast. *MDA Explained : The Model Driven Architecture : Practice and Promise*. Addison-Wesley Professional, 2003.

[Lah06] Vesa Lahtinen. System level design experiences and the need for standardization. In *System-on-Chip, 2006. International Symposium on*, 2006.

[LDM06] Larrucea, Diez, and Mansell. Practical model driven development process, 2006.

[Lon93] Jacques Lonchamp. A structured conceptual and terminological framework for software process engineering. In *ICSP*, pages 41–53, 1993.

[Mad91] Nazim H. Madhavji. The process cycle. *Softw. Eng. J.*, 6(5) :234–242, 1991.

[MBLT06] Jean-Marie Mottu, Benoit Baudry, and Yves Le Traon. Reusable mda components : A testing-for-trust approach. In *proceedings of the MoDELS/UML 2006*, Genova, Italy, October 2006.

[Mek08] François Mekerke. *Structuration des modèles orientés métiers pour les systèmes embarqués*. PhD thesis, Lisyc, 2008.

[mF04] Jean marie Favre. Towards a basic theory to model model driven engineering. In *In Workshop on Software Model Engineering, WISME 2004, joint event with UML2004*, 2004.

[MFB09] Pierre-Alain Muller, Frédéric Fondement, and Benoit Baudry. Modeling modeling. In *MoDELS*, pages 2–16, 2009.

[MJ00] III Mitola Joseph. *Cognitive Radio, An Integrated Agent Architecture for Software Defined Radio*. PhD thesis, Royal Institue of Technology, 2000.

[MoP07] MoPCoM. Mopcom soc/sopc project. http://www.mopcom.fr, 2007.

[M.S03] M.Smith. Do-178b cost and benefits : a detailed analysis, 2003.

[NN03] Farzad Nekoogar and Faranak Nekoogar. *From ASICs to SOCs : A Practical Approach*. Prentice Hall, 2003.

[ocp] Open core protocol - international partnership.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

[ofbr09] omitted for blind review. omitted for blind review. In *Model Driven Architecture, Foundations and Applications*, 2009.

[OMG03] OMG. Mda guide version 1.0.1. Technical report, Object Management Group, 2003.

[OMG05a] OMG. Mof qvt. Technical Report ptc/05-11-01, Object Management Group, 2005.

[OMG05b] OMG. UML 2.0 superstructure. Technical Report formal/05-07-04, Object Management Group, 2005.

[OMG05c] OMG. Uml profile for schedulability, performance, and time, version 1.1. Technical Report formal/2005-01-02, Object Management Group, 2005.

[OMG06] OMG. Uml for system on chip. Technical Report formal/05-07-04, Object Management Group, 2006.

[OMG08] OMG. Systems modeling language specification v1.1. Technical Report ptc/2008-05-16, Object Management Group, 2008.

[OMR09] Alexis Ocampo, Jürgen Münch, and William E. Riddle. Incrementally introducing process model rationale support in an organization. In *ICSP '09 : Proceedings of the International Conference on Software Process*, pages 330–341, Berlin, Heidelberg, 2009. Springer-Verlag.

[osc] Open systemc initiative.

[Ost09] Leon J. Osterweil. Formalisms to support the definition of processes. *J. Comput. Sci. Technol.*, 24(2) :198–211, 2009.

[PAFJ05] Muller Pierre-Alain, Franck Fleurey, and Jean-Marc Jézéquel. Weaving executability into object-oriented meta-languages. In *Proceedings of MODELS/UML'2005*, Montego Bay, Jamaica, 2005.

[PAM+08] Eric Piel, Rabie Ben Attitalah, Philippe Marquet, Samy Meftali, Smaïl Niar, Anne Etien, Jean-Luc Dekeyser, and Pierre Boulet. Gaspard2 : from marte to systemc simulation. March 2008.

[RCGT09] Ansgar Radermacher, Arnaud Cuccuru, Sebastien Gerard, and François Terrier. Generating execution infrastructures

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

for component-oriented specifications with a model driven toolchain : a case study for marte's gcm and real-time annotations. In *GPCE '09 : Proceedings of the eighth international conference on Generative programming and component engineering*, pages 127–136, New York, NY, USA, 2009. ACM.

[Rev08] Sébastien Revol. *Profil UML pour TLM : contribution à la formalisation et à l'automatisation du flot de conception et vérification des systèmes-sur-puce.* PhD thesis, INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE, 2008.

[Rol98] Colette Rolland. A comprehensive view of process engineering. *Lecture Notes in Computer Science*, 1413 :1–24, 1998.

[RSRB05] E. Riccobene, P. Scandurra, A. Rosti, and S. Bocchio. A soc design methodology involving a uml 2.0 profile for systemc. In *Proc. of the conference on Design, Automation and Test in Europe*, pages 704–709, Munich, Germany, March 2005. IEEE Computer Society.

[Sel03] Bran Selic. Models, software models and uml. *ACM*, 13 :1–16, 2003.

[SOD] SODIUS. Mdworkbench platform. http://www.mdworkbench.com.

[spi] Spirit consortium.

[sta06] standishgroup. Standish group chaos report, 2006.

[STE07] James Richard Heron STEEL. *Typage de modèles.* PhD thesis, Université de Rennes, 2007.

[SVSS+09] Alberto Sangiovanni-Vincentelli, Sandeep Kumar Shukla, Janos Sztipanovits, Guang Yang, and Deepak A. Mathaikutty. Metamodeling : An emerging representation paradigm for system-level design. *IEEE Des. Test*, 26(3) :54–69, 2009.

[Tah08] Safoaun Taha. *Modélisation Conjointe Logicielle / Matérielle de Systèmes Temps Réel.* PhD thesis, Université de Lille, 2008.

[Tho08] Frédéric Thomas. *Contribution à la prise en compte des plates-formes logicielles d'exécution dans une ingénierie gé-*

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr

Thèse d'Ali Koudri, Lille 1, 2010

nérative dirigée par les modèles. PhD thesis, Université d'Evry, 2008.

[TLB+09] Thomas Tan, Qi Li, Barry Boehm, Ye Yang, Mei He, and Ramin Moazeni. Productivity trends in incremental and iterative software development. In *ESEM '09 : Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement*, pages 1–10, Washington, DC, USA, 2009. IEEE Computer Society.

[VP02] Dániel Varró and András Pataricza. Metamodeling mathematics : A precise and visual framework for describing semantics domains of UML models. In *Proc. Fifth International Conference on the Unified Modeling Language – The Language and its Applications*, pages 18–33. Springer-Verlag, September 30 – October 4 2002.

[WLH05] Yasha Wang, Dongni Li, and Xiaoyang He. A process management tool supporting component-based process development and hierarchical management mechanism. In *CIT*, pages 906–910, 2005.

[Woo00] Michael J. Wooldridge. *Reasoning about Rational Agents.* The MIT Press, Cambridge, Massachusetts, 2000.

[WZZ+06] T Wang, X.G. Zhou, B Zhou, L Liang, and C.L. Peng. A mda based soc modeling apporach using uml and systemc. In *The Sixth IEEE International Conference on Computer and Information Technology (CIT'06)*, Seoul, KOREA, September 2006. IEEE Computer Society.

[Yin03] Robert K. Yin. *Case Study Research : Design and Methods.* Sage Publication, Inc, 2003.

[ZKJ09] He Zhang, Barbara Kitchenham, and Ross Jeffery. Qualitative vs. quantitative software process simulation modeling : Conversion and comparison. In *ASWEC '09 : Proceedings of the 2009 Australian Software Engineering Conference*, pages 345–354, Washington, DC, USA, 2009. IEEE Computer Society.

© 2012 Tous droits réservés.

http://doc.univ-lille1.fr