Avant-propos

Les systèmes informatiques n'ont jamais été autant au centre de la stratégie des entreprises qu'aujourd'hui. Les fonctionnalités qu'ils offrent, leur facilité d'utilisation, leur fiabilité, leur performance et leur robustesse sont les qualités reines qui permettent aux entreprises d'être compétitives. L'ingénierie du logiciel fournit sans cesse de nouvelles technologies facilitant la mise en œuvre de ces systèmes tout en accroissant leurs qualités. La dernière en date de ces technologies est certainement l'approche par composant, même si nous voyons déjà pointer l'approche par service.

Le revers de la médaille de ces technologies, aussi appelées plates-formes d'exécution, est qu'elles complexifient énormément les systèmes informatiques. Cela met les entreprises dans une situation inconfortable, car elles hésitent entre adopter une nouvelle plate-forme et subir le coût de la migration ou ne pas l'adopter et prendre le risque de voir des concurrents devenir plus compétitifs grâce au choix inverse.

Cet inconvénient majeur est un frein à l'évolution. Sa source réside dans la complexité des plates-formes d'exécution, pourtant conçues pour faciliter le développement et la maintenance des systèmes. Pour y faire face, il était nécessaire de définir une approche permettant de dompter la complexité. Cette approche se devait d'être flexible et générique afin de pouvoir s'adapter à tout type de plate-forme.

L'ingénierie logicielle guidée par les modèles, ou MDA (Model Driven Architecture), correspond à la définition d'une telle approche. MDA applique la séparation des préoccupations entre la logique métier des systèmes informatiques et les plates-formes utilisées et se fonde sur l'utilisation massive des modèles.

MDA consiste à élaborer les modèles de la logique métier des systèmes de façon indépendante des plates-formes d'exécution puis à transformer ces modèles automatiquement vers des modèles dépendant des plates-formes. La complexité des plates-formes n'apparaît plus dans les modèles de la logique métier mais se retrouve dans la transformation.

Les avantages de MDA sont donc la pérennisation de la logique métier de l'entreprise grâce à l'élaboration de modèles, la productivité de cette logique métier grâce à l'automatisation des transformations de modèles et la prise en compte des plates-formes d'exécution grâce à l'intégration de celles-ci dans les transformations de modèles.

Objectifs de l'ouvrage

Cet ouvrage décrit tous les standards, techniques et frameworks qui composent MDA et présente son architecture socle, qui définit les concepts de modèle, métamodèle et méta-métamodèle.

Après une présentation détaillée des standards phares de MDA, tels que UML ou OCL, nous insistons sur les aspects pratiques de sa mise en œuvre. Nous expliquons en particulier le fonctionnement de JMI et EMF, qui permettent de programmer en Java des traitements sur les modèles, tels que les transformations. Nous présentons ensuite Rational Software Modeler et MDA Modeler, deux outils du marché qui supportent MDA. Pour finir, nous montrons comment MDA prend en compte les plates-formes d'exécution. Nous détaillons en particulier la prise en compte des plates-formes J2EE/EJB et PHP.

Cet ouvrage s'efforce d'offrir une vue complète de MDA et l'illustre par une étude de cas. Cette dernière consiste à développer une application de commerce électronique selon les principes MDA. Cette étude de cas vise à mettre en lumière les avantages de MDA, à commencer par la gestion de la complexité des plates-formes.

Cet ouvrage aura atteint son but s'il aura permis de comprendre la philosophie de MDA, ses concepts fondateurs et son état d'implémentation actuel.

Organisation de l'ouvrage

- Le chapitre 1 est un parcours macroscopique de l'ensemble des standards, techniques et frameworks de MDA. Nous exposons la philosophie de l'approche et expliquons l'intérêt des modèles en précisant leur rôle. Nous définissons en outre les qualités intrinsèques des modèles que sont la pérennité, la productivité et la prise en compte des plates-formes. Ces trois qualités serviront de grille de lecture pour la suite de l'ouvrage.

La première partie est consacrée à la pérennité des savoir-faire offerte par MDA :

- Le chapitre 2 introduit le standard MOF, qui permet d'élaborer des métamodèles. Nous détaillons ce qu'est un métamodèle et expliquons comment les concevoir. Ce chapitre nous permettra ensuite de présenter les différents métamodèles qui constituent MDA.
- Le chapitre 3 est dédié au standard UML, qui est le langage préconisé pour élaborer les PIM. Nous nous attardons sur une partie du métamodèle UML afin de bien faire saisir la place que prend UML dans MDA. Nous introduisons également la notion de profil UML.
- Le chapitre 4 présente les standards AS (Action Semantics) et OCL (Object Constraint Language), qui permettent de spécifier le corps des méthodes UML. Ces standards sont indépendants des langages de programmation. Nous donnons une partie de leur métamodèle et expliquons leur lien avec UML dans MDA.
- Le chapitre 5 est consacré au standard XMI (XML Data Interchange), qui permet de représenter les modèles sous forme de documents XML. Ce standard est d'une

importance capitale pour la pérennité des modèles MDA car c'est grâce à lui que les modèles peuvent être concrètement échangés.

La partie II se penche sur les gains de productivité apportés par les modèles MDA :

- Le chapitre 6 présente les concepts de manipulation de modèles, avec notamment le standard JMI (Java Metadata Interface). Nous introduisons à cette occasion EMF, qui est la plate-forme de métamodélisation d'Eclipse *(http://www.eclipse.org/emf).*
- Le chapitre 7 détaille les concepts à l'œuvre dans la transformation des modèles. Nous présentons les différentes façons d'élaborer des transformations de modèles avec les approches par programmation, par template et par modélisation (MOF2.0 QVT).
- Le chapitre 8 examine les techniques MDA disponibles dans les outils du marché. Nous nous attardons en particulier sur les outils Rational Software Modeler et Softeam MDA Modeler.

La partie III est consacrée à la prise en compte des plates-formes par MDA :

- Le chapitre 9 introduit les concepts de plates-formes tels que définis dans MDA et présente les différentes façons de prendre en compte les plates-formes d'exécution.
- Le chapitre 10 est consacré à la plate-forme J2EE/EJB. Nous expliquons sa prise en compte dans MDA grâce à la définition d'un profil UML. Cette prise en compte sera utilisée au chapitre 12 qui détaille l'étude de cas.
- Le chapitre 11 traite de la plate-forme PHP. Nous expliquons sa prise en compte dans MDA grâce à la définition d'un métamodèle spécifique de cette plate-forme. Cette prise en compte sera elle aussi utilisée au chapitre 12.

La partie IV illustre la mise en œuvre des principes de MDA dans une étude de cas :

- Le chapitre 12 entre dans le détail de l'utilisation de l'approche MDA pour la réalisation de l'étude de cas PetStore. Nous expliquons chacune des étapes de construction de l'application PetStore à la lumière des principes MDA. La fin du chapitre synthétise les principaux apports de l'approche MDA.

La partie V est constituée d'une annexe détaillant le contenu du CD offert avec l'ouvrage.

À qui s'adresse l'ouvrage

Cet ouvrage est destiné à toute personne désirant connaître les avantages et les limites de MDA.

Le développeur trouvera les réponses aux questions qu'il se pose sur l'intérêt des langages de modélisation et sur leurs capacités notamment en terme de production (génération de code).

Le concepteur de modèles, tels que les modèles UML, pourra élargir sa vision quant aux avantages offerts par les modèles. Il pourra notamment découvrir les concepts de

métamodèle, de transformation de modèle et de modèle de plate-forme et ainsi mieux comprendre la place qu'occupe UML dans MDA.

Le décideur et l'architecte pourront comprendre tous les enjeux de MDA. Ils seront à même, grâce à la présentation des outils de support de MDA et à l'étude de cas, de savoir quand et comment s'approprier cette approche.